HAL
open science

Système multi-agents centré sur les compétences

Olivier Chator

To cite this version:

Olivier Chator. Système multi-agents centré sur les compétences. Informatique. Université de Bordeaux, 2015. Français. NNT : 2015BORD0009 . tel-01212471

HAL Id: tel-01212471

https://theses.hal.science/tel-01212471v1

Submitted on 6 Oct 2015

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

THÈSE

PRÉSENTÉE A

L'UNIVERSITÉ de BORDEAUX

ÉCOLE DOCTORALE DES SCIENCES PHYSIQUES ET DE L'INGENIEUR

Par Olivier CHATOR

POUR OBTENIR LE GRADE DE

DOCTEUR

SPÉCIALITÉ : Automatique, productique, signal et image, ingénierie cognitique

Système Multi-Agents centré sur les Compétences

Directeur de recherche : Jean-Marc SALOTTI

Soutenue le : 27 Mars 2015

Devant la commission d'examen formée de :

M. Luigi LANCIERI	Professeur	Université de Lille 1	Rapporteur
M. Jean-Louis ERMINE	Professeur	Institut Mines-Télécom	Rapporteur
M. Vincent LIQUETE	Professeur	Université de Bordeaux	Président
M. Jean-Marc SALOTTI	Professeur	Bordeaux INP	Examinateur
M. Pascal SALEMBIER	Professeur	Université de Troyes	Examinateur
M. Pierre-Alexandre FAVIER	Maître de Conférences	Bordeaux INP	Examinateur

Dédicace

Serrez vos grands-parents dans vos bras et dites : « Merci d'avoir fait ce que tu as fait pour que je puisse exister ! ».

Alex Haley

Personne ne peut faire pour les enfants ce que font les grands-parents. Ceux-ci répandent une espèce de poudre d'étoiles sur leurs vies.

Alex Haley

A Renée & Lucien

Remerciements

Heinrich disait : « il a suffi que 2 hommes se haïssent pour que la haine, de proche en proche, gagne tout l'univers ».

Et moi, je le dis, en vérité il suffit qu'un homme aime tous les hommes d'un amour sans partage pour que cet amour s'étende de proche en proche à toute l'humanité.

Jean-Paul SARTRE

Je vais certainement omettre de citer quelqu'un dans ce paragraphe, mais je n'oublie pourtant personne. Je remercie tout particulièrement :

- M. Luigi LANCIERI et M. Jean-Louis ERMINE d'avoir accepté d'être les rapporteurs de ma thèse
- M. Vincent LIQUETE, M. Pierre-Alexandre FAVIER et M. Pascal SALEMBIER pour avoir accepté d'en être les examinateurs
- M. Jean-Marc SALOTTI, mon Directeur de thèse, pour la confiance qu'il m'a témoignée et la persévérance dont il a fait preuve pour m'amener aujourd'hui à bon port. Je ne me risquerai pas ici à reprendre toute l'histoire, mais le démarrage d'une thèse fut un parcours semé d'embuches, et ce sur plusieurs années… Pourtant, il a toujours été présent et disponible lorsque j'ai fait appel à lui. Qu'il en soit sincèrement et chaleureusement remercié.

Merci à toute la chaîne de direction du Conseil Général de la Gironde sans qui la réalisation de cette thèse n'aurait pas été envisageable : M. Philippe MADRELLE (son Président), M. Gérard MARTY puis M. Laurent CARRIÉ (son Directeur Général des Services), M. Marc FAUVEAU (Directeur Général Adjoint aux Finances, à l'informatique et aux affaires juridiques).

Je souhaiterais remercier aussi Mme Julie CHABAUD (Directrice de la Mission Agenda 21) et Mme Nathalie LAURENT (Directrice du Système d'Information), toutes deux pour m'avoir permis de mener ces travaux à bien.

Merci également à M. Frédéric BEVIA (mon chef de service) et à Mme Julie ROTURIER (mission Agenda 21) pour leurs encouragements tout au long de ces 3 années.

Merci à toutes celles et ceux qui ont souffert souvent en silence, à mes côtés, lors de ces années de rude combat... Et croyez-moi, ils ont eu du mérite ! Je pense particulièrement à tous les membres de mon entourage familial. Mention spéciale et des plus affectueuses pour Sylvie, mon épouse, chère à mon cœur. Une pensée particulière pour Patricia qui a passé de longues heures à relire ce mémoire.

Merci également à Stéphane, Philippe, Jean-Paul et leurs compagnes, mes amis fidèles depuis tant d'années, d'être tout simplement tels qu'ils sont…

Enfin, plus légèrement, merci, du fond du cœur au conseiller d'orientation que j'ai rencontré en fin de terminale, au Lycée Pierre Caraminot d'Egletons (Corrèze), en 1989. Lors d'une rencontre aussi chaude que les -273 degrés Celsius du zéro absolu, ce brave homme que la nature a doté d'un flair extraordinaire et d'une diplomatie digne de ce que l'on enseigne à Normale Sup, me dit tout de go : « *Mon pauvre Monsieur, vous n'aurez jamais votre bac ! Quant à envisager des études supérieures…* ». CQFD !

Olivier CHATOR

Table des matières

THÈSE 1
PRÉSENTÉE A 1
L'UNIVERSITÉ de BORDEAUX 1
ÉCOLE DOCTORALE DES SCIENCES PHYSIQUES ET DE L'INGENIEUR 1
Par Olivier CHATOR 1
Système Multi-Agents centré sur les Compétences 1
Soutenue le : 27 Mars 2015 1
Introduction 17
Contexte de cette thèse 17
Problématique 21
Cadre de réponse théorique proposé 23
Structure de ce mémoire de thèse 24
1 CHAPITRE 1 - Etude de l'existant 25
1.1 Introduction 25
1.2 Problématique 25
1.2.1 Le cas CG33 : spécifications du besoin 25
1.2.2 Généralisation 26
1.3 Analyse & proposition de réponse 27
1.4 Définitions, concepts & notions clé 30
1.4.1 Connaissance 30
1.4.2 Savoir 32
1.4.3 Expérience 32
1.4.4 Savoir-faire 33
1.4.5 Compétence 33
1.4.6 Compétence élémentaire 35
1.4.7 Domaine de compétence 35
1.4.8 Relations entre compétences 36
1.5 Systèmes multi-agents 38
1.5.1 Intérêt des SMA pour notre problématique 38
1.5.2 Paradigme agent 39
1.6 Gestion des compétences et SMAs 43
1.7 Panorama de méthodes d'apprentissage dans les SMAs 46
1.7.1 Pourquoi exploiter des techniques d'apprentissage ? 46
1.7.2 Définition 47
1.7.3 Les différents modes d'apprentissage 47
1.7.4 L'apprentissage dans un SMA 48
1.7.5 Processus Décisionnel de Markov 49
1.7.6 Apprentissage par Renforcement 50
1.7.7 La théorie des jeux 52
1.7.8 Apprentissage bayésien 55
1.7.9 Case Base Learning 56
1.7.10 Apprentissage symbolique 59
1.7.11 Apprentissage incrémental 63

1.7.12 Apprentissage statistique 64
2 CHAPITRE 2 – Modèle théorique des agents compétence 67
2.1 Rappel synthétique de notre problématique 67
2.2 Analyse & proposition de réponse 68
2.3 Fondements théoriques 71
2.4 Définitions 71
2.4.1 Environnement ou contexte 71
2.4.2 Objectif et projet 71
2.4.3 Domaine d'objectif 73
2.4.4 Critère d'efficacité et compétence élémentaire 73
2.5 Modèle agent proposé 77
2.5.1 Introduction 77
2.5.2 Proposition 78
2.5.3 Modèle de données 79
2.5.4 Les 3 composantes clé 83
2.5.5 Ressources 87
2.5.6 Cycle de vie 88
2.5.7 Modèle comportemental 91
3 CHAPITRE 3 – Implémentation & Résultats 109
3.1 Introduction 109
3.2 Architecture globale 110
3.3 Couche N°1 : Application Web côté utilisateur 111
3.4 Couche N°2 : Base de données 113
3.5 Couche N°3 : Le SMA (Moteur de R&D) 113
3.5.1 Généralités 113
3.5.2 Arborescence des composants logiciels 114
3.5.3 Framework de développement des plugins comportementaux 120
3.5.4 Lancement, arrêt et mécanisme de traces du moteur de R&D 121
3.5.5 Traces & fichiers de log 122
3.6 Mise en œuvre du cycle de vie 122
3.7 Mise en œuvre du modèle comportemental 123
3.8 Mémorisation 124
3.9 Comportement N°1 : s'auto définir 126
3.10 Comportement N°2 : sélectionner des acteurs 126
3.10.1 Introduction 126
3.10.2 Ecran de paramétrage de la demande utilisateur 127
3.10.3 Echanges de flux 128
3.10.4 Apprentissage 129
3.11 Comportement N°3 : identifier des liens inter ACs 130
3.11.1 Introduction 130
3.11.2 Cas N°1 : Demande explicite de l'utilisateur 131
3.11.3 Cas N°2 : identification récurrente des liens 133
3.11.4 Apprentissage 136
3.12 Comportement N°4 : participer à un nouveau projet 139
3.12.1 Introduction 139
3.12.2 Cas N°1 : utiliser un objectif existant 139

3.12.3 Cas N°2 : proactivité des Agents Compétence 143
3.13 Simulateur automatisé de sélection des acteurs 147
3.13.1 Introduction 147
3.13.2 Scénarios de simulation 147
3.13.3 Eléments clés de simulation 149
3.13.4 Résultats obtenus 149
3.13.5 Proposition d'interprétation 150
3.14 Exercice d'application 151
3.14.1 Contexte 151
3.14.2 Scénarios de simulation globale 158
3.14.3 Résultats expérimentaux 159
Conclusion & perspectives 171
Discussion 171
Conclusion 172
Perspectives 175
Valorisation de la Recherche 177
Résumé de thèse 193
Thesis summary 195

Table des tableaux

Tableau 1 : Les différents types d'agents 40
Tableau 2 : Illustration du contenu d'un enregistrement de la table des agents 81
Tableau 3 : Illustration du contenu d'un enregistrement de la table mémoire d'un agent 82
Tableau 4 : Grammaire XML du modèle comportemental 96
Tableau 5 : Structure arborescente de plus haut niveau du moteur de R&D 114
Tableau 6 : Structure arborescente du répertoire « classes » 115
Tableau 7 : Contenu du répertoire « classes.org.cg33.sma.core » 115
Tableau 8 : Structure arborescente du répertoire « conf » 116
Tableau 9 : Structure arborescente du répertoire « script » 117
Tableau 10 : Structure arborescente du répertoire « src » 117
Tableau 11 : Contenu du répertoire « src.org.cg33.sma.core » 118
Tableau 12 : Structure arborescente du répertoire « plugins » 119
Tableau 13 : Structure arborescente d'un plugin capteur ou effecteur 120
Tableau 14 : Contenu d'un enregistrement de la table mémoire d'un agent 125
Tableau 15 : Sélection des acteurs : tableau détaillé des flux échangés 129
Tableau 16 : Recherche de liens inter ACs : cas 1 : échanges de flux 132
Tableau 17 : Recherche de liens inter ACs : cas 2 : échanges de flux 136
Tableau 18 : Créer un nouveau projet : cas 1 : échanges de flux 142
Tableau 19 : Créer un nouveau projet : cas 2 : échanges de flux 145
Tableau 20 : Simulateur de sélection d'acteurs : détail des scénarios 148
Tableau 21 : Compétences élémentaires pour chaque poste de rugby 153
Tableau 22 : Compétences élémentaires pour chaque poste de rugby 157
Tableau 23 : Simulation globale : coefficients de proximité 161
Tableau 24 : Simulation globale : chiffres globaux 164

Table des illustrations

Figure 1 Les principes de la Bourse aux Compétences 29
Figure 2 Le modèle AIK (Ermine, 2010) 32
Figure 3 Modèle multi-agents générique 40
Figure 4 Les différentes caractéristiques du comportement d'un agent 42
Figure 5 MAGIQUE : un SMA récursif 44
Figure 6 Architecture d'un SMA de E-Learning 45
Figure 7 Vue du réseau de transitions d'un planning négocié 46
Figure 8 Vue schématique de la théorie de la mémoire dynamique (Shank, 1982) 59
Figure 9 Les principes de la Bourse aux Compétences 70
Figure 10 Vue partielle des tables du modèle de données 80
Figure 11 Echanges de flux dans notre SMA 84
Figure 12 Etablissement de liens inter-agents : profondeur & similarités 103
Figure 13 Architecture globale du système 110
Figure 14 Portail utilisateur de la Bourse aux Compétences 112
Figure 15 Sélection des acteurs : paramétrage de la demande utilisateur 127
Figure 16 Sélection des acteurs : échanges de flux 128
Figure 17 Sélection des acteurs : écran de validation de la décision de l'AC 130
Figure 18 Recherche de liens inter ACs : cas 1 : paramétrage de la demande utilisateur 131
Figure 19 Recherche de liens inter ACs : cas 1 : échanges de flux 131
Figure 20 Recherche de liens inter ACs : cas 2 : mail avec propositions 134
Figure 21 Recherche de liens inter ACs : cas 2 : échanges de flux 135
Figure 22 Recherche de liens inter ACs : écran de validation des propositions 137
Figure 23 Créer un nouveau projet : cas 1 : paramétrage de la demande utilisateur 140
Figure 24 Créer un nouveau projet : cas 1 : échanges de flux 141
Figure 25 Créer un nouveau projet : cas 1 : constitution interactive du nouveau projet 142
Figure 26 Créer un nouveau projet : cas 2 : paramétrage de la demande utilisateur 143
Figure 27 Créer un nouveau projet : cas 2 : échanges de flux 144
Figure 28 Créer un nouveau projet : cas 2 : constitution interactive du nouveau projet 146
Figure 29 Simulateur de sélection d'acteurs : paramétrage de la demande utilisateur 147
Figure 30 Simulateur de sélection d'acteurs : variation des poids des critères d'efficacité 149
Figure 31 Simulateur de sélection d'acteurs : convergence des valeurs de poids 150
Figure 32 Simulateur de sélection d'acteurs : stabilité des pondérations calculées 150
Figure 33 Interface de l'outil pour Vernon COTTER, entraineur de l'ASM 152
Figure 34 Les 15 postes composant une équipe de rugby 153
Figure 35 Détails d'un poste de joueur 154
Figure 36 Liste des matchs de l'ASM pour la saison 2013/2014 155
Figure 37 Détails d'un match (projet) 156
Figure 38 Evaluation d'un poste, exercé par un joueur et pour un match 156
Figure 39 Simulation globale : agents présents dans JADE au lancement de simulation 160
Figure 40 Simulation globale : mails générés automatiquement par les ACs 161
Figure 41 Résultats de simulation globale – 1 saison 167
Figure 42 Résultats de simulation globale – 5 saisons 168

Avant-propos

« Il faut faire de la vie un rêve et faire d'un rêve une réalité »

Pierre curie

« Fais-le… ou ne le fais pas…, mais il n'y a pas d'essai »

Maître Yoda

Faire une thèse

3 mots. Ils raisonnent en boucle de manière lancinante et récurrente dans mon cœur et mon esprit, depuis la fin de mon D.U.T. en informatique à l'âge de 21 ans. Au moment où j'ose enfin me présenter devant le jury de mes pairs, j'en ai 45, soit 24 ans plus tard. La belle affaire ! Y aurait-il un âge limite pour enfin réaliser son rêve, concrétiser un désir aussi profond qui vous a tenaillé des années durant ? Y aurait-il une raison valable pour empêcher l'esprit du chercheur qui m'anime depuis mes plus jeunes années de s'exprimer enfin et de se révéler ? Y aurait-il une forme d'indécence particulière à ressentir au fond de soi une certaine fierté personnelle d'avoir été membre un temps d'organismes de recherche (CNRS, IPB, IMS, ENSC…) aussi reconnus pour le prestige de leurs travaux ? A chacune de ces questions, permettez-moi d'oser répondre par la négative. Je n'aurais pas fait qu'essayer, je l'aurais fait. Ma contribution, petite goutte d'eau dans l'océan de la Science, résidera dans les lignes que vous avez sous les yeux. Je les considère de manière toute personnelle comme mon « *morceau d'architecture* ». Il est l'aboutissement d'un long travail, semé de doutes, de remises en question et d'embuches de toutes sortes, mais ô combien merveilleux à réaliser.

Comme toute publication scientifique, ce mémoire présente ma propre vision des choses. Il sera et devra être débattu, amendé, critiqué sur sa forme et sur son fond. Il ouvre en fait bien plus de perspectives de recherche qu'il n'apporte de réponses. Il tente toutefois, modestement, d'apporter sa pierre à un vaste chantier : celui de la recherche de solutions pour améliorer la collaboration des hommes au sein des projets qui les unissent. Ingénieur de formation, ces 3 années au cours desquelles je suis un chercheur à part entière me font voyager en Europe. En même temps que j'enrichis mes travaux de thèse par les échanges fructueux que j'ai avec des scientifiques éminents, je ressens que je me transforme, que j'ouvre les yeux, que j'évolue, mais cette fois sur un plan plus intime et plus profond.

Quitter la matière, la technique pure. Se plonger à corps perdu dans une réflexion intense, à un niveau d'abstraction plus élevé, où l'esprit prend finalement le pas sur la matière. Je me suis tant interrogé sur ce chemin devenu quelque part initiatique... Où ces travaux vont-ils te

mener ? Que vas-tu découvrir ? Vas-tu trouver quelque chose au bout ? Je peux le dire à présent : oui j'ai trouvé quelque chose. Au bout du compte, le résultat de ces 3 ans de recherche est une rencontre à laquelle je ne m'attendais pas : une rencontre avec moi-même…et nous avons enfin pu faire la paix.

Conventions typographiques

AC	Agent Compétence
BCDD	Bourse de Compétences des projets de Développement Durable
CDA21	Comité départemental des Agendas 21
CE	Compétence élémentaire
DD	Développement Durable
DSI	Direction du Système d'Information
ENSC	Ecole Nationale Supérieure de Cognitique
FEDER	Fonds Européen de Développement Régional
CG33	Conseil Général de la Gironde
IA	Intelligence Artificielle
RAPC	Raisonnement A Partir de Cas
SMA	Système Multi-Agents

Les éléments du texte en *italique* (hormis les titres de chapitres et paragraphes) sont extraits directement de publications ou de citation de publications d'auteurs.

Introduction

« [Les machines] s'acquittent parfaitement de tâches fastidieuses et bien définies comme construire des étoiles ou détruire des planètes. Mais donnez-leur quelque chose d'ardu, comme de consoler une veuve, et elles ne sont plus rien. Le croiriez-vous, le plus gros ordinateur de notre section est capable de faire le paysage d'une planète entière ; et pourtant il ne saurait pas faire cuire un œuf ou pousser une chanson, et en éthique il en sait moins qu'un louveteau qui vient de naître. Vous voudriez que ce soit ça qui dirige votre vie ? »

Jean-Paul SARTRE

Contexte de cette thèse

Avec 10 725 Km^2, la Gironde est, en superficie, le plus grand département Français (CG33-1, 2013). Au 1er Janvier 2012, elle comptait 1 464 088 habitants répartis sur 542 communes et 63 cantons. Le Conseil Général de la Gironde (CG33) est une collectivité territoriale, une administration, qui met en œuvre les politiques décidées par des élus. Cette collectivité est placée sous l'autorité d'un Président et dirigée par un Directeur Général des Services qui s'appuie sur des Directeurs Généraux Adjoints. Le CG33 dispose en 2013 d'un budget de 1579,4 millions d'euros, en hausse de 2,1% par rapport à 2012. 6500 agents, présents sur tout le territoire girondin, y travaillent. Ce nombre important fait du CG33 le second employeur du département.

Les missions de cette collectivité territoriale s'inscrivent dans le quotidien de tous les citoyens du département à travers différents domaines. Quelques illustrations :

- Aide sociale au travers de la gestion de l'Allocation Personnalisé d'Autonomie (APA), du paiement de la Prestation Compensatoire du Handicap (PCH).
- Actions en faveur de l'enfance, de la famille, de l'insertion, du droit à la santé...
- Education et jeunesse (gestion de 103 collèges publics).
- Gestion et entretien des routes (6490 km de routes, 340 km de pistes cyclables et 1800 ouvrages d'art).
- Gestion de transports (réseau Transgironde et Ferries maritimes pour traverser l'estuaire).

- La mise en place de politiques liées au développement de l'économie (Soutien à la création, reprise et transmission d'entreprises) et de la vie associative (Maison Départementale des Sports et de la Vie Associative).
- La culture (organisation de manifestations culturelles à travers tout le territoire girondin).
- Le Service Départemental d'Incendie et de Secours.

Parmi les missions du CG33 figure également l'induction de comportements (auprès des citoyens, acteurs économiques et institutionnels), la définition de plans d'action dans de multiples domaines et la mise en œuvre concrète de politiques autour des thématiques du Développement Durable (DD). Cette mission se concrétise sous la forme de ce que l'on nomme « l'Agenda 21 », composé de 350 actions donnant lieu à des projets s'inscrivant dans plusieurs axes. Quelques illustrations (non exhaustif) :

- Valorisation des circuits courts de qualité à travers le contrat CREAG33. Ce contrat est un soutien aux filières viticoles et agricoles. D'une durée de trois ans, il regroupe l'ensemble des interventions départementales et apporte aux organismes professionnels partenaires une garantie sur la viabilité du projet et sur l'engagement dans une démarche durable de qualité.
- Consommation responsable, déclinée depuis 2007 à travers plusieurs chantiers phares : les manifestations responsables, la restauration collective saine et durable, les achats publics responsables et la prise en compte de la Santé-Environnement.
- Plan Climat, Air, Energie de la Gironde 2013 – 2017 (CG33-2, 2013) (PCET33) regroupant les actions en 9 axes principaux : transport et aménagement du territoire, habitat, bâtiment et éco-construction, transition énergétique et développement des énergies renouvelables, agriculture, viticulture et forêt, milieux naturels, biodiversité et adaptation, consommation et production responsables, information, sensibilisation et formation, vulnérabilité énergétique, gouvernance.
- Préservation de la biodiversité : schéma départemental des espaces naturels sensibles, plan d'action zones humides, réseau Girondin sur la biodiversité, soutien des programmes actifs de préservation des espèces portés par des associations, nouvelles modalités de soutien au nettoyage des plages, sensibilisation aux milieux naturels et aux espèces, objectif zéro phytosanitaires – herbicides – sur les routes départementales.
- Economie sociale à travers, par exemple, la Société Coopérative d'Intérêt Collectif (SCIC) Coactions implantée à Captieux (CG33-3, 2012). Cette société a pour objet la définition d'une démarche structurante avec les acteurs de l'Economie Sociale et Solidaire du Sud Gironde afin d'inventer son futur économique et social. Cette démarche doit permettre d'initier la co-construction d'une plate-forme de partage créatrice de liens entre les acteurs sociaux du territoire et la structuration d'une dynamique collective des acteurs du monde associatif.

Depuis 2007, ce travail du CG33 dans le domaine du DD est régulièrement reconnu et récompensé par l'Etat à travers l'attribution d'un label «Agenda 21 local France». Le CG33 a également été 2 fois lauréat des « Rubans du Développement durable » et le 1er Conseil

général de France à évaluer sa « Responsabilité Sociétale » au sens défini par la certification AFNOR 26000 Collectivités (AFNOR, 2011) (anciennement appelée AFAQ 1000NR Territoires).

Pour faire écho aux actions menées autour du DD, le Conseil Départemental des Agenda 21 locaux (CDA21) (Min DD, 2006), a été créé suite à la délibération départementale de lancement d'un premier appel à projet en mars 2005. L'objectif affiché de cet appel était de soutenir et de favoriser l'émergence d'Agenda 21 locaux destinés aux Communautés de Communes, aux Pays (regroupements fonctionnels de territoires) et aux Communes de la Communauté Urbaine de Bordeaux, tous partenaires de la politique publique territoriale en matière de DD. Initialement composé des 25 collectivités bénéficiaires de cet appel à projet, le CDA21 a pour but d'échanger des bonnes pratiques, d'assurer une montée commune en connaissance et en expérience de ces membres ainsi que de rechercher une certaine cohérence des initiatives à différents échelons de l'action publique. La participation des collectivités soutenues par le CG33 relève d'un engagement, pris par la délibération de lancement de l'Agenda 21 en 2010 (CG33-4, 2010). En écho à cette délibération, le Conseil Général dépose un dossier Fonds Européen de Développement Régional (FEDER) dont les objectifs principaux sont les suivants :

1. Capitaliser, partager et construire collectivement.
2. Monter en compétence collectivement.
3. Animer, coordonner et dynamiser le réseau.

L'un des enjeux forts exposés dans ce dossier FEDER est de « configurer un réseau pérenne d'expertises et de savoir-faire en matière de développement durable à des échelles différenciées » (CG33-5, 2010).

Le CDA21 mobilise en son sein un nombre croissant de compétences. Ce sont en effet, plus de cinquante chargés de projets agenda 21, une centaine de partenaires, des techniciens thématiques… qui se retrouvent au sein d'une même entité à des fréquences régulières. Pour autant ces compétences ne sont aujourd'hui pas suffisamment mobilisées, notamment en raison d'un manque réel d'identification de chacune d'entre elles, y compris de la part des techniciens eux-mêmes. De même, de nombreux champs thématiques ou méthodologiques du développement durable ne sont aujourd'hui pas suffisamment couverts, en raison d'un manque de compétence sur des sujets en émergence. Tout l'enjeu est donc non seulement de mieux tirer parti des compétences existantes mais aussi de renforcer la montée en compétence collective, afin de renforcer la capacité du réseau à approfondir les champs du développement durable dans toute leur complexité. Dans le cadre des Agenda 21 des collectivités territoriales, de nombreux projets de développement durable sont mis en œuvre. Quelques exemples illustratifs de réalisations en 2011 (CG33-6, 2012) :

- Aménagement durable :
    a. Animation du groupe de travail (7 réunions, 20 à 35 participants) entre les différents partenaires (Collectivités volontaires, CG33, CAUE, A'Urba, DREAL, DDTM, MEDDM).

b. Définition collective d'un aménagement durable, transfert du référentiel national des Agenda 21 à un projet d'aménagement, publication du Référentiel pour l'évaluation des projets d'aménagement durable (CG33-7, 2012).

- Réseau Préca-Energie 33 dont l'objectif est de fédérer les structures qui sont impliquées dans la lutte contre la précarité énergétique en Gironde et de devenir un outil ressource. Les interventions du réseau sont diffusées à plus de 400 professionnels girondins et compte 17 structures ayant signé la charte, parmi lesquelles le Conseil Général de la Gironde et le FSL33.

  a. Actions vers les ménages : 60 visites techniques à domicile, dans le cadre d'un dispositif local d'aides aux travaux et/ou à la demande de partenaires sociaux ou des collectivités.

  b. Actions vers les professionnels : comité de pilotage qui se réunit deux fois par an / Elaboration d'une charte du réseau et premières signatures / Rencontres annuelles collégiales en décembre 2009 et 2010. Participants : Collectivités (élus et services) : 56% (dont près de la moitié des CCAS) / Structures sociales : 24% / Structures liés aux secteurs de l'énergie, eau et bâtiment/logement : 20%.

  c. Création d'outils : site internet www.preca-energie.org, fiches retours d'expérience, fiche Action, document récapitulatif sur les aides financières. Le CREAQ, au titre du réseau, est signataire du Contrat Local d'Engagement mettant en œuvre le programme national Habiter Mieux. Partenariat avec le RAPPEL, réseau national des acteurs de la pauvreté et de la précarité énergétique dans le logement

- Territorialisation du Pacte de Solidarité Ecologique en Gironde :

  a. Présentation nationale du Pacte de Solidarité Ecologique et validation des orientations communes de l'Agenda 21 du réseau girondin.

  b. Validation du GT 8 au comité de territorialisation du Grenelle.

  c. Comité de territorialisation du Grenelle de l'environnement, GT 8 « gouvernance » et plus particulièrement d'un enjeu qui identifie une nécessité de suivi sur la mise en place du Pacte de solidarité écologique au niveau régional (CG33, les autres départements de la Région, Graine Aquitaine et le CESR).

- Biodiversité - Gestion Différenciée avec accompagnement de 4 projets pilotes sur l'intégration de la biodiversité : dans un bâtiment accueillant des enfants (CDC Montesquieu), cheminement en cœur de la ville et embellissement d'un parc (Gujan Mestras), plan de gestion des trottoirs (St Médard en Jalles), et lutte contre une espèce invasive en zone forestière récréative (Pessac).

On notera, à un niveau urbain par exemple, qu'il existe des initiatives d'envergure d'aménagement telles que des chantiers de transformation complète d'un quartier en « éco-quartier » (Bardou, 2011). Le CG33 n'est pas en reste sur ce type de chantiers et finance la construction de collèges aux normes Haute Qualité Environnementale (HQE). Le Collège de Carbon-Blanc a été le premier établissement de Gironde à s'inscrire dans cette démarche. On pourra citer également le gymnase du Collège Jules Chambrelent à Hourtin et le Collège Ausone Le Bouscat.

Dans ce type de démarches, 3 axes principaux sont intégrés lors de la construction :

1. La mise en place d'un système de management d'opérations (associant les techniciens chauffage, les usagers et les services du Département).
2. Le recours à des sources d'énergie renouvelables ou locales par le fonctionnement du chauffage et la production d'eau chaude (bois, éolien, solaire, etc.).
3. Une meilleure maîtrise de la consommation d'eau potable (par la réutilisation des eaux pluviales, l'optimisation des réseaux, la mise en place de matériels préservant les ressources en eau) et une meilleure maîtrise de la consommation d'électricité.

A l'échelle du département, le CG33 prend une part active dans la définition du Schéma de Cohérence Territoriale (SCoT) qui est un document d'urbanisme permettant aux communes, communautés de communes, urbaines ou d'agglomération, de mettre en cohérence leurs politiques en matière d'habitat, de développement économique, de déplacements, d'implantations commerciales, d'environnement et d'organisation de l'espace dans une perspective de développement durable. Il y a aujourd'hui 9 SCoT en Gironde (CG33-8, 2012). Concrètement, du 26 juin au 28 juin 2013, élus, techniciens, institutionnels, universitaires, professionnels de l'immobilier et de la construction, ont planché sur la question "SCoT des villes, SCoT des champs ?". Quelles méthodologies et stratégies communes entre les territoires urbains et ruraux, mais aussi quelles spécificités territoriales revendiquer et valoriser ? Au niveau technique, on notera que 3 chantiers ont été retenus pour l'année en cours, au titre de la réflexion sur la maîtrise foncière, à savoir : l'harmonisation des indicateurs de suivi de la consommation foncière, la production d'un panorama des outils et des expériences porteurs de solutions et la réflexion sur les formes urbaines, tenant compte des spécificités territoriales.

En parallèle, dès 2010, des enseignants chercheurs de l'Ecole Nationale Supérieure de Cognitique (ENSC) ont initié le projet EcoEco « Eco-comportements pour l'Eco-conception ». L'objectif de ce projet est de modéliser le comportement humain, de créer des bâtiments et de concevoir des éco-quartiers bien adaptés aux usagers. Il s'agit de trouver des solutions adaptées à l'induction de comportements adéquats (appelés aussi "éco-comportements"), sans pour autant contraindre les usagers dans leur rapport au bâtiment ni réduire leur bien-être. Les axes de recherche envisagés portent sur la simulation énergétique au niveau du bâtiment (ADEME, 2006), l'affordance (capacité d'un système ou d'un produit à auto-suggérer son utilisation (Gibson, 1977)), la simulation d'un éco-quartier et la généralisation de la méthode proposée. Les problématiques étudiées par les deux parties (CG33 et ENSC) étant relativement proches, des contacts ont été établis et une collaboration a pu être mise en œuvre.

Problématique

En considérant tous les exemples fournis précédemment (méthodologies d'accompagnement de démarche de sensibilisation ou construction d'un bâtiment HQE), un constat issu de l'expérience s'impose : les projets de DD mis en œuvre au CG33 sont de plus en plus complexes et requièrent la collaboration de nombreux acteurs publics et privés qui se connaissent mal (CG33-9, 2011). A ce constat vient s'ajouter le fait que les démarches

territoriales de développement durable nécessitent l'articulation de compétences spécifiques et interdépendantes, qui n'existent pas chez un acteur unique.

De manière opérationnelle, en 2011, le CG33 a décidé dans son dossier FEDER (CG33-5, 2010) le développement d'un outil informatique appelé « Bourse de Compétences des projets de Développement Durable » (BCDD). L'un des objectifs de cet outil est de renseigner chacun des acteurs d'un projet sur l'expertise et le savoir-faire des partenaires potentiels. Une illustration : je cherche à organiser une soirée de sensibilisation des particuliers sur le thème des chaudières bois. En questionnant l'outil, je connaîtrai toutes les personnes susceptibles d'aider à la mise en place de ce rendez-vous (compétences en animation de réunion, en pédagogie et communication, mais aussi compétences techniques en énergie, foresterie, projet d'aménagement…). Dans le cadre du réseau girondin, l'objectif de la BCDD que l'on souhaite constituer est de :

1. Parvenir à une identification de l'ensemble des compétences réparties au sein des différentes collectivités, administration et partenaires (institutionnels ou non).
2. Répondre à la volonté de chacun des membres du réseau de mettre à profit les compétences partagées via le système informatique BCDD.

Compte tenu de la convergence des problématiques évoquées, le CG33 a confié à l'ENSC une étude R&D faisant l'objet de la présente thèse, portée par un doctorant de la Direction du Système d'Information (DSI) du CG33. La réalisation de l'outil informatique BCDD et sa mise en exploitation opérationnelle sont attendues dès la fin de l'année 2013. L'analyse montre finalement que tout le problème consiste à mettre en lien des compétences dans une perspective d'amélioration continue des relations multi-acteurs dans le cadre des projets de DD d'un territoire donné (la Gironde dans notre cas). Cet outil doit permettre de structurer, réguler et stimuler la collaboration des membres du réseau CDA21. Un enjeu majeur se dégage : partager et diffuser au sein du réseau et au-delà. Il s'agit finalement de :

1. Connaître pour partager et améliorer la connaissance des compétences existantes au sein du réseau, afin de pouvoir « se mettre à disposition les uns des autres ».
2. Démultiplier et diffuser dans le but d'impulser une dynamique territoriale multi-acteurs et citoyenne tout en renforçant la capacité du réseau à approfondir les champs du DD dans toute leur complexité.
3. Apprendre les uns des autres en repérant les compétences existantes dans tous les champs du DD, en particulier sur les sujets en émergence, afin d'assurer une montée en compétence collective.

3 objectifs majeurs sont identifiés dans la demande du Conseil Général :

1. Favoriser la transmission des savoir-faire, c'est à dire permettre la mise en lien et la pérennisation des savoirs expérientiels.
2. Obtenir une collaboration accrue des acteurs du réseau, c'est-à-dire permettre la mobilisation des ressources et initier de nouvelles synergies.
3. Structurer et doter la coordination des acteurs d'un support méthodologique, et donc modéliser un système expert à l'échelle du territoire girondin.

Cadre de réponse théorique proposé

D'un point de vue informatique, une réponse que l'on pourrait qualifier de « classique » consisterait à réaliser une base de données contenant des objets statiques (compétences, acteurs, etc.). Cependant, l'étude théorique R&D que nous décrirons tout au long de ce mémoire nous amène à définir les compétences comme étant des éléments fondamentaux et dynamiques du modèle proposé. Notre approche est novatrice car orientée « compétences » plutôt qu' « acteurs humains » comme c'est généralement le cas, et repose sur une architecture de type multi-agents (Ferber, 1999) (Jennings, 1998). L'un des points forts de cette proposition est que notre approche est générique et applicable à de multiples secteurs d'activités. La mise en application concrète du modèle (son expérimentation) prend forme dans la réalisation de l'outil BCDD demandé par le CG33. La BCDD est « vivante » et s'enrichit de l'expérience acquise et partagée par chacun des membres du réseau.

Nous insistons bien sur cette notion de partage, car il convient de mettre en place des échanges fructueux et réciproques entre ceux qui donnent et ceux qui reçoivent. Une fois une réponse reçue et mise en œuvre, on pourra considérer que la personne qui a fait la demande devient plus compétente à son tour et peut (dans un cadre d'enrichissement réciproque) partager son expérience acquise en retour. Le système vit et reste évolutif en termes d'identification des bons interlocuteurs. Il doit assurer la pertinence, l'efficacité et l'efficience des partages d'expérience en fonction de l'évaluation des résultats obtenus en retour.

Contrairement à ce qui pouvait être mis en œuvre jusqu'ici, il faut bien concevoir ce système de Bourse de Compétence comme un outil transversal aux institutions publiques impliquées dans les projets de DD. Il met en œuvre une certaine interdépendance des personnes impliquées au service d'un objectif commun à atteindre. La vision d'une transversalité des compétences (et non plus une vision « en silos ») pour atteindre les objectifs fixés revêt ici tout son sens.

Dégageons une liste de mots clé pour décrire globalement le besoin: acteurs, objectifs, compétences, collaboration, dynamisme, apprentissage, émergence, système expert et relation. Deux conclusions ressortent de cette analyse. La première est qu'une implémentation « classique » et/ou « statique » basée simplement sur l'interrogation d'une base de données ne répond pas aux attentes, en particulier pour ce qui concerne les aspects de dynamisme, d'apprentissage et d'émergence. Nous étayerons cette dernière affirmation plus loin, au paragraphe 1.3.1. La seconde est que les compétences sont au cœur du système à construire et que tout est orienté autour d'elles : partage, collaboration, accroissement de niveau de compétence collectif. Finalement, pour répondre aux exigences formulées par le CG33, nous avons choisi de faire du concept de compétence un agent au sein d'un SMA.

Structure de ce mémoire de thèse

Ce mémoire de thèse possède 3 parties principales. Dans la première, nous abordons les principaux concepts utilisés dans nos travaux (compétences, objectifs, projets, etc.) et présentons un état de l'art autour des SMA dans le domaine de la gestion des compétences. Nous proposons également dans cette partie un focus particulier sur les mécanismes automatiques d'apprentissage existants. Dans le second chapitre, nous décrivons notre approche et la modélisation des principaux éléments clé de notre système. On y aborde, entre autre, la définition des « *Agents Compétence* », les différents fondements théoriques à partir desquels le comportement global de notre système a été déterminé, ainsi que la définition des mécanismes de perception et d'échange utilisés. De même, nous y décrivons le cycle de vie et le modèle comportemental de ces agents, ainsi que les mécanismes d'apprentissage mis en œuvre. Dans une troisième partie, nous présentons l'implémentation concrète que nous avons réalisée en fournissant les résultats que nous avons obtenus dans le cadre d'un exercice d'application accessible au paragraphe 3.14. Cet exercice applicatif décrit un système d'aide à la décision pour la sélection de joueurs de rugby. Finalement, nous livrons nos conclusions et perspectives dans le but de présenter les champs de recherche ouverts par les travaux réalisés dans le cadre de cette thèse.

CHAPITRE 1 - Etude de l'existant

« Quiconque prétend s'ériger en juge de la vérité et du savoir s'expose à périr sous les éclats de rire des dieux puisque nous ignorons comment sont réellement les choses et que nous n'en connaissons que la représentation que nous en faisons. »

Albert Einstein

1.1 Introduction

Cette première partie consacrée à l'étude de l'existant vise tout d'abord à exposer notre problématique à travers une synthèse des spécifications de besoin fournies par le CG33. L'analyse de ces spécifications nous conduit alors naturellement à présenter les définitions des différents concepts clés que nous y rencontrons. Ces définitions sont tout d'abord présentées telles que disponibles dans la littérature à ce jour, puis telles que nous les entendons dans nos travaux. Puisque notre réponse se décline sous la forme d'un outil (BCDD) s'appuyant sur les systèmes multi-agents (Ferber, 1999) (Jennings, 1998), nous présentons également un paragraphe consacré à ce sujet. Cependant, la littérature sur le sujet étant abondante et largement disponible sur Internet, nous nous focalisons sur les travaux scientifiques menés autour de la gestion des compétences dans les SMA, en droite ligne avec la problématique qui nous préoccupe. La notion d'apprentissage est fondamentale dans notre contexte. Nous souhaitons effectivement construire un système dynamique, évolutif, et apprenant. Il devient donc logique dans ce chapitre de fournir un panorama des mécanismes automatiques d'apprentissage existants.

1.2 Problématique

1.2.1 Le cas CG33 : spécifications du besoin

Au CG33, de nombreux projets sont engagés autour des thématiques du Développement Durable (sensibilisation autour des nouveaux moyens de production d'énergie, construction de bâtiments HQE, transformation d'un quartier en « éco-quartier » (Bardou, 2011), etc.). L'observation sur le terrain montre que ces projets nécessitent l'implication et la coopération de nombreux acteurs, chacun d'eux n'ayant qu'une vision partielle des capacités globales des autres (CG33-9). Les démarches territoriales de DD mettent en œuvre de compétences

spécifiques, souvent interdépendantes, mais que l'on trouve rarement chez un seul acteur. Tout l'enjeu, pour le CG33, consiste alors à mettre en place un système d'information accessible en ligne permettant la recherche, le partage, la sélection de compétences et d'acteurs, dans le but d'assurer la réussite des projets de DD en Gironde. Il s'agit également de répondre à des objectifs tels que :

- Renseigner chacun des acteurs sur l'expertise et le savoir-faire des partenaires potentiels.
- Assurer une montée en compétence des acteurs au travers des échanges (réciprocité).
- Mobiliser autour de la mise en œuvre des politiques de DD.

Le système est interrogé par un acteur souhaitant bénéficier de compétences mises en œuvre dans la gestion de projets de développement durable, et partagées entre les 61 membres du réseau des agendas 21 de la Gironde. Certaines de ces compétences sont à ce jour définies comme étant des savoirs *explicites* (i.e. documentés). A l'opposé, d'autres sont de type *implicites* (cf. paragraphe consacré aux définitions plus bas dans ce chapitre). Les membres de ce réseau partagent des compétences dans des domaines variés qu'ils souhaitent (peuvent) mettre à disposition de tous pour aider dans la mise en œuvre de leurs objectifs de DD. La bourse est « vivante » et s'enrichit de l'expérience acquise et partagée par chacun des membres du réseau. Nous insistons sur cette notion de partage. Il convient en effet de mettre en place des échanges fructueux et réciproques entre ceux qui donnent et ceux qui reçoivent. Ainsi, nul ne saurait demander plusieurs fois sans avoir, à son tour, fait bénéficier les autres membres de son expérience. Une fois une réponse reçue et mise en œuvre, nous pouvons considérer que la personne qui a fait la demande devient plus compétente à son tour et peut, dans un cadre d'enrichissement réciproque, partager son expérience acquise en retour.

Le système ne doit pas être statique. L'identification des bons interlocuteurs doit évoluer en fonction de la pertinence, de l'efficacité et de l'efficience des partages d'expérience ainsi que des résultats concrets obtenus dans le déroulement du projet. Les règles d'évolution mentionnées ci-dessus s'accompagnent de la mise en place de seuils d'alerte en cas de dysfonctionnements du système ou de problèmes constatés. Quelques illustrations :

- Participation non effective d'une personne qui était identifiée.
- Seuil d'alerte de non effectivité d'une compétence.
- Mauvais retour d'évaluation suite à mise en œuvre d'une compétence.

1.2.2 Généralisation

A partir du cas CG33, nous pouvons généraliser et dire que quel que soit le secteur d'activité professionnel concerné, l'identification et le partage optimal de compétences est au cœur de la réussite des *projets* qui concrétisent des *objectif* dans le « monde réel ». Considérons, par exemple, un projet « *construction d'un bâtiment HQE à Lormont* » qui serait une instance de l'objectif général « *Construire un bâtiment HQE* » (ADEME, 2006). Dans un tel projet, via la solution informatisée que nous apporterons, tout maître d'œuvre (MOe) doit être capable :

1. De définir son projet et d'identifier les compétences qui le composent.

2. D'identifier les acteurs ou compagnies privées capables d'exercer les compétences conduisant à la réussite du projet, c'est à dire le bâtiment HQE de Lormont construit dans notre illustration.
3. D'évaluer, à postériori, la réussite du projet.

Concernant le 1er point, deux solutions sont possibles :

A. Il existe dans le système un projet qui reflète, exactement ou partiellement, ce que le MOe souhaite faire. Il suffit par conséquent de trouver un moyen de récupérer la liste des compétences utilisées dans cet ancien projet, puis de procéder à la création du nouveau en utilisant tout ou partie des compétences prises dans cette liste.
B. Le cas est plus complexe. Il n'existe pas dans le système un tel projet unique. Par exemple, les compétences pour construire le nouveau projet pourraient être réparties dans plusieurs listes, chacune appartenant à différents projets du système. Tout l'enjeu consiste alors à trouver un moyen de répondre finalement à une spécification partielle et limitée des besoins, en termes de liste de compétences, pour la constitution du nouveau projet.

Concernant le 2ème point, pour une compétence donnée, nous pourrions penser intuitivement à présenter simplement une liste d'acteurs triée selon le niveau d'expérience. Cependant, d'autres critères de sélection que le seul niveau d'expérience doivent (peuvent) être pris en compte par le maître d'œuvre, comme par exemple : le prix, la durée, l'expertise dans la pose de tel ou tel matériau spécifique, etc. Ainsi, le système doit pouvoir suggérer une liste d'acteurs en fonction de différents critères, et veiller également à ne pas toujours sélectionner les mêmes acteurs. Dans le cas contraire, le souhait de faire monter chacun des acteurs en compétence, à travers les projets, ne pourrait être atteint. Tout l'enjeu consiste finalement à définir la bonne stratégie de sélection des acteurs, la mieux adaptée au secteur d'activité professionnel pour le compte duquel le système est utilisé.

Concernant le 3ème point, toute l'importance réside dans l'obtention d'un retour d'expérience « chiffré » sur un projet donné. L'objectif est de valoriser la qualité de mise en œuvre d'une compétence, grâce à un acteur, pour ledit projet. Ces valeurs permettront de calculer un indicateur qui sera l'un des critères pris en compte pour les prochaines itérations du 2nd point. Il permettra également de répondre au souhait de suivi de montée en compétence des acteurs ainsi que d'identifier, par exemple, les compétences sur lesquelles un effort de formation doit être fait. Plus généralement, cette préoccupation de suivi via un retour d'expérience traduit le désir de faire émerger une certaine « Intelligence Collective » (Dortier, 2006), basée sur une valorisation forte des aspects de collaboration, de partage et de diffusion (Boutte, 2009).

1.3 Analyse & proposition de réponse

La réalité des projets, quel que soit le secteur d'activité professionnel, nous pousse à intégrer un certain dynamisme au sein de notre solution. En effet, les compétences utilisées au sein des projets évoluent, peuvent se transformer par les évolutions techniques, et peuvent disparaitre (si remplacées ou finalement non utilisées). Elles doivent répondre aux préoccupations émises par les utilisateurs au moment où ils expriment leur besoin.

Prenons un exemple : je suis un Girondin ayant pour but de réaliser des économies d'énergie en utilisant des technologies durables au sein de mon habitation. A ce stade de ma réflexion, mes questions sont les suivantes :

1. Que dois-je faire comme travaux (isolation, etc.) ?
2. Qui peut m'aider à les réaliser (quelles entreprises) ?

Une réponse dite « classique » pourrait consister à réaliser simplement une base de données où les compétences et acteurs seraient des objets statiques. Cependant, ceci ne répond pas à la nécessité de dynamisme et d'évolution des compétences. L'analyse de la problématique nous conduit à définir la *Compétence* comme l'élément fondamental du système à construire. Compte tenu de ces éléments, l'étude R&D que nous avons mené dans le cadre de cette thèse nous a conduit à proposer plutôt des « agents compétence » évoluant au sein d'une architecture de type multi-agents (Ferber, 1999) (Russel, 2003). Si l'on raisonnait de manière classique, (de type logiciel de gestion de projets par exemple), le système cible serait conçu de manière « centrée sur les acteurs ». Dans ce cas, chacun d'eux possèderait une liste de compétences. A l'opposé, nous proposons une approche où la conception est centrée compétence. On considère ainsi que chacune d'elles est mise en œuvre par une liste d'acteurs. Cette approche où un agent logiciel modélise une compétence est novatrice puisque :

1. Elle oriente réellement la mise en œuvre des projets vers la nécessité d'obtenir l'exercice d'une « compétence » et non plus vers la nécessité de « disposer » de tel ou tel acteur humain pour exercer la compétence. De manière illustrée, nous demanderons par exemple simplement au système une compétence en plomberie. Nous n'appellerons plus directement M. X parce que nous savons que c'est un bon plombier. Les modalités de raisonnement deviennent différentes, nous nous recentrons sur le besoin et non plus sur sa réalisation. Il convient de penser « compétence » et non plus « acteur humain qui la détient ».
2. Elle confère aux compétences toutes les caractéristiques induites par une modélisation sous la forme d'agents logiciels. On pourra citer, entre autres : capacités de communication inter-agents, perception de l'environnement dans le SMA, mise en œuvre de comportements, apprentissage.
3. Elle propose, d'un point de vue théorique, une orientation intéressante. Le concept même de compétence, par sa modélisation sous la forme d'une entité logicielle, devient une entité « *vivante et autonome* » au sein d'un SMA. Une compétence possède un cycle de vie qui lui est propre, ainsi qu'un modèle comportemental personnalisé, évolutif, pouvant être appris. Nous pouvons alors parler d' « ***Agent Compétence*** » (AC).
4. Elle est cohérente avec les concepts de « *l'entreprise apprenante* » (Mallet, 2007) et du « *Knowledge Management* » (Prax, 2000), et par la même généralisable quel que soit le secteur d'activité professionnel. L'organisation, l'utilisation, et la transmission (acquisition) des compétences par les différents acteurs humains concourent à l'atteinte des objectifs d'émergence de l'« *Intelligence Collective* » (Dortier, 2006). Boutte nous explique que: « *Les contributions des différents auteurs éclairent sous plusieurs angles notre problématique (… dont) celui de la* ***modélisation de la compétence*** *à l'ère de la complexité.* » (Boutte, 2009). La définition de notre concept d'AC s'inscrit alors pleinement dans cette citation.

5. Elle prend en compte la granularité des compétences à travers différents niveaux hiérarchiques[1].

Reprenons notre illustration basée sur le cas CG33. La réponse souhaitée par cette institution territoriale prend la forme de l'outil informatique BCDD dans lequel est exploitée une “Bourse de Compétences”. Une représentation schématique et simplifiée des interactions entre acteurs humain et ce type de systèmes est donnée figure 1.

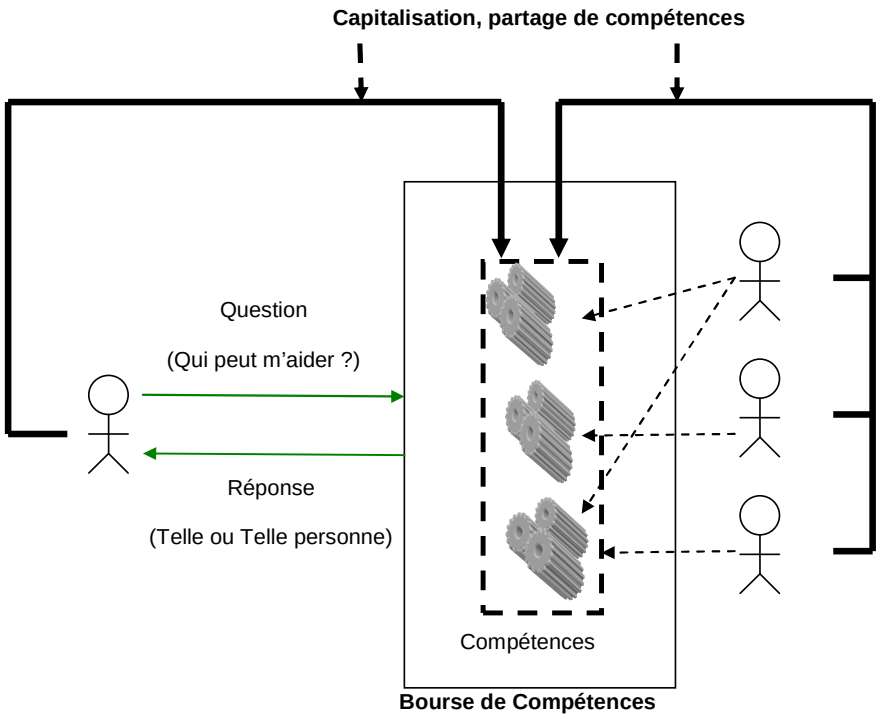


Figure 1 Les principes de la Bourse aux Compétences

Un tel système sera interrogé par un acteur humain qui souhaite bénéficier de compétences partagées au sein de la bourse. Les acteurs humains partagent bien des compétences pour aider à la mise en œuvre des projets. La bourse s’enrichit de l’expérience acquise et partagée par chacun d’eux. Le système est dynamique. Pour un projet donné, l’identification des acteurs exerçant une compétence évolue en fonction du contexte. Synthétiquement, ce n’est pas toujours le « meilleur acteur » (au sens niveau d’expérience) qui sera désigné par le système. Il est important de noter également que nous proposons de concevoir ce système de Bourse de Compétence comme un outil transversal. Il met en œuvre une certaine interdépendance des personnes impliquées au service d’un objectif commun à atteindre. La vision d’une transversalité des compétences (et non plus une vision « en silos ») pour atteindre les objectifs fixés, au travers des projets, revêt ici tout son sens. Une compétence peut être portée par 0 (si c’est une compétence que l’on souhaite acquérir ultérieurement après formation professionnelle par exemple) ou n individus physiques

[1] Cf. paragraphe « *1.4.6 Compétence élémentaire* » dans les définitions fournies plus bas

1.4 Définitions, concepts & notions clé

1.4.1 Connaissance

Il existe dans la littérature des ouvrages de référence sur le sujet dont «*Théorie générale de la connaissance* » (Schlick, 2009). D'un point de vue purement scientifique, on identifie 2 méthodes principales d'étude des connaissances : l'épistémologie et l'histoire des sciences. L'acquisition de connaissances est une des caractéristiques fortes de l'espèce humaine. L'un des points fondamentaux à retenir est qu'une connaissance est transmissible à d'autres humains via différents moyens de communication tels que l'écrit ou l'oral. Par exemple, l'école, avec ses enseignements, utilise ces deux moyens. Une connaissance appartient à un « *domaine de connaissance* », comme par exemple celui des connaissances usuelles (savoir compter), des connaissances pratiques (savoir se servir d'un marteau), etc. On notera que la langue anglaise, quant à elle, utilise le terme de « *knowledge* » pour désigner la connaissance.

Il est intéressant de noter que l'on peut répartir les connaissances selon 2 catégories :

1. ***Explicites*** : elles correspondent, au sein d'une organisation (entreprise, laboratoire, etc.), à ce qui est formalisé. Pour illustrer, on peut citer les normes ISO 9xxx, les modes opératoires d'un processus dans une entreprise, des spécifications de logiciel, etc. De manière générale, elles recouvrent l'ensemble des productions intellectuelles permettant de conserver ce que l'on pourrait appeler le "savoir de l'organisation". Elles sont transmissibles (communicables oralement ou via un support écrit), peuvent être facilement rendues visibles et partagées, que ce soit à un niveau global ou non.
2. ***Tacites (ou implicites)*** : A l' opposé des connaissances explicites, elles correspondent à ce qui n'est connu que d'un petit nombre de personnes (voire d'une seule) et sont difficiles à formaliser. Elles s'inscrivent profondément dans la pratique, la routine, l'engagement professionnel, les émotions. Une telle connaissance est difficilement communicable en l'état car non formalisée. Elle n'est souvent transmissible à une autre personne que par le partage d'expérience avec celui qui la porte.

Pour ne prendre que cet exemple, dans leur publication « *SECI, Ba and Leadership: a Unified Model of Dynamic Knowledge Creation* » (Nonaka, 2000), les auteurs proposent un modèle de création (au sens acquisition & transmission) de la connaissance. Ce modèle repose sur 3 éléments clé que sont :

1. Le processus SECI de création de connaissance. Il s'effectue à partir des interactions entre connaissances explicites et tacites et se compose de 4 éléments :
    a. *Socialisation* : Transmission de connaissances tacites impliquant de passer du temps avec le porteur de la connaissance ou de vivre dans le même environnement.
    b. *Externalisation* : Conversion d'une connaissance tacite en implicite, ce qui revient à la « figer », au moins momentanément, pour la formaliser et ainsi la maitriser.
    c. *Combinaison* : « éclatement » d'une connaissance explicite en plusieurs avec diffusion des nouvelles connaissances crées.
    d. *Internalisation* : Conversion (appropriation), sous une forme tacite et par un acteur, d'une connaissance explicite. Ceci revient à « apprendre par la pratique »

avec création d'un modèle mental suite à mise en pratique d'une connaissance explicite. On trouvera la notion d'*embodiement* de Varela (Varela, 1991) à ce niveau. Il peut y avoir ensuite socialisation ou externalisation de cette connaissance par l'acteur qui la possède.

2. Le « *ba* », qui représente le contexte partagé dans lequel évolue le processus de création de connaissances. Il est spécifique à chaque organisation et peut se diviser en 4 catégories :
    a. *Originating ba* : Contexte d'interaction entre acteurs humains, 2 à 2, en physique. Utilisé essentiellement dans le cadre de la socialisation.
    b. *Dialoguing ba* : Contexte d'interaction collective entre acteurs humains, en physique. Utilisé pour l'externalisation.
    c. *Systemising ba* : Contexte d'interaction collective, mais virtuel. Il se prête bien, par exemple, à la diffusion de connaissances explicites formalisées (ex : communication écrite).
    d. *Exercising ba* : Contexte d'interaction individuelle, mais de manière virtuelle. Utilisé souvent pour l'internalisation de connaissances à partir de supports virtuels.

3. Les « *knowledge assets* » qui, en synthèse, regroupent les entrants, sorties, et facteurs de modération du processus global de création de connaissance.

Citons également et brièvement les travaux de Ermine dans son article « *Knowledge Crash and Knowledge Management* » (Ermines, 2010) ou bien dans « *A Shannon's Theory of Knowledge* » (Le Blanc, 2007) coécrit avec Le Blanc. Les auteurs nous proposent le modèle AIK, basé d'ailleurs sur la publication (Nonaka, 2000) que nous venons de voir juste ci-dessus. Dans AIK :

1. **A** est mis pour **A**cteurs de la connaissance, regroupés en communautés de savoir, qui partagent et créent la connaissance.
2. **I** est mis pour un système d'**I**nformation qui stocke, traite et met à disposition les informations de l'organisation.
3. **K** est mis pour un patrimoine de connaissances (**K**nowledge) où s'accumulent les savoirs créés et utilisés dans l'organisation.

Dans ce modèle (voir figure 2), la transmission d'information et de connaissances s'effectue entre les acteurs (**A**) via l'utilisation ou non de l'outil informatique (**I**), et en s'appuyant sur la base de connaissances (**K**) de l'organisation.

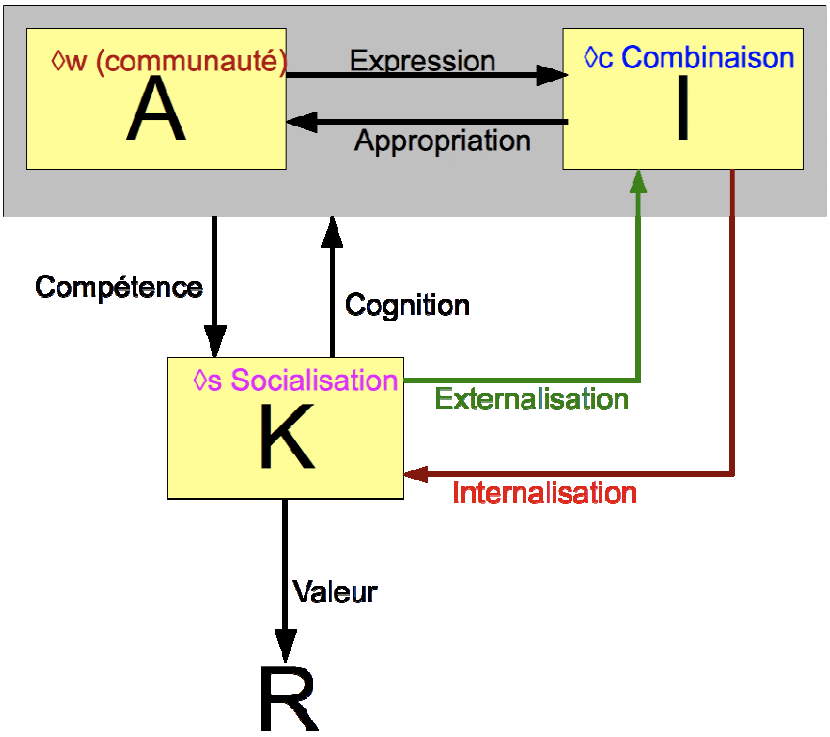


Figure 2 Le modèle AIK (Ermine, 2010)

1.4.2 Savoir

Si une connaissance, prise de manière unitaire, s'insère dans un domaine de connaissance particulier, le savoir peut être vu comme étant l'« *ensemble des connaissances d'une personne ou d'une collectivité acquises par l'étude, par l'observation, par l'apprentissage et/ou par l'expérience* » (TLFI, 2013). Cependant, le savoir ne se limite pas qu'à un simple ensemble de connaissances. Il s'accompagne de processus cognitifs d'assimilation et d'organisation des connaissances. Ces processus varient en fonction des expériences vécues, ainsi que des informations environnementales représentatives du contexte (situation, lieu, etc.) dans lequel les connaissances sont acquises. On notera ici que pour désigner la notion de savoir, la langue anglaise utilise le même terme que celui qui désigne la connaissance : « *knowledge* ».

1.4.3 Expérience

L'expérience permet l'acquisition de connaissance et la mise en œuvre d'un savoir. L'Académie Française, en 1986 (Académie Française, 1986), définit l'expérience comme étant « *Le fait d'acquérir, d'étendre ou d'enrichir une connaissance, un savoir, un savoir-faire, par l'usage et la pratique* ». Une expérience peut être répétée i fois, i variant de 1 à n. Dans le langage courant, un nombre élevé d'expériences est souvent synonyme « *d'avoir de l'expérience* ».

1.4.4 Savoir-faire

Si un savoir, pris d'un point de vue théorique, peut être concrétisé dans « la vie réelle », on parlera de « *savoir-faire* ». Par exemple, savoir démonter une roue sur une voiture se traduit par l'acte (réalisation concrète d'une expérience) de démontage de celle-ci dans la vie réelle. Savoir démonter une roue est un savoir-faire puisqu'expérimenté concrètement. En droit, l'une des définitions du savoir-faire est la suivante : « *les connaissances techniques, transmissibles, non immédiatement accessibles au public et non brevetées, et pour lesquelles quelqu'un serait disposé à payer pour en avoir connaissance* » (Mousseron, 1977).

En anglais, le terme « *know-how* » désigne un savoir-faire. Le dictionnaire Larousse en ligne fournit d'ailleurs cette définition pour ce terme anglais : « e*nsemble des notions, des connaissances et de l'expérience concernant des opérations ou des procédés utiles à la réalisation d'un produit ou d'un service* ». Dans notre exemple, le service sera celui de démontage de la roue. A titre d'information, on notera qu'un « *savoir-être* » peut être analysé comme étant un « *savoir-faire relationnel* » s'il se traduit par un ensemble d'attitudes et de comportements mis en place pour s'adapter à une situation donnée.

1.4.5 Compétence

Il n'existe pas de définition unique de ce terme. Selon Ledru et Michel (Ledru, 1991), la compétence est la capacité dont on dispose pour résoudre des problèmes de manière efficace dans un contexte donné. Conceptuellement, une compétence ne se limite pas uniquement aux notions de savoir et savoir-faire (expérientiel par exemple pour Le Boterf (Le Boterf, 1994) (Le Boterf, 2006) (Le Boterf, 2008). Pour Calut (Callut, 2012), elle implique aussi la capacité à répondre à des exigences complexes et à pouvoir mobiliser et exploiter des ressources psychosociales (dont des savoir-faire et des attitudes) dans un contexte particulier. Gillet (Gillet, 1991), quand à lui, nous propose qu'une compétence soit vue comme « *un système de connaissances, conceptuelles et procédurales, organisées en schémas opératoires et qui permettent, à l'intérieur d'une famille de situations, l'identification d'une tâche-problème et sa résolution par une action efficace* ».

La définition du concept de compétence influe directement sur l'évolution du management interne des entreprises. De nombreux travaux, décrits au sein d'une littérature très riche, sont disponibles. On pourra citer par exemple le modèle de l'Entreprise Apprenante. Selon Mallet (Mallet, 2007), l'entreprise est assimilée à une entité organique, vivante. Elle devient capable d'apprendre, de mémoriser, de créer, de réagir à différents évènements survenant dans son environnement. De même, l'apparition du concept de « *Knowledge Management* », selon Prax (Parx, 2000) considère que l'organisation de l'échange d'informations et le partage des connaissances sont devenus des facteurs clés d'une gestion performante de l'entreprise. Pour Grundstein (Grundstein, 2006), ils doivent s'inscrire dans un projet global, destiné à mettre en valeur les savoirs et les savoir-faire individuels et collectifs.

Considérons de manière plus approfondie la définition proposée par Zarifian (Zarifian, 2002) où la compétence peut être caractérisée par 3 « *caractères* » :

1. *Une attitude sociale* exprimée par une prise d'initiative et de responsabilité par rapport aux situations dont l'individu ou le groupe a la charge et auxquelles il se confronte, en visant la réussite de leur action.
2. *Une approche cognitive* où la compétence est alors définie comme une « *intelligence pratique* » de ces situations, orientée vers l'action et qui mobilise de l'expérience et des connaissances acquises.
3. *Une existence collective*. Citons directement l'auteur dans son article pour bien retranscrire sa pensée : « *dès que la situation possède une certaine surface, une certaine dimension et complexité, et dès qu'elle suppose un chaînage d'actions, apparaît, non pas une compétence collective, mais une composition de compétences au sein d'une communauté d'action. Chaque individualité possède des compétences et une amplitude d'action possible qui sont toujours limitées. La plupart des situations font appel à un ensemble de compétences qui sont amenées à se composer entre elles, selon un principe proche de la composition des forces en sciences physiques* ».

Si nous rappelons les objectifs poursuivis pour élaborer notre outil informatique BCDD, nous pouvons les classer selon les 3 caractères de Zarifian (Zrifian, 2002) :

1. Connaître pour partager, pour améliorer la connaissance des compétences existantes au sein du réseau, afin de pouvoir se mettre à disposition les uns des autres => *Attitude sociale*.
2. Démultiplier et diffuser dans le but d'impulser une dynamique territoriale multi-acteurs et citoyenne tout en renforçant la capacité du réseau à approfondir les champs du DD dans toute leur complexité => *Approche cognitive*.
3. Apprendre les uns des autres en repérant les compétences existantes dans tous les champs du DD, en particulier sur les sujets en émergence, afin d'assurer une montée en compétence collective => *Existence collective*.

Il est intéressant ici de constater que nos hypothèses de départ font sens compte tenu d'éléments pris dans la définition de Zarifian (Zarifian, 2002). Aucun acteur unique ne dispose de la globalité des compétences : « *Chaque individualité possède des compétences et une amplitude d'action possible qui sont toujours limitées* ». L'objet de cette thèse est de proposer une solution basée sur les SMA, avec introduction du concept d'agent compétence, dynamique et autonome : « *La plupart des situations font appel à un ensemble de compétences qui sont amenées* ***à se composer entre elles****, selon un principe proche de la composition des forces en sciences physiques* ». On notera que plus loin dans ce même article, Zarifian précise que la composition des compétences entre elles nécessite (requiert) que les acteurs prennent conscience (volet social et affectif) que la somme de leurs actions sera supérieure à celle de leurs actions individuelles. Cette dernière affirmation vérifie l'un des principes de base de la théorie Gestalt présentée par Dortier (Dortier, 2008) : « *un tout est supérieur à la somme des parties qui le compose* ». Dans notre proposition, une compétence C_i, avec $C_i \in \mathcal{C}$ (l'ensemble des Compétences) est immatérielle et atemporelle. Elle est dite

« implémentée » lorsqu'elle est mise en œuvre par un ou plusieurs acteurs humains Aj, avec $A_j \in \mathcal{A}$ (l'ensemble des acteurs), à un instant t.

On notera, qu'en français, le mot compétence est unique. En langue anglaise, il faut distinguer 2 notions : « *skill* » et « *competence* » (Mulder, 2007):

1. *Skill*: capacité à exploiter des connaissances et des savoir-faire dans le but de résoudre une classe de problèmes donnée.
2. *Competence*: ensemble de comportements ou d'actions qu'il convient de mettre en œuvre dans un contexte particulier.

1.4.6 Compétence élémentaire

Selon Perrenoud repris par Roux (Roux, 2011), « *une compétence peut fonctionner elle-même comme une ressource, mobilisable par des compétences plus larges* ». A l'opposé, il est intéressant de constater que certains – pédagogues du DISCAS (DISCAS, 2006) qui fut un bureau privé de consultation pédagogique québécois qui œuvra dans le monde de l'éducation de 1987 à 2006 – considèrent qu'une compétence est « complète et insécable ». Pour ces derniers, « *une compétence doit permettre de gérer complètement une situation, sinon ce n'est pas une compétence, mais une ressource ou une simple dimension de la compétence* ». Toute la difficulté consiste donc à trouver le bon niveau de granularité.

On sait que dans certains secteurs d'activité tels que l'industrie, le niveau de granularité peut être considéré comme connu. Cela peut être le cas, par exemple, pour la décomposition en tâches (processus de fabrication bien définis et donc connus à l'avance) pour réaliser la fabrication d'une pièce. Cependant, il n'existe pas de méthode de décomposition, didactique, que l'on pourrait qualifier de « standard » et qui soit généralisable à tous les domaines d'activité professionnels. A l'instant t, tous les utilisateurs interrogeant le système pour la réalisation d'un projet P sont plus ou moins **globalement** capables d'identifier les compétences C_i de « haut niveau » requises pour le concrétiser. A l'opposé, les utilisateurs ne sont pas forcément capables de définir **précisément** toutes les compétences élémentaires C_e qui composent les compétences C_i de « haut niveau » dont ils ont besoin pour réaliser P.

1.4.7 Domaine de compétence

Il existe de multiples définitions du concept de « domaine de compétence ». Rychen et Salganik (Rychen, 2011) mènent une analyse concernant la comparaison des processus d'éducation et d'enseignement entre les différents pays de l'OCDE. Ils identifient 4 domaines de compétence distincts (page 182) : organisation de l'enseignement, gestion du personnel, planification, allocation de ressources et suivi de leur utilisation. Chacun de ces domaines rassemble 15 décisions (relevant de compétences des gouvernements) prises par chaque Etat.

En référence à cette définition, nous pouvons convenir dans nos travaux qu'un domaine de compétence soit considéré comme un regroupement fonctionnel (ayant un sens au niveau

secteur d'activité professionnel), d'une ou plusieurs compétences. Réciproquement, une compétence peut appartenir à 1 ou n domaines. A titre illustratif, considérons 3 exemples.

Exemple 1. Le CG33 définit des politiques publiques et les concrétise sous la forme de projets. Ceux-ci font appel à différentes compétences, chacune d'elles appartenant à différents domaines d'application sur le territoire. Par exemple : vie au collège, environnement, DD, sports, lutte contre les discriminations, citoyenneté européenne, art et culture, solidarité et droits de l'enfant, prévention et santé. Ainsi, nous pouvons dire que :

1. Les compétences de gestion de l'Allocation Personnalisé d'Autonomie (APA) et de paiement de la Prestation Compensatoire du Handicap (PCH) appartiennent au domaine « solidarité et droits de l'enfant ».
2. La compétence de soutien aux programmes de préservation des espèces et de la biodiversité appartiendra au domaine « environnement ».

Exemple 2. Secteur d'activité du bâtiment.
Nous pouvons proposer que :

1. Les compétences de maître d'ouvrage et de maître d'œuvre appartiennent au domaine « pilotage ».
2. Les compétences d'architecte et d'expertise en études de sol appartiennent au domaine des « bureaux d'étude ».
3. Les compétences de constructeur, paysagiste, spécialiste en pose de panneaux photovoltaïque appartiennent au domaine des « artisans et entreprises ».

Exemple 3. Secteur d'activité sportif, jeu de rugby.
Si l'on assimile les postes des joueurs à des compétences, nous pouvons dire de manière communément admise que :

1. Les compétences d'ouverture, arrière, centres et ailiers appartiennent au domaine des « trois-quarts ».
2. Les compétences de piliers, talonneur, secondes lignes, troisièmes lignes et demi de mêlée appartiennent au domaine des « avants ».
3. Les compétences de demi de mêlée et d'ouverture appartiennent également à un domaine appelé « charnière ».

1.4.8 Relations entre compétences

De ce que nous venons de voir aux paragraphes précédents, nous pouvons dire que chaque compétence, prise de manière unitaire :

1. Est décomposable en compétences élémentaire (cf. paragraphe 1.4.6).
2. Prend part à la concrétisation de projets en lien avec d'autres compétences (cf. définition de Zarifian (Zarifian, 2002) au paragraphe 1.4.5).

A partir de ce constat, nous pouvons dire que des compétences peuvent être « *liées* » entre elles par différents éléments communs. C'est le cas si elles possèdent certaines compétences

élémentaires communes. De même, elles peuvent appartenir à des domaines de compétence similaires ou bien encore être employées ensemble dans des contextes d'utilisation traduits sous la forme de projets. Nous considérons alors que ces liens traduisent ce que nous nommons une « *relation* » entre les compétences. Bien entendu, les relations établies entre les compétences seront plus ou moins « fortes » (i.e. pertinentes) selon le nombre et la nature (projet, domaines) des caractéristiques qu'elles ont en commun. Reprenons, un exemple pris parmi ceux que nous avons proposé au paragraphe précédent :

1. Les compétences d'architecte et d'expertise en études de sol appartiennent au domaine de compétence des « bureaux d'étude ».
2. Les compétences de maître d'ouvrage et de maître d'œuvre appartiennent au domaine de compétence « pilotage ».

Toutes ces compétences sont nécessaires lors de la réalisation d'un projet de construction de bâtiment. Elles sont donc liées, à minima, par de tels projets. Cependant, la relation que possède la compétence d'architecte avec celle d'expertise en étude des sols peut être considérée comme plus forte (partage du même domaine de compétence) que celle qu'elle a avec la compétence de maître d'ouvrage.

1.5 Systèmes multi-agents

1.5.1 Intérêt des SMA pour notre problématique

Dans le chapitre d'introduction de ce document, au paragraphe « *Cadre de réponse théorique proposé* », nous suggérons de résoudre notre problématique globale via la création d'un système que l'on peut caractériser par les mots clés : acteurs, objectifs, compétences, collaboration, dynamisme, apprentissage, émergence, et relations. Le choix d'une implémentation « classique » de ce système reposerait simplement sur l'interrogation d'une base de données. Cependant, compte tenu du caractère « statique » des objets (tables et enregistrements) manipulés dans ce type d'approche, nous ne répondrions pas aux attentes de dynamisme, d'apprentissage, et d'émergence.

Reprenons (cf. chapitre d'introduction) l'exemple de notre projet « *organisation d'une soirée de sensibilisation des particuliers sur le thème des chaudières bois* ». Nous souhaitons que l'outil BCDD nous propose de connaître toutes les personnes susceptibles d'aider à la concrétisation de cette manifestation à travers la mise en œuvre de différentes compétences. Dans notre exemple, nous avons besoin (entre autre) de compétences en animation de réunions, communication, mais également techniques sur les chaudières à bois.

Un système qui serait adossé à une base de données nécessite des actions humaines, volontaires et manuelles, de mise à niveau de son paramétrage et de son contenu (liste des compétences disponibles, liste des acteurs capables de les concrétiser, liste des compétences utilisées dans un projet, etc.). Compte tenu de cet impératif d'une intervention humaine, on ne peut pas dire qu'un tel système soit « intelligent », possède une « vie propre » ou face preuve d'une quelconque autonomie. L'intelligence reste alors bien déportée au niveau humain et non développée par le système informatique lui-même. Cette approche ne permet pas non plus de favoriser l' « émergence » de certaines évolutions dans la structuration même des projets. En effet, la mise à jour statique des données par les utilisateurs humains ne favorise pas, par exemple, la suggestion d'établissement de liens inter-compétences auxquels on n'aurait pas pensé de prime abord. La dynamique du système ne serait pas alors pleinement assurée en fonction du contexte spécifique du terrain.

Rappelons également que notre objectif est de faire en sorte que les compétences, et non les acteurs humains, soient au cœur du système. L'approche novatrice proposée par cette thèse consiste alors à démontrer qu'il est possible de modéliser le concept même de compétence sous la forme d'entités logicielles dynamiques, apprenantes et autonomes : des « agents compétence », pour lesquels les acteurs humains ne seraient que des ressources. Par conséquent, l'inscription de nos travaux dans le champ théorique des SMA (Ferber, 1999) (Jennings, 1998) nous semble toute indiquée. Nous proposons d'aborder 3 points dans cette section:

1. Description générale et très synthétique de la théorie des systèmes multi-agents.
2. Présentation de quelques publications sur la gestion des compétences dans un SMA.

3. Panorama des méthodes d'apprentissage dans un SMA, notre système devant être dynamique, apprenant et autonome.

1.5.2 Paradigme agent

La littérature est abondante sur le sujet des systèmes multi-agents (Ferber, 1999) (Jennings, 1998). Nous proposons d'aborder brièvement quelques définitions du concept d'agent :

- Selon Ferber (Ferber, 1999) : « *Un SMA est un macro-système constitué d'agents autonomes qui interagissent dans un environnement commun pour réaliser une activité collective cohérente* ».
- Pour Demazeau (Briot, 2001) : « *Un agent est une entité réelle ou virtuelle, dont le comportement est autonome, évoluant dans un environnement, capable de le percevoir, d'agir dessus et d'interagir avec les autres agents* ».
- Pour Wooldridge (Wooldridge, 1998) : « *Un agent est un système informatique capable d'agir de manière autonome et flexible dans un environnement. La flexibilité étant composée de la réactivité, la pro-activité et des capacités sociales* ».

Ainsi, un SMA peut être assimilé à un environnement informatique dans lequel « vivent » des entités logicielles appelées « *agents* ». Chaque agent représente une entité physique (robot, humain, animaux, etc.) ou virtuelle (agent technique de transfert de fichiers, agent compétence dans le cadre de nos travaux de thèse). On retiendra en synthèse qu'un agent :

1. Agit dans un environnement (le SMA).
2. Communique avec d'autres agents, que ces derniers soient hébergés dans le même SMA que lui ou bien dans un SMA externe via des protocoles normalisés (FIPA, 2002).
3. Possède des ressources qui lui sont propres.
4. Perçoit de manière limitée et non globale son environnement. Il dispose ainsi d'une vision limitée voire nulle de cet environnement.
5. Possède des capacités d'action appelées comportements.
6. Peut dialoguer et offrir des services aux autres agents qui le sollicitent.
7. Peut éventuellement se reproduire, c'est à dire générer d'autres agents dans le SMA.
8. Possède des buts qu'il cherche à atteindre, des objectifs à réaliser, via l'utilisation de ressources, capacités d'action dont il dispose, perceptions de l'environnement dont il est capable, et actes de communication avec les autres agents.

L'une des caractéristiques fondamentales d'un agent est qu'il est, au moins partiellement, autonome. La littérature (Ferber, 1999) (Jennings, 1998) distingue quatre types d'agents répartis en deux catégories : cognitifs ou réactifs d'une part, téléonomiques ou réflexe d'autre part. De manière très réduite, un agent est dit « *cognitif* » s'il dispose d'une capacité d'apprentissage et de raisonnement, « *réactif* » dans le cas où il ne réagit qu'en fonction des stimuli reçus des autres agents. D'un point de vue comportemental, on distingue ceux dits « *téléonomiques* » (intentionnels dans le but de poursuivre des buts explicites) et « *réflexes* » (simplement induits par les perceptions). Le tableau 1 résume ce que nous venons de voir.

	Agents Cognitifs	Agents Réactifs
Comportement téléonomique	Agents intentionnels	Agents pulsionnels
Comportement réflexe	Agents "modules"	Agents tropiques

Tableau 1 : Les différents types d'agents

En général, un agent cognitif est intentionnel. Il cherche à atteindre des buts. Il existe cependant des agents dits « *modules* » n'ayant pas de buts précis à atteindre. C'est le cas, par exemple, des agents techniques dont nous avons parlé plus haut.
A l'opposé, les agents réactifs se divisent en 2 catégories :

- *Pulsionnel* : agent ayant une « mission » et qui déclenche un comportement si une condition n'est plus remplie dans l'environnement. Exemple : une valeur de seuil dépassée.
- *Tropique* : agent pulsionnel sans « mission », réagissant uniquement sur variation de l'environnement.

Selon Russel et Norvig (Russel, 2003), un agent peut être représenté par la figure 3.

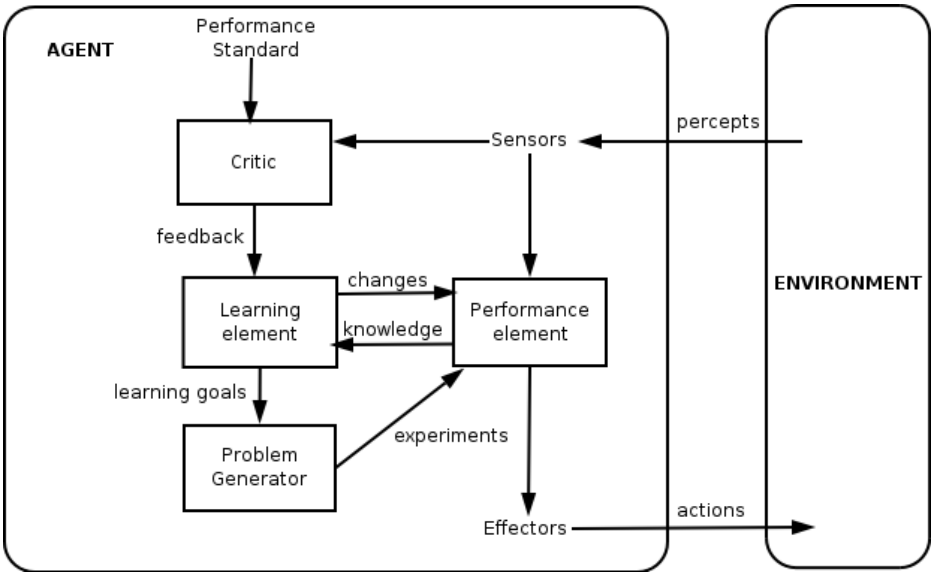


Figure 3 Modèle multi-agents générique

Deux des premières applications reposant sur les technologies multi-agents furent réalisées au début des années 1980 :

1. L'une de contrôle du trafic aérien, basée sur les travaux de Cammarata, McArthur, et Steeb (Cammarata, 1983). Il s'agit de la mise en œuvre d'études de stratégies de

coopération pour la résolution de conflits entre les plans d'un groupe d'agents, ici d'avions.

2. L'autre de surveillance de véhicules, décrit par Corkill et Lesser (Corkill, 1983). Cette application est basée sur le système DVMT (Distributed Vehicle Monitoring Task) d'analyse de mouvements.

Il existe de nombreuses définitions de ce qu'est le paradigme Agent. Nous retiendrons que c'est ici, selon Gleizes, Bernon, Migeon, et Picard, « un *paradigme pour le développement de logiciels dans lesquels l'agent est un logiciel autonome qui possède un objectif, évolue dans un environnement dynamique et interagit avec d'autres agents au moyen de langages et de protocoles »* (Gleizes, 2008). De manière classique, un objet – au sens programmation du terme – expose uniquement des attributs et des méthodes qui peuvent être accessibles depuis un programme extérieur à cet objet. Contrairement à cette approche plutôt « statique », les agents sont des logiciels dynamiques et autonomes qui introduisent une nouvelle manière de concevoir la programmation d'applications.

Dans un univers multi-agents, les applications ne sont plus implémentées sous la forme d'algorithmes où l'on décrit de manière explicite les appels de méthodes d'objet. On raisonne ici en terme de décomposition sous la forme d'agents autonomes qui échangent des informations entre eux, peuvent traiter des données issues de leur environnement et rendre des services. On retiendra finalement que le paradigme multi-agents offre la possibilité de ne plus implémenter l'algorithme de résolution d'un problème complexe de manière statique. C'est l'organisation des agents eux-mêmes qui permet cette résolution dans un environnement dynamique, évolutif et ouvert. On notera que les méthodes d'échanges entre agents au sein d'un SMA sont normalisées au travers, par exemple, d'AUML (Odell, 2001), de KQML (Finin, 1994) et de FIPA-ACL (FIPA, 2002).

Dans un Système multi-agents (Ferber, 1999) (Jennings, 199) un Agent possède des compétences qui sont mises en œuvre par des comportements. Mano, Gleizes et Glize (Mano, 2005) nous proposent, figure 4, une représentation de la mise en œuvre du comportement d'un agent où nous observons que les compétences ne sont qu'une partie des composantes internes qui le caractérisent.

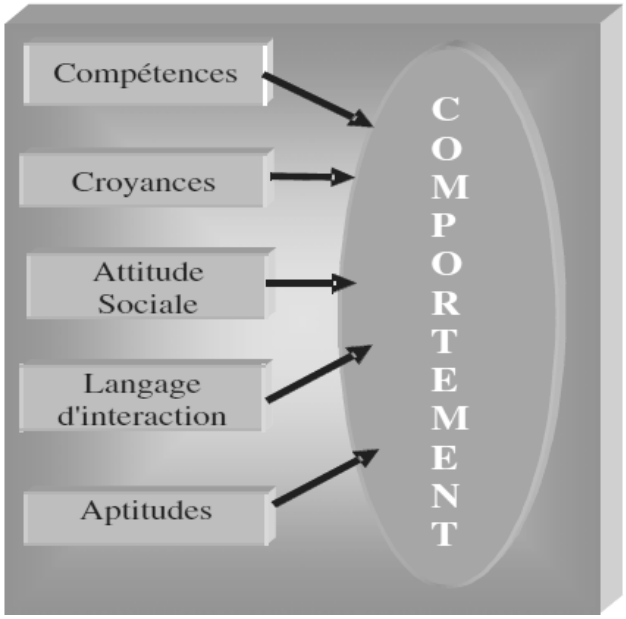


Figure 4 Les différentes caractéristiques du comportement d'un agent

Dans leur publication « *Des modèles hétérogènes de simulation par systèmes multi-agents* » (Gutknecht, 1998) Gutknecht, Ferber, et Lieurain proposent le modèle AALAADIN qui possède 3 notions de base. Citons les auteurs :

«

1. **L'agent** : nous ne posons aucune contrainte ou pré-requis sur l'architecture interne de l'agent et nous ne supposons pas de formalisme ou modèle particulier pour en décrire le comportement. L'agent est simplement décrit comme une entité autonome communicante qui joue des rôles au sein de différents groupes. Le caractère très générique de cette définition est volontaire : le constructeur de système appliquera à cette notion d'agent l'architecture d'action ou de communication idoine.
2. **Le groupe** : nous définissons le groupe comme la notion primitive de regroupement d'agents. Chaque agent peut être membre d'un ou plusieurs groupes. Dans sa forme la plus simple, un groupe peut être vu comme un moyen d'identifier par regroupement un ensemble d'agents. D'une manière plus évoluée, le groupe peut être vu comme un SMA usuel. Un point majeur de cette définition est que les différents groupes peuvent se recouper librement. Un groupe peut être fondé par n'importe quel agent, par contre, l'admission à un groupe existant résulte d'une demande formulée par l'agent candidat.
3. **Le rôle** : c'est une représentation abstraite d'une fonction, d'un service ou d'une identification d'un agent au sein d'un groupe particulier. Chaque agent peut avoir plusieurs rôles, un même rôle peut être tenu par plusieurs agents, et les rôles sont locaux aux groupes. De même qu'avec l'admission dans un groupe, la tenue d'un rôle doit être demandée par l'agent et n'est pas forcément accordée.

»

Le choix que nous avons fait dans nos travaux[2] consiste à proposer le modèle théorique des « Agents Compétence ». Dans ce nouveau modèle, les compétences ne sont plus assimilées à la seule notion de comportement d'un agent, mais **deviennent des agents à part entière**. Nous nous insérons alors pleinement dans la définition de l'agent proposée par les auteurs au point 1 ci-dessus, qui ne présume pas d'un formalisme standardisé de ce qu'est un agent.

1.6 Gestion des compétences et SMAs

Il existe effectivement des publications et des ouvrages de référence concernant la gestion des compétences via l'utilisation d'agents logiciels dans un SMA. Cependant, dans de nombreuses publications, les compétences sont simplement considérées en tant que comportements implémentés et mis en œuvre par les agents.

Le concept de « *skill agent* », tel que nous le définissons dans nos travaux, a cependant déjà été proposé dans la littérature : « *A Computational Model for a Cooperating Agent System* » (Deen, 1999). Dans cette publication, les auteurs proposent un framework opérationnel de gestion distribuée d'*hétérarchies* de tâches par l'intermédiaire de « *skill agents* ». Une hétérarchie (Mc Culloch, 1945) de tâches se distingue d'une hiérarchie par le fait qu'elle propose de mettre en avant la coopération entre les membres. Ici, une tâche globale est composée d'une hétérarchie de tâches « élémentaire » ou de plus « bas niveau ». Plutôt que d'enchainer ces dernières de manière statique pour réaliser la tâche globale, l'exécution d'une tâche élémentaire ou d'une autre se fait en fonction des évolutions de l'environnement des agents, i.e. des situations. Ainsi, les liens de précédence entre tâches élémentaires exécutées par les agents se chevauchent et s'entrecroisent. L'ascendance d'une tâche élémentaire par rapport à une autre devient alors moins évidente à déterminer, ce qui confère au système global un certain dynamisme. Synthétiquement, les « *skill agents* » dont il est question dans cette publication gèrent des tâches mais ne sont pas eux-mêmes des compétences au sens où nous l'entendons dans nos travaux.

Dans l'article « *Dynamic Skills Learning : a Support to Agent Evolution* » (Routier, 2002), Routier, Mathieu et Secq proposent qu'un agent soit tout d'abord construit sur une base élémentaire appelée « *agent atomique* ». Un agent atomique possède uniquement 2 compétences : l'une pour dialoguer avec les autres agents, l'autre pour acquérir des compétences. Une fois créé, ce nouvel agent pourra alors jouer des rôles différents au sein du SMA en fonction de l'apprentissage dynamique de compétences qu'il acquiert.

L'acquisition de ces compétences s'effectue de 2 manières, soit par paramétrage au démarrage du système, soit dynamiquement suite à transfert ou délégation de compétence depuis les autres agents. Cette publication met ainsi en avant la capacité des agents à changer de rôle au cours de leur vie, en fonction des compétences acquises. Il s'agit finalement de favoriser l'évolutivité, la réutilisabilité et la modularité des agents tout au long de leur existence dans le SMA. Pour les auteurs, d'un point de vue théorique, une compétence est un ensemble cohérent de capacités pouvant être exploitées par un agent. D'un point de vue implémentation

[2] Cf. paragraphe « *2.2 Analyse & proposition de réponse* » au chapitre 2 pour le détail

informatique concrète, une compétence est considérée comme un composant logiciel dont l'interface publique (au sens programmation objet du terme) constitue une capacité d'un agent. Cette proposition a été implémentée concrètement dans un SMA spécifique, développé par les auteurs, et appelé MAGIQUE. Ce SMA est basé sur un modèle récursif d'agents où ces derniers sont construits dynamiquement (Mezura, 1999). Dans le schéma proposé figure 5, la vision d'un tel SMA peut être qualifiée de hiérarchique (partie gauche) ou de « Agent constitué récursivement d'agents » (partie droite).

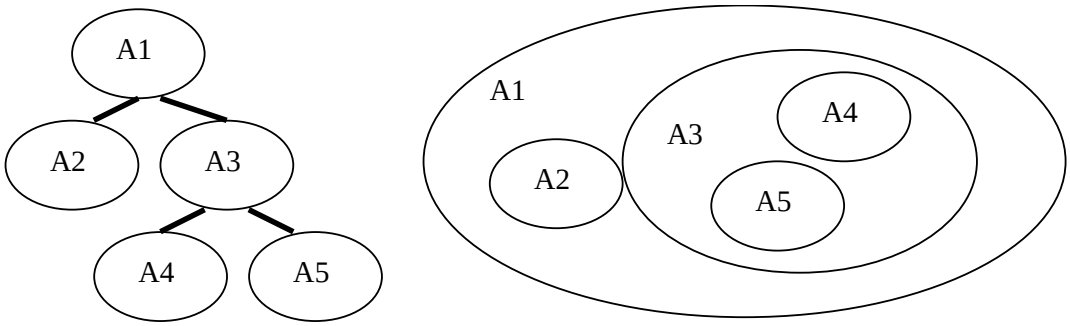


Figure 5 MAGIQUE : un SMA récursif

Dans « *Integrating Skills into Multi-Agent Systems* » (Holger, 1998), Holger, Rogalla, et Dillmann proposent une représentation encapsulée des compétences des agents au sein d'opérations élémentaires (*Elementary Operations* ou EOs). Le domaine d'activité professionnelle est celui de la robotique, les agents du SMA étant assimilés à des robots. L'objectif recherché est de rationnaliser l'utilisation des compétences exercées par les agents « robots » et d'en adapter l'utilisation en fonction des évolutions de l'environnement dans lequel ils évoluent. Il s'agit également, via les EOs, d'optimiser la gestion (enchainement, planning, etc.) des compétences des robots pour les aider à atteindre un but donné.

Dans « *An Ontology-based approach to Student Skills in Multi-Agent E-Learning systems* » (Gladun, 2007), Gladun et Rogushina proposent de construire des ontologies optimales, par domaine de compétence, destinées à améliorer l'efficacité d'un système d'E-Learning (URAN) mis à disposition d'étudiants. Il s'agit finalement d'aider les étudiants à adapter leurs méthodes d'apprentissage en comparant leur propre ontologie avec un « étalon » proposé par un tuteur. Voir figure 6.

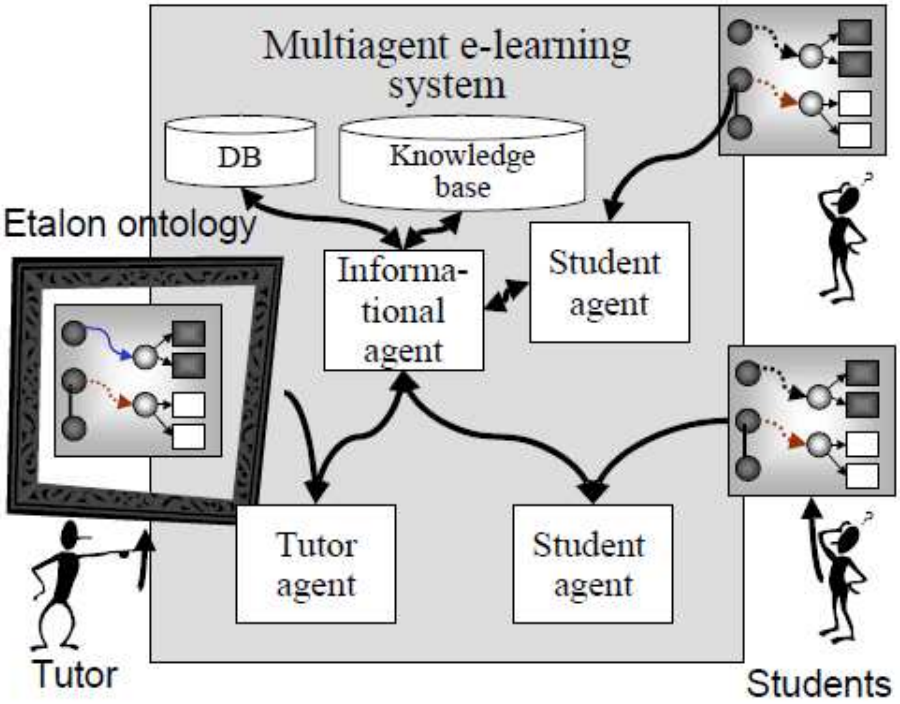


Figure 6 Architecture d'un SMA de E-Learning

Dans leur article « *Distributed planning using a framework for Devolution* » (Núñez, 1997), Núñez Suárez, Winstanley, et Griffiths proposent une solution permettant de résoudre les problèmes de planification en se basant sur les technologies multi-agents. Les agents sont capables de traiter des problèmes dans des domaines particuliers et de se fédérer entre eux pour résoudre un problème plus global. Pour ce faire, ils rejoignent des groupes de négociation dans lesquels ils expriment leurs intentions, échangent leurs buts respectifs et conditions de réalisation. Il y a alors comparaison des différentes propositions des agents avec dégagement d'un consensus et constitution d'un planning d'enchainement de ces propositions. L'article introduit un framework de développement, un protocole de communication inter-agents dédié et une méthode d'élaboration d'un planning cible à partir des propositions négociées. La négociation est inspirée des travaux de Jin & Koyama dans « *Multiagent planning through expectation based negotiation* » (Jin, 1990). Un agent ne joint le groupe de négociation que si les échanges dont il est question ont un effet sur ses intérêts et si les pré-conditions permettant l'atteinte de son but sont réunies. Voir figure 7.

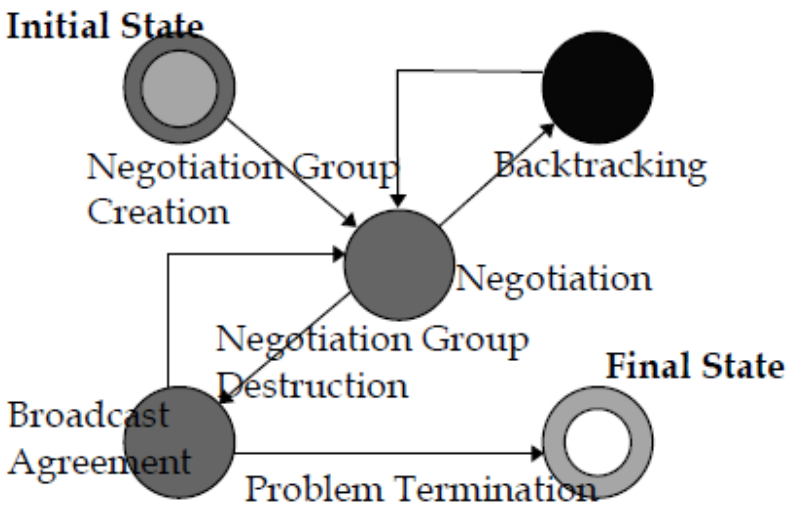


Figure 7 Vue du réseau de transitions d'un planning négocié

1.7 Panorama de méthodes d'apprentissage dans les SMAs

1.7.1 Pourquoi exploiter des techniques d'apprentissage ?

Dans le modèle décrit au chapitre suivant, les compétences sont des agents à part entière notés AC. Chaque AC peut être vu comme une entité dynamique et autonome. Il cherche à réaliser de la manière la plus pertinente qui soit (i.e. relativement au contexte professionnel d'utilisation du système) certains buts principaux que sont :

1. L'auto définition, c'est-à-dire à identifier les compétences élémentaires qui le composent. Ces compétences élémentaires seront parfois renseignées directement par l'utilisateur du système, mais pas toujours et pas toujours de manière identique. Nous avons donc un premier besoin d'apprentissage automatique pour que la compétence détermine sa propre définition. Par exemple, pour la compétence panneaux solaires, un utilisateur peut indiquer plus précisément une compétence élémentaire de pose sur le toit et une autre de raccordement électrique, alors qu'un second ne mentionnera pas le raccordement et un troisième peut simplement demander une personne compétente en panneaux solaires sans rien préciser de plus. Comment apprendre la bonne définition de cette compétence afin de sélectionner un acteur pertinent ?
2. L'établissement des liens avec d'autres ACs en fonction des projets auxquels ils participent de manière conjointe. Un utilisateur du système peut en effet créer un nouveau projet dont la finalité est très proche d'un projet existant. En apprenant les différents liens existants entre les agents compétences, le système doit pouvoir suggérer un ensemble de compétences nécessaires à la réalisation du projet.
3. L'identification pertinente (en intégrant des résultats des retours d'expérience passés) des acteurs capables de les mettre en œuvre. L'apprentissage doit ici obéir à différentes stratégies, afin de ne pas proposer systématiquement le même acteur pour la même compétence, même si ce dernier a obtenu les meilleures évaluations.

Notre souhait est de tendre vers un certain optimum de l'efficacité de nos AC, afin qu'ils s'adaptent au mieux au contexte professionnel d'utilisation du système. Plus le nombre de projets en historique y sera important, plus l'adéquation au domaine fonctionnel sera élevée, nos ACs pouvant avoir un comportement variable au cours de leur évolution. Cette adaptabilité suggère donc des règles comportementales qui évoluent en fonction de la maturité acquise. L'apprentissage joue ainsi un rôle clé dans notre système. Afin de faire les bons choix, nous proposons une revue approfondie des différentes méthodes du domaine.

1.7.2 Définition

En Intelligence Artificielle, Simon (Simon, 1969) fournit la définition suivante : « *L'apprentissage induit des changements dans le système qui sont adaptatifs dans le sens qu'ils permettent au système de faire la même tâche une nouvelle fois plus efficacement* ». Dans nos travaux, cette notion d'apprentissage est essentielle puisque le système que nous cherchons à mettre en place se doit d'être dynamique, évolutif et apprenant.

1.7.3 Les différents modes d'apprentissage

Selon Mitchell (Mitchell, 1997), on distingue principalement (et de manière synthétique) 4 modes d'apprentissage :

1. **Supervisé** (Biemacki, 2003). Il s'agit ici de prédire le groupe ou « classe » d'appartenance d'une donnée cible (un patient par exemple) en fonction des valeurs de variables dites « prédictives » (les symptômes) qui la caractérisent. Cette déduction repose sur un modèle (valeurs de données prédictives et classe cible), fourni au préalable par un expert, reposant sur des exemples réels. Dans ce mode d'apprentissage, il y a 2 étapes : l'une que l'on peut qualifier d'apprentissage (ou de paramétrage) statique et l'autre de calcul de la classification de la donnée cible (notre patient) compte tenu des valeurs (les symptômes) des données prédictives. Les méthodes les plus représentatives de ce mode d'apprentissage sont :
    a. *k plus proches voisins* : calcul de la distance euclidienne avec les autres données cibles et affectation à la classe contenant le plus grand nombre de données cible parmi les k plus proches voisins.
    b. *Classification Bayésienne* : on applique une formule mathématique de calcul probabiliste pour identifier la meilleure classe d'appartenance de la nouvelle donnée cible.
    c. *Séparateurs à Vastes Marges (ou SVM)* (Vapnik, 1998) : repose sur le calcul de marges par rapport à un hyperplan séparant 2 classes (pouvant être linéairement séparables).

2. **Non supervisé** (Grabmeier, 2002). Contrairement au mode supervisé, la classification des données cibles n'est pas connue au départ. Le système ne dispose que des exemples de données cibles et des valeurs des données prédictives de chacune d'elles. L'algorithme de

traitement va donc chercher à répartir automatiquement les données cibles dans des « classes » en fonction de valeurs communes de données prédictives (souvent par calcul de distance entre les données cibles). Cette méthode peut permettre de faire émerger des conclusions auxquelles on ne s'attendait pas compte tenu de la combinatoire des valeurs des données prédictives d'entrée fournies au système. Les méthodes les plus représentatives de ce mode d'apprentissage sont :

a. *Les k-moyennes* : cette méthode consiste à répartir les données cibles en classes homogènes. Pour ce faire, elle cherche à minimiser la variance intra-classe.
b. *Self Organizing Map (SOM) ou carte de Kohonen* (Kohonen, 2000) : cette méthode appartient au domaine des réseaux de neurone non supervisés. Elle distingue un espace de grande dimension (celui des données) et un de faible dimension (celui de la carte sur laquelle sont représentés les neurones sous forme géométrique). On cherche, par un calcul sur le poids, à associer des données de l'espace de grande dimension à des neurones de la carte qui contiennent déjà des données proches (au sens distance) de la donnée à classer.
c. *Classification Ascendante Hiérarchique (CAH)* (Lance, 1967) : cet algorithme appartient à la méthode des Heuristiques et introduit des variations dans le calcul des distances (lien simple, lien complet, lien moyen, centroïde et critère de Ward).

3. **Semi-supervisé** (Vapnik, 1998) : ce mode, encore appelé « *apprentissage transductif* » peut être vu comme l'association des 2 modes vus ci-dessus. Certaines (voir la plupart) des données cibles n'ont pas été classées par un expert. Le système apprend alors selon les deux modes avec l'intervention d'un expert qui valide ou non les propositions du système.

4. **Par renforcement** (Sutton, 1998) Il s'agit d'une méthode d'apprentissage par essai/erreur qui sera détaillée plus loin dans ce document.

1.7.4 L'apprentissage dans un SMA

Dans un système multi-agents, un agent doté de capacités d'apprentissage analyse les variations de son environnement, s'adapte et réagit en fonction. Il déclenche ainsi un comportement adapté à la typologie des variations observées. L'objectif de cet apprentissage consiste, pour un (unique ou un groupe d') agent à atteindre des objectifs fixés ou non à l'avance (donc potentiellement évolutifs). L'atteinte de ces buts est représentée par la détermination d'une « *politique* » ou « *stratégie* ». A ces éléments vient s'ajouter le fait qu'un agent peut recevoir des messages (ou stimuli) en provenance d'autres agents, ces derniers influant alors sur les données d'apprentissage du premier.

La littérature que nous avons parcourue fournit 4 méthodes principales d'apprentissage pour les agents :

1. Processus Décisionnels de Markov (Putterman, 1994) avec apprentissages par renforcement (Mitchell, 1997).
2. La théorie des jeux : jeux matriciels (Von Neumann, 1994) et jeux stochastiques (Shapley, 1953).

3. Les réseaux bayésiens (Boutilier, 1999) (Dean, 1989).
4. Les méthodes de Case-Based Learning (Plaza, 2002) (Arkin, 2003).

Nous présentons également dans cet état de l'art deux autres paragraphes. Le premier est consacré aux méthodes d'apprentissage automatique symbolique dont les principales techniques sont les arbres de décision (Quinlan, 1994), la Programmation Logique Inductive (PLI) (Cornuéjols, 2002) et l'Inférence Grammaticale (IG) (De La Higuera, 2002). Le second paragraphe proposé concerne les algorithmes d'apprentissage incrémental (Borodin, 1998).

Dans un système multi-agents, on distingue 3 types d'agents (Labarre, 2005) :

1. Les agents indépendants : possèdent des buts différents de notre problématique d'apprentissage. Ils sont considérés comme des « *bruits qu'on pourrait assimiler aux variations de l'environnement* ».
2. Les agents collaboratifs : posent un problème pour l'apprentissage. Il devient difficile de savoir comment répartir les bénéfices globaux apportés par le système au niveau de chacun des agents ayant collaboré.
3. Les agents compétitifs : ont des buts opposés. L'esprit de compétition induit que l'atteinte des buts intègre éventuellement la notion de défaite des autres agents impliqués.

Dans la présente étude bibliographique, nous proposons :

1. D'expliciter les différents concepts et méthodes d'apprentissage exprimés dans ce paragraphe d'introduction.
2. De décrire la problématique spécifique que nous abordons, en termes d'apprentissage, dans le cadre de nos travaux de thèse.
3. D'identifier le modèle d'apprentissage que nous considérons le mieux adapté à cette problématique.

1.7.5 Processus Décisionnel de Markov

La littérature nous décrit un processus de décision markovien (MDP) comme étant issu de la *théorie de la décision* (Kast, 2002) et de la *théorie des probabilités* (Jacod, 2003). Un tel processus s'insère dans le traitement des *problèmes de décision séquentielle dans l'incertain* (Bernstein, 2002) (Becker, 2004). L'un des textes principaux est celui de Sutton et Barto (Sutton, 1998). En IA, on observe que prendre des décisions de manière *séquentielle* permet la résolution de problèmes de planification, ce qui peut être illustré par le calcul du chemin le plus court dans un graphe. On observe également dans ce type de problème que la prise de décision engendre des conséquences inconnues de l'agent lui-même (donc *incertaines*) au moment où il la prend. Un modèle de prise de décision séquentielle dans l'incertain couple ces 2 observations.

De manière concrète, les MDP sont l'expression mathématique de ce modèle bivalent : approche du plus court chemin dans un environnement dit *stochastique*, c'est-à-dire composé

d’un ensemble de variables aléatoires. Un MDP intègre 3 composantes essentielles, **pour un seul et unique agent** :

1. Un *état* interne qui décrit la situation (pas au sens localisation) de l’agent à chaque instant.
2. Un ensemble d’*actions* disponibles qui permettent d’assurer le dynamisme de l’état interne.
3. La notion de *revenu* (ou récompense) associée à chacune des transitions d’état (Putterman, 1994).

La résolution d’un MPD consiste à définir ce que l’on nomme *des politiques, stratégies, ou encore règles de décision, qui spécifient l’action à entreprendre à chacune des étapes pour toutes les situations futures possibles de l’agent* (PDMIA, 2008). Une politique peut être qualifiée de :

- *Déterministe* : on sait avec précision quelle action choisir pour chaque situation rencontrée.
- S*tochastique* : on fournira une distribution de probabilités sur les actions à effectuer.

L'objectif recherché par l’agent est de maximiser ses récompenses en cherchant à atteindre la politique optimale. Pour ce faire, on utilise plusieurs critères qui définissent l'importance des récompenses espérées pour l'agent. Les MDPs permettent de modéliser l’évolution du monde dans lequel évolue l’agent. Si l’on suppose que l’agent a connaissance des paramètres du modèle, il en résulte que ce dernier peut résoudre le MDP en utilisant un algorithme d’itération des politiques ou des valeurs dans le but de construire une politique déterministe (Belman, 1957) (Bertsekas, 1987). La résolution de MDP est au cœur de la notion d'*apprentissage par renforcement* (Mitchell, 1997).

1.7.6 Apprentissage par Renforcement

Dans le paragraphe précédent, nous avons vu que l’agent a la connaissance de ses fonctions de transition d’état (au travers des politiques, stratégies ou règles de décision) et de récompense. L’un des problèmes posés par cette méthode apparait justement lorsque l'agent n’a pas connaissance de ces fonctions. La détermination d’une politique optimale doit alors se faire en suivant une approche expérientielle, par essai / erreur, dans le but d’effectuer des actions en fonction de chaque situation dans laquelle l’agent se trouve. On cherchera ainsi à pouvoir estimer directement la valeur de la fonction d’actions sans avoir à apprendre les différentes probabilités de transition d’état. La détermination de l’action optimale sera alors effectuée en fonction de l’état de l’agent. Il semble utopique de vouloir présenter et détailler tous les axes de recherche concernant les MDP et l’apprentissage par renforcement tant le domaine est vaste. On citera par exemple (liste non exhaustive) :

- Apprentissage par renforcement en temps (Degris, 2012).
- Apprentissage par renforcement relationnel (Rodrigues, 2010).
- Apprentissage par renforcement de grande taille (Degris, 2006).

- Apprentissage par renforcement hiérarchique (Diettrich, 2000).
- Systèmes de classeurs (Gerard, 2002).
- Méta-apprentissage (Schmidhuber, 1996).
- Représentations prédictives de l'état (Littman, 2001).
- MDP factorisés (Boutilier, 1995).
- **TD-Learning** (Sutton, 1998) (Sutton, 2008) (Bradtke, 1996) (Bertsekas, 1996) et **Q-Learning** (Sutton, 1998) (Labarre, 2005) (Watkins, 1989) (Jaakkola, 1994) (Littman, 1994) (Hu, 1998) (Bowling, 2000).
- **R-Max** (Brafman, 2001) : extension de l'algorithme E^3 qui a pour postulat un environnement statique.
- **Friend-or-Foe-Q** (Littman, 2001) : application aux jeux de Markov à somme quelconque avec utilisation des équilibres de Nash compétitifs et coopératifs.
- **PHC et WoLF** (Bowling, 2001) : concept de « pas d'apprentissage » avec comparaison de l'espérance de 2 politiques dites « habituelle » pour la première et « moyenne » pour la seconde.
- **SARSA** pour **State-Action-Reward-State-Action** (Rummery, 1994).

1.7.7 La théorie des jeux

1.7.7.1 Principes généraux

Parmi (liste non exhaustive) les publications faisant référence dans le domaine, on cite :

1. L'ouvrage «*Theory of Games and Economic Behaviour* » (Von Neumann, 1944).
2. Les 4 articles « *The bargaining problem* » (Nash-1, 1950), « *Equilibrium points in n-person games* » (Nash-2, 1950), « *Non cooperative games* » (Nash, 1951), et « *Two-person cooperative games* » (Nash, 1953) de Nash.
3. L'article « *Stochastic games* » (Shapley, 1953) de Shapley.
4. L'article « *An iterative method of solving a game* » (Robinson, 1951) de Robinson.
5. L'ouvrage « *The theory of Learning in Games* » (Fudenberg, 1998) de Fudenberg & Levine.

Parmi toutes les définitions disponibles dans la littérature, nous retiendrons la suivante : *la théorie des jeux est un ensemble d'outils analytiques qui ont été développés pour nous faciliter la compréhension des situations d'interaction entre des décideurs (agents, joueurs) rationnels* (Yildizoglu, 2003). Il est intéressant de noter, toujours en référence de ce même auteur, que les hypothèses de base de cette théorie sont les suivantes :

1. *Les décideurs sont rationnels,* ce qui signifie qu'ils visent à atteindre des objectifs non liés et en dehors du monde dans lequel ils évoluent (un SMA pour nous).
2. les décideurs intègrent dans leur processus de décision leur capacité d'anticipation du comportement des autres décideurs.

Au point 1 ci-dessus, on entend exprimer par le terme « *rationnel* » le fait que l'agent (le décideur dans notre cas) tente d'arriver à la situation la meilleure pour lui-même, sans « esprit de sacrifice » vis-à-vis des autres. On distingue 3 différents contextes d'interaction possibles entre les agents et basés sur :

1. La typologie relationnelle existante entre les agents : sont ils coopératifs entres eux?
2. Une dimension temporelle : interaction séquentielle ou bien simultanée ?
3. Le niveau de connaissance de l'environnement dont disposent les agents : complet ou bien incomplet ?

Nous nous focaliserons dans ce qui suit sur la présentation de jeux matriciels et jeux stochastiques. On citera cependant, pour information, les notions de :

- **Jeux fictifs** (Brown, 1951) : les agents possèdent des croyances individuelles concernant les stratégies suivies par les autres agents. Les croyances sont exprimées sous la forme d'une statistique calculée à partir des actions passées des autres agents. Les résultats de ce calcul sont alors considérés comme étant les probabilités de leurs actions à venir.
- **Jeux adaptatifs** (Young, 1998) : c'est une variante du jeu fictif. On introduit ici une exploration stochastique des actions ainsi que la suppression d'influence des croyances

initiales. L'important est ici l'utilisation d'une mémoire limitée (stockage des dernières actions conjointes entre agents au temps t).

1.7.7.2 Jeux matriciels

L'un des premiers ouvrages abordant cette problématique est « *An iterative method of solving a game* » (Robinson, 1951). Ici, on l'aura deviné, les jeux se représentent sous la forme de matrices. Ce sont donc des tuples (n, $A_{1..n}$, $R_{1..n}$) où, dans le contexte adapté à nos agents :

- n est le nombre d'agents.
- Ai est l'ensemble des actions que l'agent peut réaliser.
- Ri la matrice n-dimensions $A_1...A_n$ qui donne les bénéfices possibles pour chacune des combinaisons possible des actions des agents.

Comme dans le cas du paragraphe consacré aux MPD avec apprentissage par renforcement, l'objectif de chaque agent est ici de maximiser ses récompenses. Cependant, dans un contexte des jeux matriciels et de compétition permanente, rechercher la meilleure stratégie pour un agent donné nécessite de connaître les matrices de tous les autres agents.

1.7.7.3 Equilibre de Nash

Une notion fondamentale reste toujours présente dans les jeux matriciels : celle d'équilibre de Nash (Nash, 1951). Selon son auteur, l'équilibre de Nash *est un concept de solution dans lequel l'équilibre entre plusieurs joueurs, connaissant leurs stratégies réciproques, est devenu stable du fait qu'aucun ne peut modifier sa stratégie sans affaiblir sa position personnelle* (Nash, 1951). L'équilibre de Nash tend à permettre d'atteindre une situation stable. Selon les références théoriques, l'algorithme d'apprentissage convergerait vers un équilibre de Nash. A titre informatif, l'équilibre de Nash a été généralisé suite à l'invention d'équilibre de Nash corrélé (Aumann, 1974) qui instaure l'introduction d'un « arbitre » qui affecte différentes stratégies à chacun des agents selon une certaine loi de probabilité.

1.7.7.4 Jeux stochastiques

L'article fondateur de cette théorie est « *Stochastic games* » (Shapley, 1953). L'article « *Markov games as a framework for multi-agent reinforcement learning* » (Littman, 1994) a défini la modélisation de l'apprentissage multi-agents sous la forme de jeux stochastiques comme pertinent. Ces jeux sont également appelés jeux stratégiques ou jeux de Markov. Ils peuvent être assimilés à la fusion des 2 modèles que nous avons vus dans les paragraphes précédents, à savoir MDP & jeux matriciels. Cette nouvelle modélisation embarque donc avec elle la possibilité d'existence d'équilibres de Nash. La littérature parcourue (très riche et variée) nous conduit à penser que les jeux stochastiques sont le modèle de base souvent mis en œuvre dans les algorithmes d'apprentissage des SMA et que la problématique posée ici concerne surtout la définition des mécanismes d'interaction entre agents.

On distingue essentiellement 2 types de jeu stochastiques en termes d'attribution de récompense :

1. *A somme nulle* : la somme des récompenses de tous les agents est toujours égale à 0, ce qui signifie que l'on intègre les notions de gagnant / perdant si le jeu est effectué par 2 agents uniquement. Ce type de jeu est purement compétitif.
2. *A somme quelconque (ou non nulle)* : on attribue une valeur de récompense différente pour chaque agent.

De même, les jeux peuvent être :

1. *A information parfaite* : chaque agent a une connaissance exacte de la situation du jeu à tout moment.
2. *A information imparfaite* : chaque agent n'a qu'une connaissance partielle de l'environnement et des informations concernant les autres agents du système.
3. *Simultanés* : tous les agents peuvent agir en même temps (pas de notion d'enchainement)
4. *Coopératifs* : certains agents se mettent ensemble dans le cadre d'un intérêt commun face aux autres agents ou autres groupes d'agents éventuels. Le plus difficile dans ce contexte est de déterminer la répartition (égalitaire ou non) de la récompense entre les agents « coalisés » (Claus, 1998).

Un jeu stochastique :

1. Est utilisé dans le cas où les agents suivent des buts essentiellement divergents.
2. Est dynamique et possède des transitions stochastiques dans le but de formaliser la notion d'incertitude.
3. Induit que le choix effectué par un agent dépend de ce que font les autres.
4. Ne couvre pas un contexte ou des équipes d'agents poursuivraient un objectif commun.

1.7.7.5 Algorithmes

Quelques exemples illustratifs :

- **IL et JAL** (Claus, 1998) : limitation aux jeux matriciels coopératifs. Ce dernier mot est à prendre au sens « communiquant ».
- **NashQ-learning** (Greenwald, 2003) : appliqué aux jeux stochastiques non-coopératifs, avec récompenses décorrélées.
- **CE-Q learning** (Greenwald, 2003) : *Correlated Equilibria* est une variante de NasQ-Learning. L'apprentissage des valeurs des équilibres corrélés est préféré à celle des équilibres de Nash.

- **Friend-or-Foe Q-learning** (Littman, 2001) : appliqué à différentes classes de jeux stochastiques. C'est la combinatoire de 2 algorithmes compétitifs et collaboratifs qui tend à prouver la convergence vers un équilibre de Nash.
- **Hyper-Q learning** (Tesauro, 2004) : apprentissage de la valeur des stratégies conjointes mixtes plutôt que celles des actions conjointes (stratégies conjointes pures).
- **IGA** (Singh, 2000) : *Infinitesimal Gradient Ascent* intègre une croissance de gradient de valeur faible liée à un taux d'apprentissage variable tendant vers zéro. Cet algorithme s'accompagne d'une preuve de convergence vers un équilibre de Nash.

1.7.8 Apprentissage bayésien

1.7.8.1 Principes généraux

Elayeb, dans le cadre de sa thèse à l'université de Toulouse (Elayeb, 2009), fournit au chapitre 3 une description synthétique et bien illustrée de ce qu'est un réseau bayésien. Nous en présentons les principes fondamentaux ci-après. Les réseaux bayésiens (Howard, 1981) (Pearl, 1988), également connus sous la dénomination "*Belief Networks*" ou "*Causal Networks*", sont la combinaison des approches probabilistes et de la théorie de graphes. Ils permettent de représenter des situations de raisonnement probabiliste à partir de connaissances incertaines. Dans un RB, les nœuds sont les variables (états des agents pour nous) du système et les arcs les relations de causalité entre les nœuds. Il existe, entre les nœuds liés, une valeur probabiliste de passer du nœud « père » au(x) nœud(s) « descendant(s) ».

La constitution d'un RB doit aboutir à la modélisation — sous la forme d'un graphe non cyclique — des dépendances conditionnelles des différentes variables entres elles. Cette constitution suit 2 phases (Hallouli, 2004) :

- **Phase 1 : Apprentissage** (François, 2003) → On cherche à trouver la structure et les probabilités associées du réseau, à partir des informations contenues dans une base de données et de traitements principalement statistiques. 2 approches sont disponibles (François, 2003) :
  1. *Méthodes de marquage* : génération d'un graphe déconnecté, utilisation de méthodes de recherche pour ajouter des arcs et tester, via l'utilisation d'un score, si la nouvelle structure est meilleure que l'ancienne.
  2. *Méthodes d'analyses de dépendances* : analyse des dépendances des données en base et déduction de la structure.
- **Phase 2 : Inférence** (Cooper, 1990) (Dagum, 1993) → En utilisant les résultats obtenus lors de la première phase, le RB propage l'information à l'intérieur de la structure, permettant toute interrogation sur la base. Le RB peut alors restituer, pour chaque état partiel ou complet de la base (instanciation partielle ou complète des variables de celle-ci) des probabilités d'occurrence de toutes les valeurs possibles de toutes les variables.

1.7.8.2 Algorithmes

On distingue 2 types d'algorithmes :

1. Inférence exacte :
   - **Clustering** (Lauritzen, 1988) : effectue l'inférence en transformant le réseau en un arbre pour lequel chaque nœud regroupe plusieurs nœuds du réseau initial.
   - **JLO, dit « algorithme de l'arbre de jonction »** (Lauritzen, 1989) (Jensen, 1990) : est utilisé dans le cadre de réseaux comprenant uniquement des variables à valeurs discrètes.
   - **Dawid** (Sawid, 1992) : s'appuie sur la modélisation sous forme de RB d'une fonction de probabilité.
2. Inférence approchée (ici illustré avec des algorithmes d'échantillonnage stochastiques) :
   - **Logic sampling** (Henrion, 1988) : utilise l'échantillonnage probabiliste logique basé sur une notion de rejet.
   - **Likelihood weighting** (Fung, 1989) : utilise les pondérations par vraisemblance.
   - **Autres algorithmes:** Forward-Backward sampling (Fung, 1994), Self importance (Shachter, 1989), Heuristic importance (Shachter, 1989).

1.7.9 Case Base Learning

1.7.9.1 Principes généraux

CBL (Case Base Learning) est une approche relevant des méthodologies de Case Base Reasoning (CBR) encore appelées « Raisonnement à partir de Cas » (RAPC) en langue française. L'objectif de ces méthodes est de résoudre de nouveaux problèmes (*target cases*) à partir d'expériences passées (*case base*), de solutions apportées à des problèmes similaires dont on a stocké les caractéristiques en mémoire (Schank, 1982) (DARPA, 1991) (Schank, 1990) (Kolodner, 1993). CBR est une combinaison de connaissances et de processus de gestion et de réutilisation d'expériences passées. Un *cycle* (une itération) CBR se décompose en 5 étapes :

1. **Présentation** : formatage (au sens sérialisation si l'on prend une image programmation objet) du problème soumis au système (*target case*) de telle sorte qu'il soit en adéquation avec le format des contenu des « souvenirs » stockés en mémoire et devienne ainsi compatible avec les méthodes de récupération (étape 2 ci-dessous).
2. **Récupération** : extraction en mémoire des problèmes similaires (*case base*) qui y ont été stockés lors des inférences précédentes.
3. **Adaptation** : modification de la solution du cas source élu dans le but de construire la solution au *target case*.
4. **Validation** : évaluation de la solution suggérée à l'étape d'adaptation pour le *target case*. Si la solution n'est pas satisfaisante, une correction est effectuée.

5. **Mise à jour** : mise à jour de la connaissance du système à l'issu du raisonnement avec intégration du *target case* en mémoire (Aamodt, 1994) (Fuchs, 2006).

Essentiel à noter : contrairement aux mécanismes vus précédemment (MPD, théorie des jeux et réseau bayésien), la validation et l'évaluation des décisions prises par le système est au cœur des méthodes de RAPC. Il y a donc une interaction forte entre système apprenant et acteurs humains, au moins dans un premier temps. Ce n'est probablement pas si rédhibitoire pour certains systèmes car on tendra probablement, dans certains domaines fonctionnels, vers une stabilisation conforme aux attentes humaines. Par anticipation, et comme nous le verrons plus loin, le modèle d'apprentissage RAPC colle très bien à nos travaux dans le cadre de la constitution d'une bourse de compétences.

Le RAPC tente de répondre à des problèmes où la modélisation de systèmes à base de connaissance, reposant sur des méthodes déclaratives de règles comportementales, s'avère particulièrement difficile. La littérature à laquelle il est fait référence plus haut dans ce paragraphe nous décrit l'utilisation du RAPC comme étant efficace si l'on considère certaines contraintes. Citons, dans l'énumération ci-dessous et en *italique*, les éléments fournis par Loriette-Rougegrez (Loriette-Rougegrez, 1998) :

1. L'expérience représente la plus grande partie des connaissances sur un sujet.
2. Les experts parlent de leur domaine en citant des exemples.
3. Les problèmes ne sont pas complètement compris (modèles faibles, connaissances du domaine faibles).
4. Beaucoup d'exceptions au cas général.
5. Besoin de construire une mémoire d'entreprise.
6. Un volume de données satisfaisant doit exister.

Les mécanismes offerts par l'approche RAPC permettent de se décorréler complètement du domaine fonctionnel représenté par le système dans lequel on évolue. L'apprentissage ne repose pas sur des règles préétablies mais sur l'expérience. Quelques illustrations (liste non exhaustive) de systèmes RAPC modélisant les cas en tant qu'objets : REBECAS (pREdiction Basée sur dEs CAS) (Loriette-Rougegrez, 1994), FLORAN (Filtrage fLou et Objet pour Raisonner par Analogie) (Salotti, 1992), CAREFUL (Case REtrieval based on FUzzy Logic) (Jaczynski, 1994), WIND-1 (Hansen, 2000) utilisé pour la prédiction météorologique.

Les systèmes RAPC peuvent être classés en 2 catégories (Loriette-Rougegrez, 1998) :

1. **Applications à « situation statique »** : c'est le modèle précurseur. Toutes les variables définissant le problème sont connues à l'avance et fixées. Il n'y a donc pas de problème de structuration entre le nouveau problème et les « souvenirs » stockés en mémoire. Les variables sont souvent représentées en base de données sous la forme d'une liste de type a=v où a est l'attribut et v sa valeur. Exemple illustratif : l'application CHIEF (Hammond, 1986).
2. **Applications à « situation dynamique »** : contrairement au modèle précédent, on traite des cas où les valeurs des variables ne sont pas connues à l'avance et évoluent en fonction des situations. On recherche les cas similaires en mémoire.

1.7.9.2 RAPC & modèle multi-agents

Une illustration de la déclinaison du RAPC au paradigme multi-agents est le problème des proies et prédateurs (Haynes, 1988). Dans cette expérimentation, chaque agent prédateur peut apprendre les (donc accéder aux) cas comportementaux des autres prédateurs. Ainsi, un agent prédateur peut prédire le mouvement des autres prédateurs de manière à renforcer la coordination globale du groupe (Ontañón, 2005).

Les SMA basés sur les techniques de RAPC peuvent être classés selon différents critères dont l'organisation du stockage de la connaissance (des cas) au sein du système (centralisée ou non), la méthodologie de traitement des cas (un seul ou n agents exécute(nt) le raisonnement sur les cas à chaque cycle). Traitant spécifiquement de l'autonomie des agents dans le cas des RAPC, on distingue deux facteurs clé pour un agent que sont : sa capacité à déterminer s'il est qualifié pour résoudre le problème, sa capacité à échanger avec les autres (négociation, collaboration) dans le but d'obtenir une solution au problème posé (Plaza, 2005). Au sein du processus de traitement de la connaissance (étape d'adaptation), il existe également deux manières de concevoir les choses :

- **Raisonnement au niveau mono-agent (agent unitaire)** : chaque agent du SMA doit être en mesure de répondre au problème. La conséquence est qu'il doit disposer d'une vision globale des cas mémorisés et des mécanismes de résolution. Exemples : POMAESS (Yua, 2003), CCBR (Mc Ginty, 2001).
- **Raisonnement au niveau SMA global (niveau multi-agents)** : tout ou partie des cinq étapes du cycle CBR sont réparties sur un ou plusieurs agents. Si l'avantage de la répartition est évident, la contrepartie de cette approche est de limiter les capacités des agents et d'introduire une complexité en raison de la gestion des échanges et de la synchronisation à gérer. Un juste équilibre doit être trouvé selon le « découpage agent » en vigueur au sein du SMA. Exemples : PROCLAIM (Tolchinsky, 2006), CBR-TEAM (Prassad, 1996).

1.7.9.3 RAPC dynamique dans un contexte SMA

Au fur et à mesure que le système évolue et apprend, les « souvenirs » stockés en mémoire font de même. Dans cette configuration, le raisonnement doit prendre en compte ce fait et ne pas considérer chaque décision comme étant rattachée à un nouveau cas. Deux aspects importants doivent être considérés lorsque l'on parle de raisonnement dynamique :

1. En cohérence avec le déroulé de la vie humaine, le dynamisme n'empêche pas l'acquisition continuelle de nouvelles connaissances.
2. Le dynamisme est vu comme incrémental (mise à jour de données) sur les connaissances déjà acquises et mémorisées dans le but de s'améliorer sans cesse.

La littérature parcourue propose différents mécanismes de mémorisation. Entre autre, les travaux de Shank (Shank, 1982) proposent une « *théorie de la mémoire dynamique* ». Les hypothèses principales formulées par cet auteur sont que certains processus sont indissociables (se souvenir, comprendre, expérimenter et apprendre) et que l'expérience influe sur l'évolution de la mémoire. Dans le schéma qui suit, les MOPs (Memory Organization Packets) permettent l'organisation d'un ensemble de scènes dans le but d'atteindre un but. Les méta-MOPs, quand à eux, organisent les MOPS. Les TOPs (Thematic Organization Packets) structurent les connaissances en fonction de similitudes existantes entre différentes situations. Voir figure 8.

Figure 8 Vue schématique de la théorie de la mémoire dynamique (Shank, 1982)

Citons également l'importance des travaux de raisonnement par analogie où il s'agit de s'appuyer sur un cas connu (validé) pour inférer des informations sur un autre qui l'est moins. On identifie deux types de raisonnements par analogie : heuristique ou recours (Coulon, 1990). Dans le premier cas (heuristique), on dispose de la connaissance complète des données du problème à traiter. On recherche ainsi une solution dans les cas validés mémorisés et on optimise les temps de réponse du système. Dans le second cas (recours), on ne dispose que de données partielles décrivant le problème et on cherche à proposer une solution sans être certain que cette dernière résolve effectivement le problème.

1.7.9.4 Algorithmes

Quelques exemples illustratifs :

- **K-plus proches voisins** (Mc Sherry, 2006) : assure que l'ensemble des réponses au problème posé soit non-vide.
- **Hill Climbing** (Mühlenbein, 1992) : ajuste progressivement les poids lors de la phase d'adaptation de manière à minimiser l'erreur de résolution.
- **Fuzzy decision Tree** (Shiu, 2001) : construction d'arbres de décision flous.
- **C4.5** (Quinlan, 1993) et **RISE** (Domingos, 1996) : obtention de connaissances d'adaptation plus générales à partir d'un ensemble de cas d'adaptation.
- **Charm** (Zaki, 2002) : algorithme de « fouille » qui effectue de manière efficace l'extraction de motifs fermés fréquents (MFF).

1.7.10 Apprentissage symbolique

1.7.10.1 Principes généraux

Dans l'apprentissage symbolique (AS), le système apprend en construisant une représentation symbolique d'un ensemble de concepts à partir d'exemples et de contre-exemples de ces

derniers. L'AS est généralement représenté sous la forme d'expressions logiques, d'arbres de décision, de règles de production ou de réseaux sémantiques. Exemples de tels systèmes :

1. Le programme ARCH (Winston, 1975), qui représente les exemples et les concepts sous la forme de réseaux sémantiques.
2. Le programme AQVAL (Michalski, 1973), utilisé dans le cadre des problématiques de reconnaissance de patterns, et qui travaille à partir de l'expression de règles logiques sur des valeurs de variables.
3. Le programme ID3 (Quinlan, 1986), pour « *Iterative Dichotomiser 3* », génère des arbres de décision.

L'AS vise à élaborer des méthodes optimales dont l'objectif est d'extraire des connaissances structurelles ou décisionnelles à partir d'un ensemble d'information qui ne l'est pas. Deux exemples de mise en œuvre :

1. Le « *Data Mining* » dont l'objectif vise à rapprocher (au sens sémantique) des informations pouvant être dispersées dans un très grand volume de données.
2. La reconnaissance de manuscrits par rapport à un corpus de mots stockés.

Si, pour un programme informatique donné, on considère que :

1. **T** est la tâche à effectuer.
2. **E** est une expérience, i.e. une occurrence de l'exécution de **T** dans le temps.
3. **P** est la mesure de performance de l'exécution de **T** pour **E**.

On dira alors que le programme informatique apprend si **P** augmente au travers des différentes expériences réalisées. Juger l'accroissement de la valeur de **P** revient à optimiser la fonction **f** permettant de la calculer. Cette fonction est apprise à partir de l'ensemble des expériences, chacune d'elles pouvant être représentée par un couple **(b,f(b))**, où **b** représente l'ensemble des données de l'expérience et **f(b)** est la valeur de la fonction cible, donc celle de **P**, sur cet ensemble de données.

Les principales techniques d'AS sont les arbres de décision (Quinlan, 1994), la Programmation Logique Inductive (PLI) (Cornuéjols, 2002) et l'Inférence Grammaticale (IG) (De La Higuera, 2002). Ces 3 techniques sont abordées de manière très synthétique dans ce qui suit.

1.7.10.2 Apprentissage

1.7.10.2.1 Arbres de décision

Classiquement, un tel arbre est un outil d'aide à la décision dont l'objectif est de faire apparaître à l'extrémité de chaque branche (appelée feuille) les différents résultats possibles en fonction des décisions prises à chaque étape. L'utilisation de ce type d'arbres est avérée dans de nombreux domaines professionnels. Par exemple, en informatique, certains algorithmes produisent des arbres de décision dans le but de regrouper (classer) des clients dans des groupes communs. Ce classement s'effectue en fonction d'un ensemble de critères définis à l'avance (âge, sexe, etc.) et d'un objectif fixé (chiffres d'affaires, réponse à un mailing, …). En

utilisant de tels arbres, on cherche à prédire avec le plus de précision possible les valeurs prises par la variable à prédire (objectif, variable cible, valeur d'une variable de sortie, …) à partir d'un ensemble de descripteurs (variables prédictives, variables discriminantes, variables d'entrées, …).

Synthétiquement, l'apprentissage par arbre de décision vise à construire un modèle prédictif permettant d'évaluer une valeur de sortie à partir de l'observation des valeurs d'autres éléments du même système. Ici, les feuilles représentent les valeurs de la variable à atteindre. Les embranchements, quand à eux, décrivent les combinaisons de variables d'entrée qui mènent à ces valeurs. Si nous prenons l'exemple de la « fouille de données », il existe deux principaux types d'arbre de décision:

1. *Ceux de classification (Clustering Tree)* : prédiction de la classe d'appartenance de la variable-cible. Ici, la prédiction est une étiquette de classe.
2. *Ceux de régression (Regression Tree)* : prédiction d'une quantité de type nombre réel (e.g. le prix d'une automobile) où la prédiction est une valeur numérique.

1.7.10.2.2 Programmation Logique Inductive

La programmation logique inductive (PLI) est tout particulièrement utilisée dans le cadre du traitement des problématiques linguistiques. Elle s'appuie sur une base de faits et vise à générer des règles sous la forme de formules mathématiques issues de la logique des prédicats. A partir d'un ensemble d'exemples et de contre-exemples, la PLI permet de générer des expressions logiques qui établissent des liens entre des variables. Pour illustrer le fonctionnement d'un programme de PLI, considérons les trois faits suivants :

1. Jean est le père de Pierre.
2. Paul est le père de Jean.
3. Paul est le grand-père de Pierre.

Un programme de PLI doit être capable de trouver une formule du type « Pour tous les x et z tels que z est le grand-père de y, il existe x tel que x est le père de y et y est le père de z ».

En PLI, il s'agit finalement d'expliquer au mieux les observations positives, et à contrario de rejeter autant que possible les observations négatives. Deux caractéristiques fortes se dégagent pour la PLI : le formalisme d'expression des règles (logique des prédicats) est très bien connu, la notion de généralisation peut être introduite en cohérence avec les règles logiques générées.

1.7.10.2.3 Inférence Grammaticale

L'inférence grammaticale (IG) trouve ses applications dans les domaines de traitement du langage, la génétique, l'étude de la structuration des documents, etc. Elle consiste en l'apprentissage de grammaires à partir de séquences, d'exemples et de contre-exemples. Exemple : l'apprentissage de la grammaire d'une langue à partir d'un ensemble d'exemple contenant au moins un exemple positif (i.e. un sous-ensemble fini d'un langage) et un contre-

exemple (un ensemble fini de chaînes n'appartenant pas au langage). La spécification d'un problème d'inférence grammaticale repose sur la spécification de 5 points :

1. *La classe de grammaires*, comme par exemple celle des *grammaires régulières* selon la classification de Chomsky (Chomsky, 1956) ou l'une de ses sous-classes.
2. *L'espace d'hypothèses*, c'est-à-dire l'ensemble des descriptions dans lequel chaque grammaire de la classe considérée possède au moins une description. Exemples pour les grammaires régulières : espace des automates, des expressions régulières ou des ensembles réguliers.
3. *Un ensemble d'exemples*, pour chaque grammaire à inférer et un protocole de présentation de ces exemples.
4. *La classe des méthodes d'inférence* en considération. Il s'agit en particulier des méthodes constructives ou heuristiques.
5. *Les critères d'une inférence réussie*, tels que l'*identification à la limite* (Gold, 1967) par exemple.

Finalement, l'IG est un processus qui a pour objectif de lire des exemples (ayant une représentation admissible) et de proposer une(des) solution(s) prise(s) dans l'espace des hypothèses.

1.7.10.3 Algorithmes

L'objectif des algorithmes d'AS est de vérifier l'appartenance ou non d'un exemple donné à un concept déterminé, en s'appuyant sur un ensemble (espace des hypothèses) contenant des exemples que l'on sait positifs et/ou négatifs. Citons quelques exemples illustrations :

- **Find-S** (Mitchell, 1997) : trouve l'hypothèse la plus spécifique qui satisfasse tous les exemples positifs dans l'ensemble d'apprentissage.
- **Candidate-Elimination** (Genesereth, 1987) : génère un ensemble d'hypothèses plausibles, mais prises dans *l'espace de versions* qui est un sous-ensemble de celui des hypothèses.
- **ID 3** (Quinlan, 1986) : algorithme de classification supervisé, permettant de classer les nouveaux exemples dans un ensemble de classes. Il est basé sur un arbre de décision.
- **C4.5** (Quinlan, 1986) : amélioration d'ID3.
- **List-Then-Eliminate** : cité uniquement pour mémoire car relativement peu efficace. Il possède une limite importante puisque toutes les hypothèses doivent être connues au départ.

1.7.11 Apprentissage incrémental

1.7.11.1 Principes généraux

L'apprentissage incrémental consiste à apprendre en fonction des données d'entrée qui arrivent au fil du temps. C'est le cas, par exemple, des algorithmes d'apprentissage par renforcement que nous avons décrits au paragraphe 1.7.6. Les algorithmes mettant en œuvre l'apprentissage incrémental sont qualifiés de « *en ligne* » par opposition à d'autres qui sont qualifiés de « *hors ligne* ». Par opposition aux traitements de type « *hors lignes* » il n'y a pas, dans le cas qui nous préoccupe, de vision globale de la totalité des données qui vont être reçues avant de démarrer les traitements. Par exemple, il est possible de calculer directement une moyenne sur un ensemble de valeurs d'entrée dans le cadre d'un algorithme « *hors ligne* » puisque toutes les valeurs sont connues au démarrage au traitement. A l'opposé, dans le cas d'un algorithme « *en ligne* », on ne peut que calculer une moyenne incrémentale puisque les valeurs d'entrée sont présentées progressivement au système. Ainsi, l'utilisation de l'apprentissage incrémental implique d'accepter la prise de décision. Le risque assumé est alors d'aboutir à un résultat cible non optimal une fois toutes les données traitées. On notera également qu'il convient d'accepter aussi la transformation des données sous une forme résumée dans le but d'optimiser la consommation des ressources de stockage et de mémoire du calculateur.

La publication (Salperwyck, 2009) dresse un état de l'art de l'apprentissage incrémental. Elle nous montre qu'il convient de considérer le cas particulier de l'apprentissage incrémental « sur flux ». Ici, on considère des données d'entrée qui ne sont accessibles qu'au moment de leur passage, i.e. qu'elles n'ont pas de persistance. C'est le cas, par exemple, lorsqu'un utilisateur effectue une requête via son navigateur web et transmet au serveur un formulaire de saisie avec des paramètres dont il a spécifié les valeurs.

1.7.11.2 Apprentissage

On l'aura compris, l'objectif de l'apprentissage incrémental est d'apprendre au fil de l'eau, en présentant bien les nouvelles données d'entrée à l'algorithme, mais sans que celui-ci n'ai besoin de tout réapprendre en partant de zéro. Il s'agit d'optimiser la fonction de calcul qui permet de prédire le résultat en fonction de ces données d'entrée. Il n'y a pas non plus ici de stockage de l'intégralité des données mais plutôt la conservation d'une forme « résumée » de ces dernières. Ces mécanismes sont mis en œuvre au sein d'algorithmes dits de « *fouille de flots de données* » (Agrawal, 1992) qui sont concrètement employés dans des domaines nécessitant de manipuler de fortes volumétries de données. Exemples : la gestion de réseaux de télécommunications, la modélisation des utilisateurs dans un réseau social, le web mining...

1.7.11.3 Algorithmes

Quelques exemples illustratifs, synthétisés dans la publication (Salperwyck, 2009):

- **Arbres de décision** : voir détails au paragraphe précédent.
- **Séparateurs à Vastes Marges (SVM)** (Cortes, 1995) : utilisés pour travailler sur des données à grande dimension.
- **K Plus proches voisins** (Dickerson, 1992) : classements suite à calcul de distances par rapport à des cas similaires déjà résolus.
- **Systèmes à base de règles** (Jackson, 1998) : ce sont les systèmes experts.
- **Bayésien naïf** (Domingos, 1997) : basé sur le théorème de Bayes (Bayes, 1763). Il s'agit d'effectuer un classement du cas courant en supposant que chaque caractéristique d'une classe est indépendante des autres.

1.7.12 Apprentissage statistique

1.7.12.1 Principes généraux

De manière communément admise, on distingue la statistique décisionnelle (encore appelée inférentielle) de la statistique descriptive. Si la première utilise une base de données dans le but de prédire la valeur de variables non observées, la seconde se focalise sur la description des liens entre les différentes variables observées. Dans le cas de la statistique décisionnelle, on identifie deux types de modèles :

1. *Paramétrique* : la prédiction sera basée sur un nombre fini de paramètres, indépendamment de la taille de la base de données.
2. *Non paramétrique* : la prédiction utilise une base de données composée de n couples d'entrées/sortie, supposés identiquement distribués.

La notion d'apprentissage statistique (Vapnik, 1998) repose sur la branche non paramétrique de la statistique décisionnelle. Quelques exemples de cas d'utilisation : classement et analyse d'images, reconnaissance d'objets, classement de documents textuels, diagnostic médical, etc. On trouve généralement ce type d'apprentissage dans les domaines nécessitant de la fouille de données, type data mining.

Si l'on considère les hypothèses suivantes :

1. Nous disposons d'une base de données de n couples $Z_1 = (X_1, Y_1), \ldots, Z_n = (X , Y)$
2. Les couples Z_i sont des réalisations indépendantes d'une même loi P inconnue.
3. Les X_i appartiennent à un espace X (égal à $\mathbb{R}^d$ pour un grand entier d) et sont les entrées.
4. Les Y_i appartiennent à un espace Y fini (ou un sous-ensemble de $\mathbb{R}$) et sont les sorties.

Alors, le but de l'apprentissage statistique est de prédire Y associé à tout nouvel X, sachant que le couple (X, Y) est une nouvelle réalisation P qui reste indépendante des autres réalisations de P antérieurement observées.

1.7.12.2 Apprentissage

Le calcul de la valeur de Y à partir de X s'effectue via une *fonction de prédiction* notée F(X,Y). De même, il existe une fonction dite « *de perte* » et notée $\ell(y, y')$. Elle traduit le fait qu'il y ait un écart entre la sortie y réellement attendue et la sortie prédite y'. Selon les situations, on parle de :

1. *Classement* : $\ell(y, y') = 1_{y \neq y}$, i.e. $\ell(y, y') = 1$ si y différent de y', $\ell(y, y') = 0$ sinon.
2. *Régression (aux moindres carrés)* L_p : ici, $\ell(y, y') = |y - y'|^p$ où $p \geq 1$ est un réel fixé.

La qualité d'une fonction de prédiction $g : X \rightarrow Y$ est alors mesurée par son *risque*, que l'on appelle également *erreur de généralisation*. Voir équation 1.

$$R(g) = \mathrm{E}\left[\ell(Y, g(X))\right] \qquad (1)$$

Calculer le risque revient ainsi à valoriser l'espérance par rapport à loi P de la perte encourue, par la fonction de prédiction g, sur la donnée (X, Y).Ainsi, le rôle des algorithmes d'apprentissage consiste à déterminer la *meilleure* fonction de prédiction (fonction *cible* ou fonction *oracle*) qui minimise $R(g)$ le plus possible. Par exemple :

1. *Classement* : $\ell(y, y') = 1_{y \neq y}$. La fonction qui à une entrée x renvoie la sortie la plus probable (au sens de la distribution conditionnelle de Y sachant X = x : L(Y |X = x)) est "la" fonction cible en classement.
2. *Régression aux moindres carrés* : Y = R et $\ell(y, y') = |y - y'|2$. La fonction qui à une entrée x renvoie la sortie moyenne $E(Y |X = x)$ est "la" fonction cible en régression aux moindres carrés.

1.7.12.3 Algorithmes

Quelques exemples illustratifs:

1. **Algorithmes par moyennage local** : ces algorithmes visent à estimer $E(Y |X = x)$, en moyennant les Y_i des X_i proches de x. On citera les algorithmes par partition, par noyau, et k-plus proches voisins (k-p.p.v.) (Dickerson, 1992).
2. **Algorithmes par minimisation du risque empirique** : réseaux de neurones, Séparateurs à Vastes Marges (SVM) (Cortes, 1995).

CHAPITRE 2 – Modèle théorique des agents compétence

« Peu de choses sont impossibles à qui est assidu et compétent... Les grandes œuvres jaillissent non de la force mais de la persévérance »

Samuel Johnson

2.1 Rappel synthétique[3] de notre problématique

Pour bien comprendre le problème, reprenons ici la description du besoin que nous avons décrit au paragraphe « *1.2.1 Le cas CG33 : spécifications du besoin* ». Au CG33, la mise en œuvre de nombreux projets de Développement Durable (sensibilisation autour des nouveaux moyens de production d'énergie, construction de bâtiments HQE, transformation d'un quartier en « éco-quartier » (Bardou, 2011), etc.). Les retours d'expérience ont montré toute l'évolution de leur complexité. De plus, ils requièrent la participation de nombreux acteurs qui, souvent, ne disposent que d'une vision partielle des capacités globales des autres participants (CG33-9, 2011). L'un des points clé à retenir est que la réussite des projets est finalement subordonnée à l'exercice d'un ensemble de compétences spécifiques et interdépendantes, mais que cet ensemble ne saurait être disponible chez un acteur unique. Compte tenu de cette situation, le CG33 a souhaité mettre en place un système d'information accessible en ligne. Destiné à assurer la réussite des projets de DD en Gironde, les principaux objectifs à atteindre sont les suivants :

- Rechercher, partager, puis sélectionner des compétences ainsi que les acteurs capables de les concrétiser.
- Renseigner chacun des acteurs sur l'expertise et le savoir-faire des partenaires potentiels.
- Assurer une montée en compétence des acteurs au travers des échanges (réciprocité). La notion de partage (échanges fructueux et réciproques) est alors fondamentale et placée au cœur des préoccupations.
- Mobiliser autour de la mise en œuvre des politiques de DD.
- Créer un système informatisé « *vivant* » qui s'enrichisse des expériences acquises et partagées par chacun.

[3] Cf. paragraphe « *1.2 Problématique* » au chapitre 1 pour tous les détails

On notera également que l'un des points fondamentaux de la demande exprimée est de réaliser un système qui ne doit pas être statique. L'identification des bons interlocuteurs doit évoluer en fonction de la pertinence, de l'efficacité et de l'efficience des partages d'expérience ainsi que des résultats concrets obtenus dans le déroulement du projet.

En généralisant à partir du cas CG33, nous pouvons dire que quel que soit le secteur d'activité professionnel concerné, l'identification et le partage optimal de compétences est au cœur de la réussite des *projets* qui concrétisent des *objectifs* dans le « monde réel ». Ainsi, via la solution informatisée que nous apporterons, tout maître d'œuvre doit être capable pour un projet :

1. De le définir en identifiant les compétences qui le composent.
2. D'identifier les acteurs capables d'exercer les compétences conduisant à sa réussite.
3. D'en évaluer, à postériori, la réussite.

Plus généralement, le 3ème point traduit le désir de faire émerger une certaine « Intelligence Collective » (Dortier, 2006), basée sur une promotion forte des aspects de collaboration, de partage et de diffusion (Boutte, 2009).

2.2 Analyse & proposition de réponse

La réalité des projets, quel que soit le secteur d'activité professionnel, nous pousse à intégrer un certain dynamisme au sein de notre solution. En effet, les compétences utilisées au sein des projets évoluent, peuvent se transformer par les progrès techniques, et peuvent disparaitre (si remplacées ou finalement non utilisées). Elles doivent répondre aux préoccupations émises par les utilisateurs au moment où ils expriment leur besoin. Un exemple : je suis un Girondin ayant pour but de réaliser des économies d'énergie en utilisant des technologies durable au sein de mon habitation. A ce stade de ma réflexion, mes questions sont les suivantes :

1. Que dois-je faire comme travaux (isolation, etc.) ?
2. Qui peut m'aider à les réaliser (quelles entreprises) ?

Une réponse dite « classique » pourrait consister à réaliser simplement une base de données où les compétences et acteurs seraient des objets statiques. Cependant, ceci ne répond pas à la nécessité de dynamisme et d'évolution des compétences. L'analyse de la problématique nous conduit à définir la *Compétence* comme l'élément fondamental du système à construire. Compte tenu de ces éléments, l'étude R&D que nous avons mené dans le cadre de cette thèse nous a conduit à proposer plutôt des « agents compétence » évoluant au sein d'une architecture de type multi-agents (Ferber, 1999) (Russel, 2003). Si l'on raisonne de manière classique, (de type logiciel de gestion de projets par exemple), le système cible est conçu de manière « centrée sur les acteurs ». Dans ce cas, chacun d'eux possède une liste de compétences. A l'opposé, nous proposons une approche où la conception est centrée sur les compétences. On considère ainsi que chacune d'elles est mise en œuvre par une liste d'acteurs. Cette approche où un agent logiciel modélise une compétence est novatrice puisque :

1. Elle oriente réellement la mise en œuvre des projets vers la nécessité d'obtenir l'exercice d'une « compétence » et non plus vers la nécessité de « disposer » de tel ou tel acteur humain pour exercer la compétence. De manière illustrée, nous demanderons simplement au système une compétence en plomberie. Nous n'appellerons plus directement M. X parce que nous savons que c'est un bon plombier. Les modalités de raisonnement deviennent différentes, nous nous recentrons sur le besoin et non plus sur sa réalisation. Il convient de penser « compétence » et non plus « acteur humain qui la rend ».
2. Elle confère aux compétences toutes les caractéristiques induites par une modélisation sous la forme d'agents logiciels. On pourra citer, entre autres : capacités de communication inter-agents, perception de l'environnement dans le SMA, mise en œuvre de comportements, apprentissage.
3. Le concept même de compétence, par sa modélisation sous la forme d'une entité logicielle, devient une entité « *vivante et autonome* » au sein d'un SMA. Une compétence possède un cycle de vie qui lui est propre, ainsi qu'un modèle comportemental personnalisé et évolutif, pouvant être appris. Nous pouvons alors parler d' « ***Agent Compétence*** » (AC).
4. Elle est cohérente avec les concepts de « *l'entreprise apprenante* » (Mallet, 2007) et du « *Knowledge Management* » (Prax, 2000), et par là même, généralisable quel que soit le secteur d'activité professionnel. L'organisation, l'utilisation, et la transmission (acquisition) des compétences par les différents acteurs humains concourent à l'atteinte des objectifs d'émergence de l'« *Intelligence Collective* » (Dortier, 2006). Boutte nous explique que: « *Les contributions des différents auteurs éclairent sous plusieurs angles notre problématique (… dont) celui de la* ***modélisation de la compétence*** *à l'ère de la complexité.* » (Boutte, 2009). La définition de notre concept d'AC s'inscrit alors pleinement dans cette citation.
5. Elle prend en compte la granularité des compétences à travers différents niveaux hiérarchiques comme déjà mentionné au chapitre 1.

Reprenons notre illustration basée sur le cas CG33. La réponse souhaitée par cette institution territoriale prend la forme de l'outil informatique BCDD dans lequel sont exploités des "Bourses de Compétences". Une représentation schématique et simplifiée des interactions entre acteurs humains et ce type de systèmes est donnée. Voir figure 9.

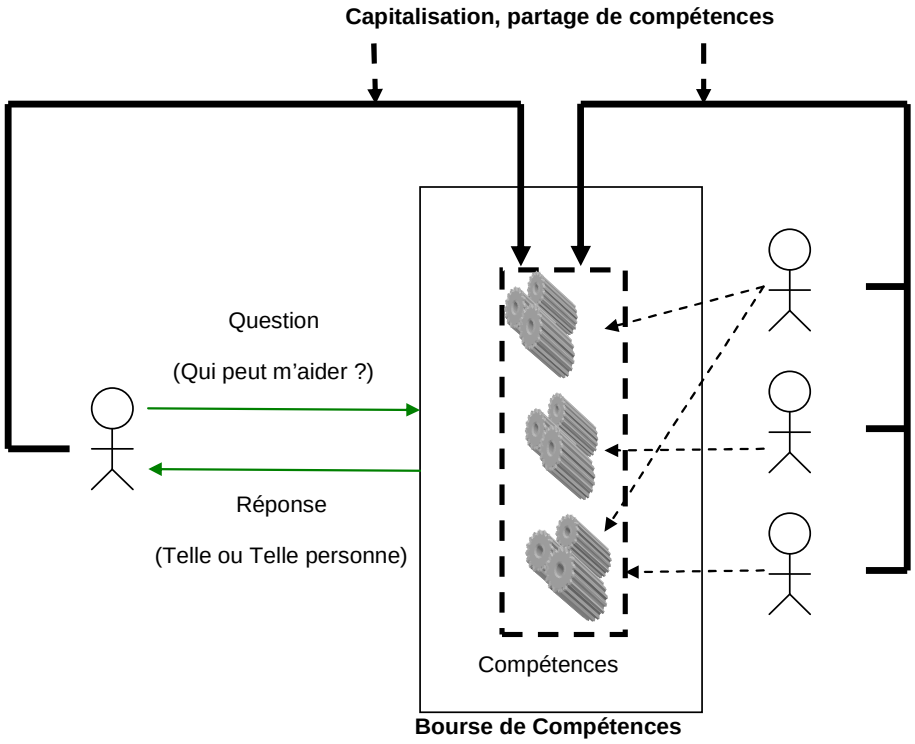


Figure 9 Les principes de la Bourse aux Compétences

Un tel système sera interrogé par un acteur humain qui souhaite bénéficier de compétences partagées au sein de la bourse. Les acteurs humains partagent bien des compétences pour aider à la mise en œuvre des projets. La bourse s'enrichit des expériences acquises et partagées par chacun d'eux. Le système est dynamique. Pour un projet donné, l'identification des acteurs exerçant une compétence évolue en fonction du contexte. Synthétiquement, ce n'est pas toujours le « meilleur acteur » (au sens niveau d'expérience) qui sera désigné par le système. Il est important de noter également que nous proposons de concevoir ce système de Bourse de Compétence comme un outil transversal. Il met en œuvre une certaine interdépendance des personnes impliquées au service d'un objectif commun à atteindre. La vision d'une transversalité des compétences (et non plus une vision « en silos ») pour atteindre les objectifs fixés, au travers des projets, revêt ici tout son sens. Une compétence peut n'être « incorporée » par personne (si c'est une compétence que l'on souhaite acquérir ultérieurement après formation professionnelle par exemple) ou être partagée par plusieurs acteurs.

2.3 Fondements théoriques

Comme nous l'avons vu dans le chapitre consacré à l'état de l'art[4], dans un Système multi-agents (Ferber, 1999) (Jennings, 1998), un Agent possède des compétences qui sont mises en œuvre par des comportements. Dans leur publication « *Des modèles hétérogènes de simulation par systèmes multi-agents* » (Gutknechtl, 1998), Gutknecht, Ferber, et Lieurain proposent le modèle AALAADIN qui possède 3 notions de base (agent, groupe rôle). Concernant la notion d'agent, citons les auteurs : « *Nous ne posons aucune contrainte ou pré-requis sur l'architecture interne de l'agent et nous ne supposons pas de formalisme ou modèle particulier pour en décrire le comportement. L'agent est simplement décrit comme une entité autonome communicante qui joue des rôles au sein de différents groupes. Le caractère très générique de cette définition est volontaire : le constructeur de système appliquera à cette notion d'agent l'architecture d'action ou de communication idoine* ».

Nous nous appuyons sur cette définition d'un agent. Dans le cadre de cette thèse, tout le problème consiste alors à modéliser un système multi-agents cohérent dans lequel les agents **SONT** des compétences.

2.4 Définitions

2.4.1 Environnement ou contexte

Tout projet s'insère dans un » environnement ou contexte qui fait sens dans un secteur d'activité professionnel. Exemples :

1. Dans le secteur des problématiques du DD : contexte de l'Agenda 21 de la Gironde, contexte d'un concours international d'architecture, contexte du programme d'aménagement des zones prioritaires d'éducation porté par le Ministère de l'Education Nationale.
2. Dans le secteur sportif du rugby, on peut considérer que les matchs sont joués dans les contextes suivants : tournoi des 6 nations, H-Cup (coupe d'Europe), top14 (championnat Français), etc.

2.4.2 Objectif et projet

Un objectif traduit généralement le résultat que l'on cherche à atteindre suite à la mise en œuvre d'actions pour y parvenir. En Programmation neuro-linguistique (Maders, 2000), un objectif se doit d'être SMART, c'est-à-dire :

1. Simple / Spécifique : formulé de manière claire et non ambigüe.
2. Mesurable : possédant des indicateurs factuels permettant de pouvoir juger de son atteinte.

[4] Cf. paragraphe « *1.5.2 Paradigme Agent* » au chapitre 1

3. Ambitieux : motivant pour chacun.
4. Réaliste : atteignable avec les moyens dont on dispose.
5. Temporel : borné dans le temps (dates de début et de fin, durée, etc.).

Dans nos travaux, nous définissons LO en tant que liste complète des objectifs unitaires O_k tels que $O_k \in$ LO. Concrètement, sur le terrain, chaque fois qu'un objectif unitaire O_k doit être réalisé, il y a mise en œuvre d'un projet P_k. Informatiquement parlant, au sens programmation objet du terme, nous proposons de définir une classe **générique** O dont les instances sont des projets P_k visant à atteindre un objectif O_k. Par exemple, en reprenant nos illustrations du paragraphe précédent, un objectif O_k peut être la construction d'un collège HQE (secteur d'activité du bâtiment) ou bien la sélection de joueurs participants à un match de rugby (secteur sportif, jeu de rugby). Un projet est réalisé concrètement dans le contexte d'un environnement donné. Chaque projet P_k est composé de 1 (à minima) à n compétences et sa mise en œuvre produit des résultats concrets et mesurables : un bâtiment construit, un match de sport à disputer, une prestation de service réalisée, etc.

Un Projet P_k (instance de la classe générique O) est ainsi :

1. Borné dans le temps, possède un délai de réalisation, une date de début et une date de fin. Chacune de ces 3 notions (délais et dates) peut être variable.
2. Défini par une liste **de compétences** C_i qui concourent à sa réalisation.

Il est évalué à la fin de sa réalisation, ou à l'issue de certaines étapes. Son évaluation globale est définie par la somme des évaluations unitaires E_m **de chaque mise en œuvre de compétence C_i composant le projet (voir sections 2.4.4 et suivantes)**.

Chaque projet P_k fait concrètement appel à la mise en œuvre de chaque compétence nécessaire pour sa réalisation dans le monde réel. Un projet peut être ainsi vu comme une "incorporation" ou une "incarnation" au sens où Varela (Varela, 1991) les a définis. Voici deux exemples illustratifs :

Exemple 1. Secteur d'activité du bâtiment.
Projet : construire le collège de X. Le système :

1. Identifie les compétences requises : maîtrise d'œuvre, architecture, etc.
2. Identifie les acteurs humains pouvant exercer chacune des compétences.
3. Met en place le système d'évaluation de la construction.

Exemple 2. Secteur d'activité sportif, jeu de rugby.
Projet : gagner le match du 03 Juin 2014 contre l'ASM Clermont Auvergne (instance de la classe d'objectif « gagner le match contre l'ASM Clermont Auvergne »).
Si l'on assimile les postes des joueurs à des compétences, le système :

1. Identifie les compétences requises : 15 postes de joueurs.
2. Identifie les 15 acteurs humains pouvant exercer chacune des compétences identifiées.
3. Met en place le système d'évaluation du projet courant, par exemple à postériori et à la fin du match.

2.4.3 Domaine d'objectif

Un domaine d'objectif est pour les objectifs ce qu'un domaine de compétences est pour ces dernières. Il est défini par le regroupement fonctionnel d'un ou plusieurs objectifs relatifs à une même thématique. Un objectif appartient à 1 ou n domaines d'objectifs. Voici deux exemples illustratifs :

Exemple 1. Dans le cadre de notre problématique liée au DD, le CG33 possède des domaines d'objectifs tels qu' « Urbanisme et aménagement » et « Services aux populations et dynamiques locales ». Les différents objectifs sont classés selon ces domaines :

1. Les objectifs « Revitaliser les centres bourgs » et « Améliorer les usages du bâti» appartiennent au domaine « Urbanisme et aménagement ».
2. Les objectifs « Soutenir l'innovation des associations » et « Déployer services numériques aux usagers » appartiennent au domaine « Services aux populations et dynamiques locales ».

Exemple 2. Si l'on considère le secteur sportif et le jeu de rugby, introduisons les domaines d'objectif « Entrainements » et « Matchs réels ». On propose alors que :

1. Les objectifs « Travailler les pénalités » et « Travailler les mêlées » appartiennent au domaine « Entrainements ».
2. Les objectifs « Préparer déplacement à l'extérieur » et « Gagner un match » appartiennent au domaine « Matchs réels ».

2.4.4 Critère d'efficacité et compétence élémentaire

2.4.4.1 Problématique

Nous avons présenté au chapitre 1 la définition d'une compétence élémentaire[5]. On rappellera ainsi que selon Perrenoud repris par Roux (Roux, 2011), « *une compétence peut fonctionner elle-même comme une ressource, mobilisable par des compétences plus larges* ». Considérons que :

- **P** est un projet.
- **C** est l'ensemble de toutes les compétences référencées dans le système. Remarque : Les compétences sont identifiées au niveau métier. Elles doivent être créées manuellement dans le système par l'administrateur. Elles sont ensuite toutes instanciées dans le SMA sous la forme d'un agent compétence et disponibles dans les listes de compétence présentées à l'utilisateur dans les différentes interfaces.
- **CE** est l'ensemble de toutes les compétences élémentaires $\mathbf{C_e}$ et tel que **CE** $\not\subset$ **C. Une compétence élémentaire n'est généralement pas une compétence, mais nous ne**

[5] Cf. paragraphe « *1.2.6 Compétence élémentaire* » au chapitre 1

l'excluons pas. De plus, si $\mathbf{C}_i$ est une compétence, alors $\mathbf{C_i}$ sera qualifiée de « haut niveau » si elle est composée de compétences non élémentaires.

L'un des problèmes identifié est le niveau de granularité du découpage des compétences $\mathbf{C_i}$ en compétences élémentaires $\mathbf{C_e}$. Dans certains secteurs d'activité, ce niveau de granularité est connu et fait d'ailleurs l'objet de descriptions formalisées, au moins pour ce qui concerne les savoirs explicites (Schlick, 2009) (Gladun, 2007) (Le Blanc, 2007). C'est le cas, par exemple, pour des processus de production industriels. A l'opposé, ce n'est pas forcément vrai lorsque, à l'instant t, un utilisateur interroge le système pour la réalisation d'un projet **P**. Si l'utilisateur est généralement plus ou moins capable d'identifier les compétences $\mathbf{C_i}$ de « haut niveau » requises pour réaliser P, il n'est pas forcément capable de définir précisément toutes les compétences élémentaires $\mathbf{C_e}$ qui composent les compétences $\mathbf{C_i}$.

Cette difficulté nous amène à discuter d'un problème complémentaire qui est celui de l'évaluation d'un projet une fois qu'il a été réalisé. Pour y répondre, nous proposons de manière relativement classique d'évaluer (à postériori) la mise en œuvre de chaque compétence $\mathbf{C_i}$ dans un projet **P** grâce à des critères notés $\mathbf{C_e}$. Pris de manière unitaire, chacun des $\mathbf{C_e}$:

1. Possède un poids $\mathbf{PC_e}$ (de valeur 1 par défaut) qui traduit le niveau d'importance qu'on accorde au critère $\mathbf{C_e}$ par rapport aux autres critères retenus dans **CE**.
2. Participe en fonction de son poids $\mathbf{PC_e}$ à l'évaluation globale de l'instance de compétence $\mathbf{C_i}$ pour le projet P.

2.4.4.2 Equivalence entre critère d'évaluation et compétence élémentaire

Dans notre système, nous avons proposé d'évaluer concrètement l'exercice d'une compétence $\mathbf{C_i}$ mise en œuvre dans un projet sur la base de critères d'efficacité. Prenons un exemple dans le jeu de rugby en considérant le poste (la compétence) de demi de mêlée. Les critères de jugement de cette compétence sont typiquement :

- Pourcentage de passes réussies.
- Pourcentage de plaquages réussis.
- Nombre de ballons pris à l'adversaire.
- Nombre de ballons touchés.
- Capacité de pénétration (nombre de mètres avancés avec le ballon en mains).
- Vitesse pure (pour les trois-quarts et arrières).
- Nombre de kilomètres parcourus rapporté au temps de jeu.
- Pourcentage de réussite de pénalités et drops.

A l'issu d'un match, l'entraineur de l'ASM Clermont Auvergne, par exemple, procède à l'évaluation de l'efficacité d'exercice de la compétence (du poste) en attribuant des notes (sur 4 points par exemple) à chaque critère. Or, précisément, si une compétence peut être évaluée à

partir d'une liste de critères, pourquoi ne pas définir les compétences élémentaires comme étant équivalentes à ces critères ? C'est l'idée que nous avons retenue.

Formulons l'hypothèse d'assimiler chaque compétence élémentaire C_e à un critère d'efficacité de jugement d'une compétence C_i dans un projet P.

Au paragraphe « *1.2.6 Compétence élémentaire* », nous avons vu qu'il n'existe pas de méthode standardisée et didactique de décomposition des $\mathbf{C_i}$ en $\mathbf{C_e}$ compte tenu des spécificités de chaque secteur d'activité métier. Notre hypothèse de travail permet d'apporter une réponse, bien que partielle, à cette problématique puisque :

1. Définir des critères d'efficacité de jugement pour chaque $\mathbf{C_i}$ revient à définir de manière implicite une liste de **Ce**.
2. Définir des critères d'efficacité garantit une adéquation maximale avec le contexte métier dans lequel le système est utilisé. Quel que soit le domaine fonctionnel, toute évaluation repose concrètement sur des éléments (ici nos critères) qui « font sens », sont significatifs, utiles, représentatifs, interprétables et exploitables dans le contexte métier où ils sont déterminés.

Nous identifions cependant des limites à notre proposition parmi lesquelles :

1. **Un risque potentiel de non exhaustivité de l'identification des C_e**. Rien ne garantit finalement que l'ensemble des critères d'efficacité de jugement représente l'exhaustivité des **Ce** définissant une $\mathbf{C_i}$. En reprenant notre illustration, la compétence de demi de mêlée peut nécessiter une compétence élémentaire « *savoir définir la meilleure option de jeu suite à une mêlée* » qui n'est pas incluse dans la liste des critères de jugements présentée. Nous pouvons toutefois dire que, à minima, que la liste des critères de jugement est incluse dans (ou potentiellement égale à) la liste exhaustive des $\mathbf{C_e}$ définissant une $\mathbf{C_i}$.
2. **Un risque de non atteinte du niveau optimal de granularité de la décomposition en C_e**. Les exigences du domaine métier président au niveau de granularité de la décomposition. Reprenons notre exemple sur la compétence de demi de mêlée dans le jeu de rugby :
   - L'entraineur de l'ensemble de l'équipe ASM Clermont Auvergne nécessite une évaluation des 8 critères proposés pour ses prises de décision stratégiques lors de la composition de son équipe pour chaque match.
   - L'entraineur dédié spécifiquement aux lignes arrières, quand à lui, peut ne pas être intéressé par certains critères tels que « *pourcentage de plaquages réussis* » ou bien encore « *nombre de kilomètres parcourus rapporté au temps de jeu* ». Cependant, il aurait pu souhaiter un niveau de granularité différent pour le critère « *pourcentage de réussite de pénalités et drops* ». En rugby, ces deux manières de marquer les points (pénalités d'un côté, drops de l'autre) nécessitent d'exercer des gestes techniques différents. Notre entraineur aurait pu souhaiter une décomposition (ou un niveau de granularité) différent afin de pouvoir cibler son entrainement du joueur sur le point le plus faible de ce dernier.
   - L'entraineur spécifique des lignes avants, de son côté, aurait pu désirer encore une autre décomposition pour entrainer le joueur sur les aspects du jeu dont il a la charge.

Malgré les limites que nous venons de voir, en considérant les exemples fournis, nous pouvons dire que les $\mathbf{C_e}$ déduites de la liste des critères sont bien les plus « essentielles », les plus représentatives, ou du moins les plus importantes dans la définition d'une $\mathbf{C_i}$**.** Notre proposition fait sens, même si elle peut être considéré comme partielle en termes de réponse vis-à-vis des niveaux d'exhaustivité et de granularité. Toutefois, sur le terrain, on constate que les processus de production industriels, les projets et les actions attendues dans l'exercice d'une compétence (d'un poste) évoluent. Rien ne semble donc réellement « figé », ce qui place notre proposition en adéquation forte avec la réalité observée. Que ce soit dans notre système ou sur le terrain, la liste des $\mathbf{C_e}$ composant une $\mathbf{C_i}$ vari et accompagne les évolutions induites par le contexte métier et de la réalité des projets.

Ainsi, pour un projet P, nous restons cohérents lorsque nous cherchons à déterminer les poids $\mathbf{PC_e}$ associés aux critères d'efficacité $\mathbf{C_e}$ qui définissent la compétence $\mathbf{C_i}$. On notera qu'un critère $\mathbf{C_e}$ peut être (ou non) évalué de manière systématique lorsque la compétence $\mathbf{C_i}$ est mise en œuvre dans **P**. Il n'est alors pas absurde de penser que certains critères $\mathbf{C_e}$ ne sont pertinents que dans le cadre de certains **P** donnés et bien identifiés. Dans ce cas, le critère Ce devient « spécifique » à **P**.

Finalement, nous définirons l'évaluation globale (cf. équation 3) d'un projet P, démarré à un instant t, comme étant la somme de chaque évaluation unitaire $\mathbf{E_m}$ (cf. équation 2) d'instances de compétences élémentaires $\mathbf{C_e}$ (en prenant en compte chacun de leur poids **PCe**), pour les instances (incarnations) de chaque compétence $\mathbf{C_i}$ mises en œuvre dans ledit projet. On parle bien, dans le contexte de **P**, d'instances de compétences et de compétences élémentaires, et non des compétences et compétences élémentaires elles-mêmes.

$$E_m = \frac{PC_e \times valeur(C_e)}{\sum_{e=1}^{n'} PC_e} \qquad (2)$$

et

$$Evaluation(P) = \sum_{m=1}^{n'} E_m \qquad (3)$$

Prenons deux exemples illustratifs :

Exemple 1. Secteur d'activité du bâtiment.
Pour juger de l'efficacité de mise en œuvre de la compétence « poser des panneaux photovoltaïques » nous proposons les critères d'efficacité suivants :

1. Nombre de panneaux posés : ce critère induit une compétence élémentaire. Pour le réaliser, il est nécessaire de « savoir poser des panneaux ».

2. Délais de réalisation, c’est à dire durée du chantier de pose. Ce critère est en lien avec l’acteur humain mettant en œuvre la compétence.

Exemple 2. Secteur d’activité sportif, jeu de rugby.
Si l’on assimile les postes des joueurs à des compétences, supposons que pour juger de l’efficacité de mise en œuvre de la compétence de demi de mêlée, nous proposions d’évaluer les critères d’efficacité suivants :

1. Nombre de pénalités réussies, nombre de transformations réussies et nombre de drops réussis. Ces 3 critères peuvent être vus comme autant de compétences élémentaires. Pour les réaliser, il est nécessaire de posséder un savoir faire de base : « savoir botter » (placer le ballon entre les perches suite à un coup de pied).
2. Nombre de matchs joués. Ce critère est en lien direct avec l’acteur humain mettant en œuvre la compétence.

Dans les deux exemples, la liste des critères proposés est bien entendu non exhaustive et purement subjective. Dans l’exemple de pose des panneaux photovoltaïques, intuitivement, on peut penser que le critère « délai de réalisation » n’est pas catégorisable, à priori, parmi les compétences élémentaires. Il semble relever simplement d’un constat, à savoir le délai de réalisation de la pose par l’acteur humain. Mais il est toujours possible d’assimiler ce critère à une compétence élémentaire sous-jacente qui serait de « savoir travailler dans les délais ». Le constat est identique pour l’exemple du jeu de rugby. On peut penser que le critère « nombre de matchs joués » n’est pas catégorisable dans les compétences élémentaires. Mais il est toujours possible d’assimiler ce critère à la notion « d’expérience », c’est-à-dire à une compétence élémentaire sous-jacente qui serait l’« acquisition de comportements appropriés en situation difficile non apprise à l'entrainement ». En conséquence, même dans le cas où les critères d’efficacité ne semblent pas vraiment être de prime abord des compétences élémentaires, ils en sont tout de même, d'une manière ou d'une autre, une abstraction.

2.5 Modèle agent proposé

2.5.1 Introduction

En informatique, on distingue le hardware du software. En cohérence avec notre problématique, il nous semble pertinent d’envisager une même dissociation au niveau humain. Nous pouvons alors considérer qu'une compétence :

- Est unique, bien qu’incarnée (au sens de Varela (Varela, 1991)) par plusieurs acteurs humains.
- Qu'elle possède une « vie propre ».
- Que ses moyens d'actions sont multiples et réalisés à travers un corps délocalisé (l’acteur humain) ou plutôt multi-localisés (répartie sur plusieurs acteurs physiques).

Prenons l’exemple d’un agent compétence (AC) « *pose de panneaux photovoltaïques* ». Plusieurs acteurs humains peuvent posséder cette compétence, mais celle-ci sera toujours

représentée sous la forme d'un agent unique dans notre SMA. Chaque AC n'a qu'une vision restreinte de l'environnement simulé par le SMA. Dans notre implémentation, les données descriptives de cet univers sont concrètement stockées dans une base de données. D'un point de vue théorique et modélisation, un AC n'a par exemple :

- Pas accès aux « souvenirs », c'est-à-dire concrètement aux « tables mémoire », des autres ACs du système.
- Pas de liste exhaustive des autres ACs présents dans le SMA.
- Que la capacité d'analyser et de travailler à partir des données de situation environnementale qui lui sont propres. Exemples : les projets auxquels il a participé, les compétences élémentaires qui le composent, les liens qu'il a établi avec les autres ACs, etc.

2.5.2 Proposition

En application avec la demande formulée au CG33, nous devons garantir la maitrise et la cohérence globale entre liste des ACs évoluant au sein du système et domaine professionnel. Il existe en effet un référentiel national pour l'évaluation des projets d'aménagement durable (CG33-7, 2012), dans lequel une liste de compétences bien définie est fournie. Dans ce contexte, les ACs ne peuvent être créés que par un administrateur. Bien entendu, cette manière de procéder pourra évoluer dans des versions ultérieures de l'outil. Il conviendra alors d'adapter ce dernier afin qu'il intègre des processus de proposition / validation de création de nouveaux ACs dans le système. La demande initiale consiste également à concevoir un système dynamique, évolutif et apprenant. Dans notre cas concret, au CG33, les ACs participent à la définition de projets de DD que souhaitent mettre en œuvre les utilisateurs. L'analyse de la problématique ne révèle pas, à ce jour, de nécessité de mise en œuvre de niveaux complexes de dialogue inter-agent. Il s'agit plutôt de se focaliser sur l'optimisation de l'identification des compétences à mettre en œuvre dans les projets de DD. En cohérence avec la théorie générale des SMA, il devient alors approprié de définir un AC comme étant de type cognitif, non-conversationnel, et non dialogique (Ferber, 1999) (Weiss, 1999). Nous proposons finalement qu'un AC :

1. Soit défini principalement par 3 composantes clé, chacune d'elle étant abordée en détail dans le paragraphe » « *2.5.4 Les 3 composantes clé* » plus bas :
    - Des mécanismes de perception de son environnement et de communication avec les autres ACs.
    - Des attributs internes dont une liste de compétences élémentaires.
    - Des mécanismes d'action décrits au sein d'un « modèle comportemental » dédié et évolutif par apprentissage.
2. Possède des ressources : il est réalisé dans le concret par un(des) acteur(s) humain(s) et/ou fait appel à d'autres ACs.
3. Possède à minima 4 buts :
    a. Constituer sa propre définition.

b. Restituer une liste d'acteurs humains capables de le concrétiser au sein d'un projet.
c. Etablir des liens (relations) avec d'autres agents.
d. Se porter candidat à une participation à de nouveaux projets.

4. Possède un « cycle de vie » décliné en 3 « *âges* » et qui sont : *enfance*, *adolescence* et *maturité*. Chaque âge traduit un niveau d'autonomie différent.
5. S'insère dans un SMA de type « évènementiel » plutôt que « temporel ».

Concernant ce dernier point, dans un SMA classique, les agents évoluent dans un univers où le temps est simulé comme dans la vie réelle. Dans notre cas, le principe de fonctionnement est différent. Ce sont essentiellement les interrogations formulées par les utilisateurs, les mises à jour du système ou bien l'évolution des situations environnementales qui font évoluer l'univers des ACs et par voie de conséquence les ACs eux-mêmes. Nous pouvons donc dire, de manière globale, que notre SMA est de type « évènementiel ».

2.5.3 Modèle de données

Les Agents Compétence stockent concrètement les données descriptives qu'ils manipulent et qu'ils créent dans une base de données. Dans ce paragraphe, le schéma de la figure 10 expose une vue des tables les plus importantes.

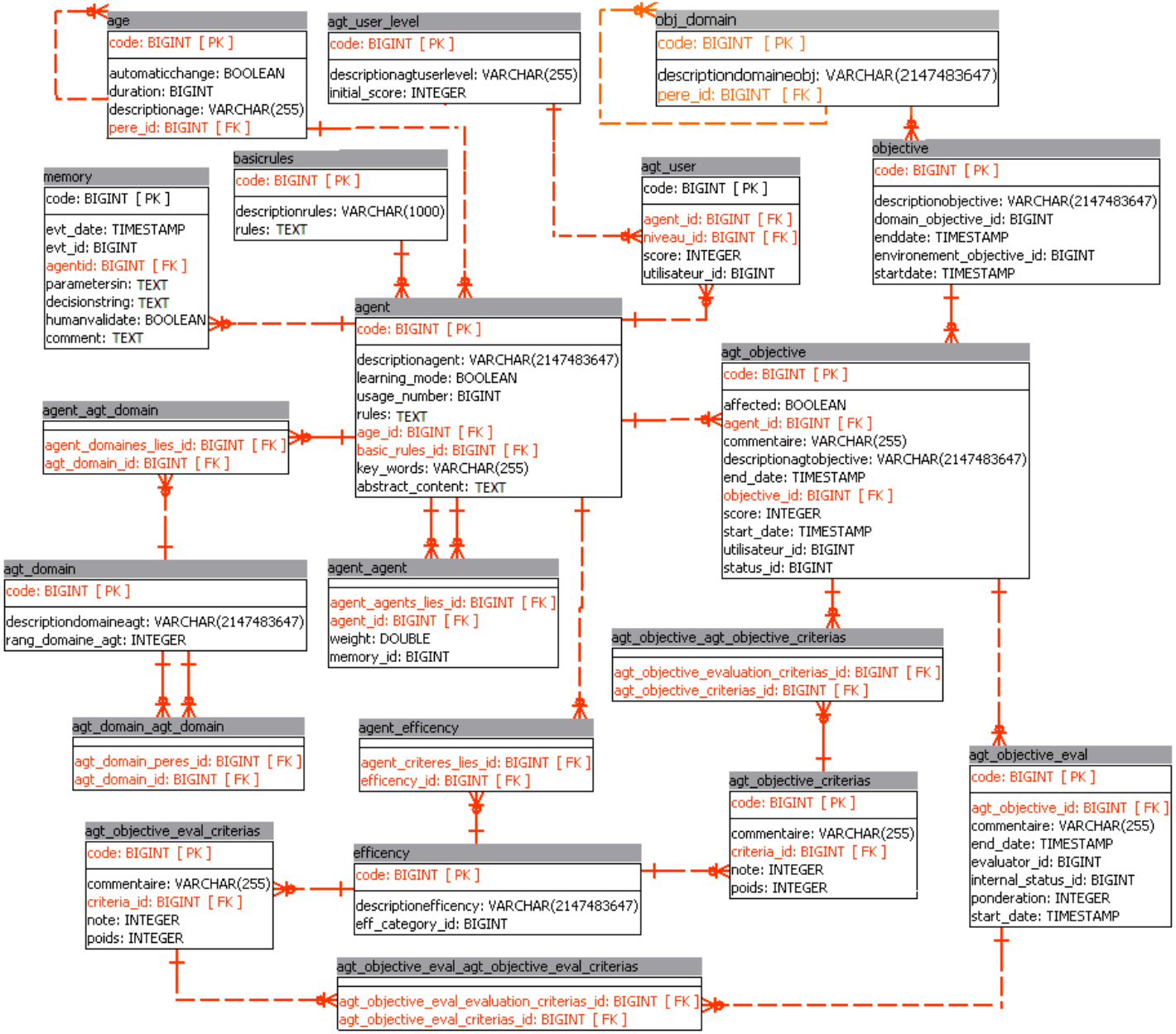


Figure 10 Vue partielle des tables du modèle de données

Présenter le modèle avec l'intégralité des tables qui le composent est trop fastidieux. Nous limitons notre description aux tables principales, en lien direct avec la modélisation des ACs. La table « *agent* » est au cœur du système et contient un enregistrement **Enr$_i$** par AC. Soit **Ci** la valeur du champ « *code* » identifiant un AC dans chaque **Enr$_i$**..

Champ	Description	Exemple de contenu
code	Identifiant unique	1556
descriptionagent	Description textuelle	Accompagner chefs de projets / M.O.
learning_mode	Mode apprentissage	TRUE
usage_number	Nombre de sollicitations cumulées dans tous les projets	10
rules	Règles comportementales	<?xml version="1.0" encoding="UTF-8"?>……</xml>
age_id	Identifiant de l'âge courant	1
basic_rules_id	Identifiant des règles de base	541
key_words	Mots clé	Projet, Maîtrise d'œuvre, Pilotage
abstract_content	Description détaillée	L'accompagnement des chefs de projet permet d'aider les chefs de projet à : piloter leurs projets, gérer les risques (Identification et mise en place de plans d'actions), mettre en place des tableaux de bord et les rapports, améliorer la communication au sein des projets, améliorer la gestion de la documentation projets

Tableau 2 : Illustration du contenu d'un enregistrement de la table des agents

Parmi les tables liées à *agent* on trouve celle nommée *age* qui décrit les étapes du cycle de vie des ACs. La table *basicrules* contient les règles éducatives de base comme par exemple les règles de croissance entre les différents âges des ACs. Autre table importante : *memory*. En fait, il existe dans le modèle 1 table *memory_Ci* pour chaque AC. Ces *memory_Ci* assurent l'étanchéité des souvenirs de chaque AC et concrétisent le souhait de n'offrir qu'une vue partielle de l'environnement global à chacun d'eux. On notera qu'au moment de sa « *naissance* » dans le SMA, chaque AC recopie les règles de *basicrules* qui le concernent dans le champ *rules* de son $\mathbf{Enr_i}$ dans la table *agent*. De même, la table *memory_Ci* qui lui est dédiée est créée si elle n'existe pas.

Champ	Description	Exemple de contenu
code	Identifiant unique	1
evt_date	Date de création	2013-09-30
evt_id	Type d'évènement sur le souvenir (1 = création, 2 = validation humaine)	1
agent_id	Identifiant de l'agent	3192
parameters_in	Paramètres en entrée	stepNumber=1;useOnlyResponses=false;controller=RDEngine;resultsNumber=1;signalCode=200;useMemory=true;action=callRdEngineForBestActor;agentId=3192
decision_string	Décision prise	<reponses><reponse><num>1</num><memoryCode>#</memoryCode><userid>3152</userid><firstname>Xavier</firstname><lastname>PASTEAU</lastname><service>XPS</service></reponse></reponses>
human_validated	Validation humaine	TRUE
comment	Commentaire décrivant à quel type de comportement de l'agent s'applique le souvenir	Trouver meilleur acteur

Tableau 3 : Illustration du contenu d'un enregistrement de la table mémoire d'un agent

Tout AC appartient à un domaine de compétence intégré à la table *agt_domain*. Chacun de ces domaines peut avoir un ou plusieurs domaines pères référencé dans *agt_domain_agt_domain*. On notera simplement que la valeur du champ *rang_domain_agt* est utilisée dans l'interface GRAILS de l'utilisateur pour déterminer l'ordre de restitution à l'écran des domaines dans un explorateur[6] de compétences. Comme pour les compétences, tout objectif enregistré dans *objective* appartient à un

[6] Type explorateur de fichier sous Windows par exemple

domaine d'objectif de *obj_domain*. Ces derniers sont également hiérarchisables entre eux via la valeur du champ *pere_id*.

Chaque AC est concrétisé dans le mode réel par un ou plusieurs acteurs humains. Ceux-ci sont recensés dans une table des utilisateurs absente de la figure 10. Chacun d'eux peut alors être lié à un AC via des enregistrements dans la table *agt_user*. Chaque acteur peut posséder un certain niveau de compétence défini pour une compétence donnée, ce qui se traduit par un enregistrement dans *agt_user_level*. Dans le chapitre 2, nous avons vu que les ACs peuvent être liés entre eux. Ces liens sont fournis (puis appris) dans la table *agent_agent*. Nous avons également proposé qu'une compétence soit vu dans nos travaux comme décomposable en n compétences élémentaires, encore appelées critères d'efficacité. La table « *efficency* » liste ces critères. La jointure avec la table *agent* est assurée par les enregistrements de la table *agent_efficency*.

Les projets sont des instances d'objectifs[7] pris dans *objective*. Nous avons proposé dans notre modèle théorique que les projets soient composés d'un enchainement d'AC. Ils sont évalués à postériori par un ou plusieurs acteurs humains, et selon un ensemble de critères d'efficacité. Chaque enregistrement de la table *agt_objective* décrit un projet avec en particulier le lien entre celui-ci (*objective*), les ACs (*agent*) qui le composent et les critères d'efficacité (*efficency*). La table *agt_objective_eval* stocke les données relatives aux formulaires d'évaluation de chaque projet. On y trouve, par exemple, l'identifiant de chaque évaluateur (e.g. demandeur et offreur de compétence au CG33) dans le champ *evaluator_id*. Les détails de chacun de ces formulaires, c'est-à-dire la valorisation accordée à chaque critère d'efficacité, pour chaque projet, sont stockés dans les enregistrements de la table *agt_objective_eval_agt_objective_eval_criterias*.

2.5.4 Les 3 composantes clé

2.5.4.1 Mécanismes de perception et d'échange

Les agents évoluant dans notre SMA reçoivent et transmettent différentes informations. Si l'on se place du point de vue d'un agent, on appelle « *requête* » l'ensemble des informations (ou paramètres) reçus en entrée lorsqu'une demande lui est adressée. Tout au long de leur vie, les agents sont à l'écoute de requêtes qui leur sont adressées. Elles peuvent provenir d'autres agents à l'intérieur du SMA ou bien des utilisateurs humains du système. Dans ce dernier cas, pour assurer la compatibilité avec les normes FIPA (FIPA, 2002), il ne doit pas y avoir d'échange direct entre les utilisateurs et les agents. Dans notre implémentation, nous utilisons l'outil (framework de développement multi-agents) Open Source JADE (Bellifemine, 1999). Comme proposé par JADE, on utilise un agent technique nommé « *WebRequester* » pour ce type d'échanges. Voir figure 11.

[7] Cf. paragraphe « *2.4.2 Objectif* » plus haut

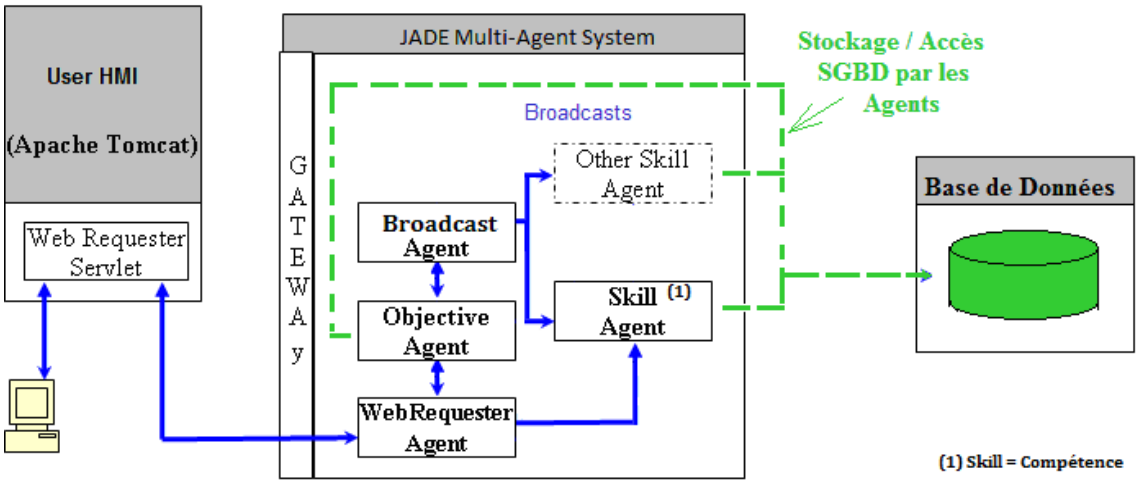


Figure 11 Echanges de flux dans notre SMA

☞ **Remarque :**

Afin d'éviter les redites, la description détaillée de tous les mécanismes de perception et d'échange (i.e. schémas d'échange de flux, liste détaillée et illustrée des paramètres d'entrée présents dans chaque requêtes, cinématique des mécanismes, etc.) sera présenté dans le paragraphe « *Modèle comportemental* » du chapitre 3 décrivant l'implémentation de nos travaux. Cependant, de manière exhaustive mais synthétique, on retiendra que :

A. **L'agent *WebRequester*** est purement technique.
Il se contente de véhiculer les requêtes en provenance des utilisateurs humains vers l'un des agents (Objective ou Compétence) du SMA. Il renvoie ensuite la réponse de l'agent interrogé vers les utilisateurs humains.

B. **L'agent *Broadcast*** est purement technique.
Il se contente d'effectuer un broadcast de la requête qu'il reçoit en provenance d'un agent quelconque du SMA (e.g. Objective) vers tous les autres agents (e.g. Compétence) et de consolider les réponses reçues. Il renvoie finalement le résultat consolidé à l'agent qui l'a sollicité au départ.

C. ***L'agent Objective*** reçoit des requêtes en provenance des utilisateurs humains, via l'agent technique WebRequester. Ces dernières permettent de consulter la liste des compétences ayant été mises en œuvre dans des objectifs passés ou bien d'en créer de nouveaux. Cette création peut être réalise de 2 manières distinctes :
   1. Soit par duplication partielle ou globale de la liste des compétences impliquées dans un objectif choisi.
   2. Soit par interrogation des ACs, à travers les agents Objective puis Broadcast, à qui l'on transmet l'information concernant la création d'un nouvel objectif. Chaque AC interrogé se porte alors candidat ou non à participation au nouvel objectif.

D. ***L'agent Compétence*** reçoit 2 types de requêtes :

1. Suite à broadcast émis depuis l'agent Objective, via l'agent technique Broadcast, lors de la création de nouveaux objectifs.
2. En provenance des utilisateurs humains, via l'agent technique WebRequester, lors des demandes de liste d'acteurs pouvant exercer une compétence, lors de demandes de création de liens[8] inter-ACs, ou bien lors de l'intégration d'évaluations à postériori d'acteurs ayant exercé la compétence dans le cadre d'un projet[9].

2.5.4.2 Attributs internes

Un AC est défini par un ensemble d'attributs internes. De manière synthétique[10], on distingue:

1. Un identifiant numérique unique.
2. Une description textuelle simplifiée (champ *description*). Ex : « Ailier droit » au rugby.
3. Une description textuelle détaillée (champs *abstract_content*).
4. Un booléen indiquant que l'agent est en mode apprentissage ou non.
5. Le nombre de fois où l'agent a été utilisé dans des projets.
6. Un modèle comportementale évolutif au format XML.
7. Un âge[11] (*enfance*, *adolescence* ou *maturité*) courant.
8. Une liste de mots clé. Exemples pour le rugby : « attaquer, rapidité, agilité, marquer, contre-attaquer ».
9. Une date de création de l'agent par l'administrateur[12].
10. Une référence vers les règles éducatives de base qui lui ont été attribuées au moment de sa création (champs *basic-rules_id*).

A ces attributs, il convient d'ajouter une liste de compétences élémentaires[13] C_{et} (avec $C_{et} \in CE_t$), référencées dans la table *agent_efficency*. De plus, ils peuvent évoluer, ce qui rend nos ACs dynamiques. En effet, la liste des C_{et} varie, le nombre de fois où l'agent est utilisé dans les projets également, tout comme le modèle comportemental lui-même[14].

---

[8] Cf. paragraphe « *1.4.8 Relations entre compétences* » au chapitre 1
[9] Cf. paragraphe « *1.2 Définitions, concepts & notions clé* » au chapitre précédent
[10] Cf. paragraphe « *2.5.3 Modèle de données* » plus haut
[11] Cf. paragraphe « *2.5.6 Cycle de vie* » plus bas
[12] Cf. paragraphe « *2.5.2 Proposition* » plus haut
[13] Cf. paragraphe « *1.2.6 Compétence élémentaire* » plus haut
[14] Cf. paragraphe « *2.5.7 Modèle comportemental* » plus bas

2.5.4.3 Intentionnalité et buts

Tels que nous les avons définis au paragraphe « *2.5.2 Proposition* », les ACs modélisent le concept de compétence sous la forme d'agents logiciels qui vivent et évoluent au sein d'un système apprenant. Ils font preuve de dynamisme et ne sauraient être considérés comme purement statiques. Ce sont des agents cognitifs. Cependant, dans la réalité des faits et la plupart des applications informatiques, ce sont souvent les acteurs humains qui sont modélisés sous la forme d'agents, les compétences ne traduisant que leurs comportements (Ferber, 1999) (Holger, 1998). Considérons l'inverse : une compétence modélisée sous la forme d'un agent. Parler de perception, d'intention, de motivation, ou moyen d'action sur l'environnement, ne semble pas trivial de prime abord. Sachant qu'en IA la notion d' « *embodiment of mind* » (Ferber, 1999) (Pfeifer, 2007) (Shapiro, 2010) est essentielle pour pouvoir parler d'agent cognitif, nous proposons de définir ces éléments artificiellement. Nous suggérons ainsi qu'il y a du sens à supposer qu'une compétence, sous la forme d'un agent, soit motivée à minima par l'atteinte des 4 buts suivants :

1. Constitution de sa propre définition : identification de la liste *compétences élémentaires* qui la compose. Cette liste peut évoluer en fonction des besoins des projets, et conduire éventuellement à la disparition de la compétence elle-même.
2. Restitution d'une liste d'acteurs humains capable de la concrétiser au sein d'un projet. Cette restitution vient en réponse à la demande d'un utilisateur du système qui recherche un acteur humain capable de répondre à son besoin, traduit par un projet.
3. Etablissement de relations avec les autres compétences du SMA. Les relations (appelées « liens ») entre 2 compétences, peuvent être mises en évidence par ce que nous appelons l'analyse des « similitudes » existantes entre elles. Ce peut être leur participation à des projets communs, des éléments de définition qu'elles partagent, etc.
4. Souhait de se porter candidate à une participation à un nouveau projet qui vient d'être créé dans le système.

2.5.4.4 Mécanismes d'action

Un AC met en œuvre des mécanismes d'apprentissage et des règles. Il exerce des actions qui traduisent des comportements et réalise les 4 buts présentés au paragraphe précédent. Tout au long de sa vie, chaque AC va ainsi :

1. Constituer sa propre définition.
2. Restituer une liste d'acteurs humains capables de le concrétiser au sein d'un projet.
3. Etablir des liens (relations) avec d'autres ACs.
4. Se porter candidat à une participation à de nouveaux projets créés par l'utilisateur.

A la réalisation des 4 buts évoqués ci-dessus s'ajoutent :

1. Faire évoluer son « cycle de vie » décliné en 3 « *âges* » (*enfance*, *adolescence* et *maturité*). Chaque âge traduit un niveau d'autonomie différent[15].
2. Intégrer les évaluations à postériori de sa mise en œuvre au sein d'un projet (apprentissage).

Dans le cadre de nos travaux, nous avons modélisé puis implémenté notre propre gestion de modèle comportemental[16].

2.5.5 Ressources

Nous avons vu en introduction qu'un AC peut être réalisé à travers un corps délocalisé (l'acteur humain) ou plutôt multi-localisé, c'est-à-dire pouvant être réalisé dans la « vie réelle » par plusieurs acteurs physiques. Les acteurs humains sont alors ici considérés comme des « *ressources* »[17] des compétences. De même, au sein d'un projet P, un AC peut (et non doit) faire appel à d'autres ACs pour se réaliser pleinement. Le bon niveau de granularité (i.e. de découpage) et de hiérarchisation éventuelle des appels inter-ACs entres eux est alors fortement dépendant des processus métier du domaine professionnel au sein duquel le SMA est utilisé. Si cela s'avère nécessaire, il s'agira concrètement de traduire des savoirs explicites (donc décrits dans un processus formalisé), en une hiérarchie d'appels inter-ACs. Dans ce contexte, les autres ACs dont il est question peuvent alors être assimilés à des ressources du premier AC dans le contexte de P. En généralisant, nous pouvons dire que les ACs peuvent éventuellement, en fonction des processus métier, posséder :

- Un certain niveau hiérarchique les uns par rapport aux autres.
- Un certain enchainement interne, leur permettant de faire appel à telle ou telle autre compétence.

☞ **Remarque :**
Nous ne considérons pas que la base de données soit une ressource à proprement parlé, au sens théorie des SMAs, des ACs. En effet, le SGBD est uniquement vu ici comme un dispositif technique de stockage des informations descriptives de l'environnement des agents, et non comme l'environnement lui-même.

[15] Cf. paragraphe « *2.5.6 Cycle de vie* » plus bas
[16] Cf. paragraphe « *2.5.7 Modèle comportemental* » plus bas
[17] Cette affirmation n'inclut absolument aucune arrière pensée déontologiques et/ou philosophique quant à la place de l'être humain au sein du système.

2.5.6 Cycle de vie

2.5.6.1 Introduction

Dans leur publication (Routier, 2002)[18], Routier, Mathieu et Secq proposent qu'un agent puisse changer de compétences ou en développer de nouvelles. A l'opposé, dans cette thèse, une compétence est un agent du SMA à part entière. Il est unique au sein du système, doit pouvoir être créé, vivre (être utilisé) et éventuellement disparaitre du SMA. Ceci se produit, par exemple, dans le cas où l'évolution de savoirs explicites traduits à travers un processus métier formalisé montrent que :

1. La compétence est englobée par une autre.
2. La compétence n'a plus de raison d'être ou n'est plus utilisée.

☞ **Remarque :**
Concernant le point 2, il peut être envisagé également que le système puisse suggérer à l'administrateur de supprimer une compétence. Ce peut être le cas, par exemple si celle-ci n'est plus demandée ou utilisée dans un projet depuis un certain (paramétrable) temps. Il est bien entendu également possible de doter les ACs d'une capacité d'alerte, déclenchée de manière autonome, via l'ajout d'une règle dédiée à leur modèle comportemental.

Les ACs doivent donc être dynamiques et évolutifs. Cette dernière notion peut se comprendre selon 2 orientations, l'une concernant l'évolution même des connaissances acquises par apprentissage, l'autre concernant le cycle de vie même des ACs. En conséquence, nous proposons de décliner ce cycle en 3 étapes que nous appelons des « *âges* ». Ces âges sont l'*enfance*, l'*adolescence* et la *maturité*.

Lors du passage d'un âge à l'autre, on dira que l'AC « *grandit* » (ou évolue). Les transitions entre les âges sont assurées de manière automatique par des règles comportementales que l'on qualifiera de « *règles de croissance* ». Ces dernières se basent sur l'analyse de conditions environnementales propre à l'AC. A ce stade de nos travaux, les valeurs des seuils de transition entre les âges sont positionnées de manière empirique. L'un de nos objectifs étant de faire grandir nos ACs aussi vite que possible, il nous a fallut trouver un compromis entre atteindre un haut niveau d'expertise et aider l'utilisateur le plus rapidement possible.

[18] Cf. paragraphe « *1.4 Gestion des compétences & SMAs* » au chapitre précédent

2.5.6.2 Enfance

C'est l'âge qui est attribué à un nouvel AC créé par l'administrateur dans le système. L'AC se voit attribués :

1. Une liste minimale de compétences élémentaires (C_{et}) pour le définir. On désigne ces C_{et} sous le terme de « critères d'efficacité » car elles vont permettre d'évaluer l'AC dans le cadre des futurs projets où il sera impliqué.
2. Une liste d'acteurs humains capables de le mettre en œuvre dans les projets.
3. Un modèle comportemental contenant des « règles éducatives de base[19] » dont des « règles de croissance » entre les 3 âges.

Suite à la création d'un nouvel AC, nous ne disposons d'aucun retour d'évaluation de projet où il aurait été mis en œuvre puisqu'il n'a pas encore été utilisé. Dans ce cas, si un utilisateur recherche un acteur (e.g. un joueur de rugby) capable de rendre une compétence (e.g. Ailier Droit), la suggestion de l'AC n'est pas pertinente. Il retourne simplement à l'utilisateur une liste d'acteurs capables de le concrétiser, triée en fonction du nombre de fois où chacun d'eux a été impliqué dans un projet (e.g. un match), le plus expérimenté à la fin. Une fois le projet achevé, l'utilisateur évalue (via les C_{et}) la concrétisation de l'AC, par l'acteur sélectionné, dans le cadre du projet. Les données d'évaluation sont alors stockées en base de données.

A cet âge, l'AC ne communique pas réellement avec les autres agents. Il ne cherche pas, par exemple, à établir de liens avec les autres agents ni à participer aux nouveaux projets introduits dans le système. Il n'utilise que les « règles éducatives de base » qui lui ont été fournies au moment de sa création. Il n'a pas la capacité de les modifier. Tout l'enjeu consiste à « faire grandir » notre AC. On considère qu'il va grandir automatiquement lorsque la condition de passage à l'âge supérieur est réalisée. Cette condition repose sur une valeur de seuil et correspond au nombre de fois où l'AC a été concrétisé dans des projets (e.g. 3 projets). Le passage à l'âge supérieur est automatique une fois la condition réalisée.

2.5.6.3 Adolescence

A l'âge d'enfance, lors de la recherche d'acteurs pour une compétence donnée, l'AC ne prend en compte que les nombres de participation des acteurs à des projets antérieurs. Il n'y a pas de mise en œuvre de mécanismes d'apprentissage particuliers, l'AC se contentant de présenter une liste d'acteurs triée. Adolescent, la différence fondamentale est l'intégration des mécanismes d'apprentissage[20]. Lors de la mise en œuvre de ces mécanismes, l'AC utilise :

- Les résultats des évaluations des projets antérieurs (précédentes expériences).

---

[19] Cf. paragraphe « *2.5.7 Modèle comportemental* » plus bas
[20] Cf. paragraphe « *2.5.7.5 Comportement N°2 : sélection d'acteurs* » plus bas

- Les poids (niveaux d'importance) attribués à chaque critère par l'utilisateur au moment du lancement de sa requête. Si aucun poids n'est spécifié, la valeur de chacun des poids est 0 par défaut.

Un AC, adolescent va construire la liste d'acteurs potentiels après avoir effectué un calcul dynamique des valeurs des poids de chaque critère. Il va également tenter de réaliser 1 nouveau but : établir des relations (ou liens) avec d'autres ACs[21]. Dans les deux cas, une validation humaine est requise est stockée dans les souvenirs de l'agent (table dédiée en base de données).

Ici également, nous souhaitons « faire grandir » notre AC. On considère qu'il va grandir automatiquement lorsque la condition de passage à l'âge supérieur est réalisée. Cette condition repose sur une valeur de seuil et correspond au nombre de fois où l'AC a été concrétisé dans des projets (e.g. 10 projets). Le passage à l'âge supérieur est automatique une fois la condition réalisée.

2.5.6.4 Maturité

A cet âge, un volume significatif de données concernant les projets et leurs évaluations à travers des données représentatives d'expériences réelles, sont disponibles et exploitables par l'AC. Dans ce contexte, nous considérons que ce dernier devient capable de faire des suggestions pertinentes et de réaliser un autre but : proposer sa candidature à une participation à un nouveau projet inséré dans le système[22]. Ainsi, dès qu'un nouveau projet est créé, une liste de compétences, (avec éventuellement la liste des acteurs possibles si cela est autorisé[23]), est proposée à l'utilisateur. Ce dernier peut toujours adapter cette liste mais gagnera beaucoup de temps pour l'élaboration de son projet si les choix effectués par le système sont pertinents et correspondent à son besoin.

A cet âge, l'AC mature a une bonne connaissance des liens qu'il possède avec les autres. Par paramétrage du système, il est aussi possible d'annuler la demande de validation humaine. Par exemple, dans notre cas de BCDD, nous pouvons imaginer caractériser un nouveau projet (e.g. construire un nouveau collège) sans connaître la liste des compétences nécessaires ni les acteurs à impliquer. Dans cette situation, le système peut identifier les AC composant le projet et chacun de ces AC affecter automatiquement sa concrétisation à un des acteurs sans nécessiter de validation humaine. L'AC est donc entièrement autonome, ne nécessitant plus de validation obligatoire par l'homme.

[21] Cf. paragraphe « *2.5.7.6 Comportement N°3 : liens inter-agents* » plus bas
[22] Cf. paragraphe « *2.5.7.7 Comportement N°4 : candidater à un nouveau projet* » plus bas
[23] Dans la BCDD au CG33, le principe d'anonymisation s'oppose à présenter les acteurs nominativement

2.5.6.5 Conclusion

Le cycle de vie que nous proposons assure le dynamisme et l'évolutivité des agents compétence tout au long de leur vie dans le SMA. Leur niveau d'autonomie s'accentue progressivement, lorsqu'ils atteignent les différents âges et grandissent. Les comportements qu'ils réalisent intègrent des mécanismes d'apprentissage par l'expérience, avec stockage d'informations de validation en mémoire, les rendant de plus en plus pertinents. Les calculs de différents coefficients et poids des C_{et} s'optimisent de plus en plus. Le modèle comportemental de chaque AC évolue même concrètement à travers la création dynamique de règles comportementales, suite à la recherche de lien avec d'autres ACs par exemple. Nous sommes donc loin d'un modèle statique tel que généralement implémenté dans un logiciel de gestion de projet ou une bourse d'échange. Nous le constatons, l'apprentissage est au cœur du système.

2.5.7 Modèle comportemental

2.5.7.1 Introduction

Avec la modélisation que nous proposons, sous la forme d'ACs, nous devons répondre concrètement aux enjeux suivants (Chator, 2013) :

1. Renseigner un utilisateur du système sur l'expertise et le savoir-faire des partenaires potentiels en identifiant, pour chaque compétence, l'acteur (humain ou entreprise) qui est le plus à même de répondre à une demande d'assistance sur le savoir-faire en question.
2. Mettre en relation (faire communiquer) les agents compétence dans le but d'atteindre un objectif proposé par un utilisateur du système. Exemple : introduction d'un nouvel objectif dans le système qui proposera automatiquement à l'utilisateur les compétences à mettre en œuvre, une proposition d'enchainement de ces compétences et leur planification.
3. Déterminer le meilleur enchainement des compétences, suite aux retours d'expérience et validation des choix proposés par le système en fonction du contexte.
4. Définir le cycle de vie des agents compétence, leurs « intentions » et leur modèle comportemental.
5. Définir un système apprenant générique, applicable à tout domaine fonctionnel.

Comme déjà mentionné plus haut, le modèle comportemental de base d'un AC est composé de 4 comportements :

- La constitution de la propre définition de l'AC via l'identification de la liste *compétences élémentaires* qui le compose.

- La restitution d'une liste d'acteurs humains capables de le concrétiser au sein d'un projet[24].
- L'établissement de relations (liens) avec les autres ACs[25].
- L'expression d'un souhait de se porter candidat à une participation à un nouveau projet[26].

Les 3 derniers comportements sont déclenchés par l'AC en fonction de requêtes en provenance des autres agents qui font appel à lui, d'évolutions de son environnement, ou bien directement d'interactions avec l'utilisateur. Nous les décrirons de manière détaillée dans ce qui suit, ainsi que les mécanismes d'apprentissage qui leur sont liés.

2.5.7.2 Technique d'apprentissage

2.5.7.2.1 Contexte

Dans cette première version de nos travaux, pour un AC donné, il nous a été demandé d'intégrer des mécanismes d'apprentissage afin d'optimiser:

- La méthode de sélection des acteurs pouvant concrétiser l'AC au sein d'un projet.
- La méthode d'établissements de liens avec d'autres ACs dans le but de suggérer des listes de compétences adaptées pour les futurs projets.

Deux autres exigences ont également été formulées :

- Faire en sorte que le système s'adapte toujours au mieux à la réalité des cas d'utilisation terrain dans sa déclinaison BCDD au CG33.
- Garder la maitrise du système en termes d'administration et de gestion, i.e. contrôler l'autonomie de prise de décision des ACs lors de l'affectation à un acteur de la réalisation d'une compétence dans un projet. Cette demande est issue de la volonté prépondérante de construire un système où le partage, la réciprocité et le « bénéfice pour tous », en particulier en termes de montée en compétence des offreurs de compétence, sont des préoccupations majeures.

Pour répondre à ces demandes, nous avons défini une cinématique de traitement divisée en trois étapes :

1. Génération de la liste (d'acteurs ou de compétences potentiellement liées) avec stockage en mémoire de l'AC de cette nouvelle proposition en attente de validation.
2. Action manuelle de validation de la liste (complète ou partielle) par l'utilisateur humain via l'interface graphique.

[24] Cf. paragraphe « *2.5.7.5 Comportement N°2 : sélection d'acteurs* » plus bas
[25] Cf. paragraphe « *2.5.7.6 Comportement N°3 : liens inter-agents* » plus bas
[26] Cf. paragraphe « *2.5.7.7 Comportement N°4 : candidater à un nouveau projet* » plus bas

3. Stockage de cette information de validation en mémoire de l'AC ayant généré la liste pour prise en compte dans les futures itérations de traitement.

Le passage par l'étape 2 de validation manuelle est la seule méthode qui puisse nous permettre de toujours répondre constamment à l'exigence de représentativité terrain. Même si d'un point de vue théorique les ACs sont capables d'être entièrement autonomes à l'âge de la maturité, on garde le contrôle sur leur prise de décision.

2.5.7.2.2 Analyse

Plaçons-nous à un niveau d'abstraction plus élevé. L'analyse des demandes et de la cinématique de traitement proposée pour y répondre nous permet de dire que notre méthodologie globale d'apprentissage, au fil de l'eau :

1. Doit toujours être basée sur des cas réels et prendre en compte des validations humaines.
2. N'a pas de connaissance à l'avance des conclusions auxquelles elle doit aboutir.
3. Ne dispose pas de la notion de transition d'état qui est absente de nos ACs.
4. Ne peut s'appuyer sur les notions de récompense / pénalités, absentes de nos ACs.
5. Ne peut s'appuyer sur des considérations de compétitivité inter-ACs.

Le point 1 se vérifie dans la cinématique des traitements. Suite à la constitution de la liste des acteurs ou des ACs potentiellement liés, un AC créé un souvenir (non validé) dans sa table mémoire. Il passera à la valeur « validé » en fonction de la décision de l'utilisateur humain, puis pris en compte pour les futures itérations de constitution des listes. C'est la base même de nos mécanismes d'apprentissage au fil de l'eau.

Le point 2 se vérifie, par exemple, lors de la constitution de la liste des acteurs. Le calcul des valeurs optimales (qui seraient donc la cible) des poids à affecter aux compétences élémentaires sont obtenues à partir de la prise en compte des validations terrain successives (expériences) obtenues par le passé[27]. On ne peut donc pas dire que les valeurs cibles que doit restituer l'algorithme d'apprentissage sont connues à l'avance. C'est la même chose pour l'optimisation par apprentissage de la recherche d'établissement de liens entre ACs. Effectivement, on ne connait pas à l'avance quelles sont les meilleures requêtes générées[28] permettant d'identifier les ACs liées à l'AC que l'on interroge.

Le point 3 se vérifie par le fait que l'apprentissage réside uniquement dans l'optimisation des comportements de constitution de listes d'acteurs ou d'ACs liés. De manière globale, un AC n'a pas d'état particulier dans son cycle de vie qui le fasse évoluer significativement et qu'il cherche à atteindre. Si l'on souhaite considérer que le passage d'un âge à l'autre peut constituer une forme de transition d'état, c'est toujours

[27] Cf. paragraphe « *2.5.7.5 Comportement N°2 : sélection d'acteurs* » plus bas
[28] Cf. paragraphe « *2.5.7.6 Comportement N°3 : liens inter-agents* » plus bas

possible en théorie. Cependant, ceci n'induit aucun impact au niveau du fonctionnement interne des comportements eux-mêmes qui restent toujours déclenchés sur demande de l'utilisateur. Selon l'âge (à partir de l'adolescence), le comportement de création de lien peut être mis en œuvre par l'AC. Cependant, quelque soit l'âge, rien ne change dans les opérations internes du comportement lui-même. On ne peut donc pas parler de gestion de transition d'états induisant un impact particulier pour nos ACs.

Le point 4 se vérifie par le fait que lors de la mise en œuvre des mécanismes de constitution de listes d'acteurs ou d'ACs liés, un AC ne cherche à maximiser aucune récompense. Il n'est pas non plus pénalisé si sa proposition est rejetée par l'utilisateur humain.

Le point 5 se vérifie par le fait que chaque AC est indépendant des autres pour constituer sa liste d'acteurs ou d'ACs liés. Même dans le but de candidater à une participation à un nouvel objectif créé par l'utilisateur, un AC n'entre pas en concurrence (compétition) avec les autres ACs. Il s'agit plutôt de complémentarité au sens où les ACs participent alors à un but commun : concrétiser le projet créé par l'utilisateur.

2.5.7.2.3 Choix d'une méthode

Nous avons vu que nos ACs ne disposent pas de notions de transition d'états, de mécanismes de gestion de récompenses / pénalités, ni de notions de temps et d'espace influant sur la mise en œuvre de leurs comportements. Ce sont les requêtes utilisateur qui dynamisent le système pour ce qui concerne l'apprentissage appliqué à la construction des listes d'acteur et de compétences liées. L'utilisation de techniques d'apprentissage telles que les Processus décisionnels de Markov ou d'apprentissage par renforcement de type essai / erreur (TD-Learning, Q-Learning, etc.) ne nous paraissent donc pas adaptées dans notre contexte. Le constat est identique en regard des mécanismes d'inférence déroulés dans le cadre de la théorie des jeux. Il n'y a pas de notion de compétitivité entre les ACs dans notre contexte. L'apprentissage mis en œuvre dans les réseaux bayésien, quand à lui, repose sur des données calculées de manière probabiliste et non sur des faits réellement observés comme c'est plutôt le cas pour nous. Les méthodes d'apprentissage supervisé (Biemacki, 2003) prédisent le groupe ou « classe » d'appartenance d'une donnée cible connue à l'avance en fonction des valeurs de variables dites « prédictives » qui la caractérisent. Bien que nous intégrions de la validation humaine volontaire dans nos travaux, on ne peut pas parler d'apprentissage supervisé puisque les valeurs cibles (des poids des compétences élémentaires par exemple) ne sont pas connues au départ. Le constat est identique pour les méthodes non supervisées (Grabmeier, 2002), ou semi-supervisées (Vapnik, 1998) où les données cibles doivent être également être réparties dans des classes d'appartenance, ce qui n'est pas représentatif de notre contexte.

A l'opposé, l'utilisation des méthodologies de Case Base Learning (CBL) peut sembler de prime abord être proche de notre besoin. En effet, dans ce type d'apprentissage, l'inférence :

1. Repose sur des itérations d'expériences concrètes, évaluées et mémorisées.
2. Tient compte de décisions prises par le système, validées ou non par un utilisateur humain.
3. Permet une adaptation spécifique et progressive du système.

Cependant, les algorithmes existants (Mc Sherry, 2006) (Mühlenbein, 1992) (Shiu, 2001) (Quinlan, 1993) (Domingos, 1996) (Zaki, 2002) ne sont pas utilisables en l'état. Dans notre contexte, l'objectif n'est pas de trouver des cas similaires à une situation donnée qui serait décrite par les valeurs (prédictives) des paramètres de la requête de l'utilisateur transmise à l'AC unique que l'on interroge. Ce que nous apprenons consiste à optimiser la restitution d'une liste d'acteurs ou de liens potentiels vers d'autres ACs. Il ne s'agit pas, par exemple, de trouver (ou de classer) des ACs en fonction des valeurs des variables de paramétrage transmises au moment de l'exécution de la requête utilisateur. Même dans le cas où l'on demande à notre AC de trouver les compétences qui lui sont liées, l'implémentation ne transmet aucun paramètre. De manière simplifiée, on se contente simplement de demander à l'AC interrogé : « *trouves-moi les compétences qui te sont liées* ». On ne peut donc pas dire que nous recherchions des cas similaires à une situation qui serait traduite par les valeurs prédictives des paramètres de la requête utilisateur.

Finalement, nous proposons nos propres méthodes d'apprentissage dans le but d'optimiser la constitution des listes d'acteur ou de compétences liées à un AC, à travers des mécanismes qui s'inspirent des méthodes d'apprentissage incrémental. Nous avons choisi également de décliner un algorithme spécifique par comportement. Ceci n'est pas choquant dans la mesure où les actions déclenchées lors de la constitution des 2 types de liste sont très différentes de l'une à l'autre.

2.5.7.3 Modélisation

Nous proposons un modèle comportemental générique sous la forme d'un arbre XML, avec une grammaire spécifique. Cet arbre contient des règles comportementales unitaires. Tous les comportements d'un AC, sauf celui d'auto définition[29], y sont traduits

Balise	Attribut	Requis	Commentaire
rules	description	X	Balise principale
	sensor	X	Dans l'implémentation, désigne le nom de la classe java pilotant (pool de thread) tous les capteurs de l'AC
ruleGroup	description		Texte décrivant le groupe de règles
	weight		Poids par rapport aux autres groupes de règles
rule	description	X	Texte décrivant la règle
	weight		Poids par rapport aux autres règles dans le groupe
	mandatory		1 si la règle est obligatoire, 0 sinon
when	description		Texte décrivant la condition de déclenchement
	sensor	X	Capteur de l'AC à utiliser (i.e. nom de la classe java dans notre implémentation) pour vérifier que la condition de déclenchement est remplie
	params		Liste des paramètres du capteur au format ***nom=valeur***. Le séparateur de liste est le caractère **|**
	result		Nom de la variable, débutant par le caractère ***$***, contenant le résultat de l'exécution du capteur
	operator		Opérateur logique dans l'expression de la condition
	table		Dans l'implémentation, désigne le nom de la table du modèle de données où la condition doit être vérifiée
	field		Dans l'implémentation, désigne le nom du champ de la table du modèle de données où la condition doit être vérifiée
	value		Dans l'implémentation, désigne la valeur du champ de la table du modèle de données où la condition doit être vérifiée. Cette valeur est exprimée sous la forme d'une expression régulière (standard java)
otherwise			Clause utilisée si la condition dans **<when>** n'est pas vérifiée.
do	description		Texte décrivant le comportement déclenché
	effector	X	Effecteur de l'AC à utiliser (i.e. nom de la classe java dans notre implémentation) pour réaliser le comportement voulu
	params		Liste des paramètres de l'effecteur au format ***nom=valeur***. Le séparateur de liste est le caractère **|**
	result		Nom de la variable, débutant par le caractère ***$***, contenant le résultat de l'exécution de l'effecteur

Tableau 4 : Grammaire XML du modèle comportemental

[29] Cf. paragraphe « *2.5.7.4 Comportement N°1 : autodéfinition* » plus bas

Voici une première illustration de l'implémentation d'un modèle comportemental. On y trouve :

1. Un premier `ruleGroup` dont la description est « `Growing Policy` ». Ce groupe contient les deux règles de croissance entre les différents âges du cycle de vie de l'AC.
2. Un second `ruleGroup` dont la description est « `Periodic Rules`» contenant la règle « `Finding Agent Links`». Ce groupe contient l'ensemble des règles à déclencher de manière périodique. Par exemple, le comportement d'identification des liens inter-agents. La périodicité de déclenchement est définie dans le paramétrage interne de chaque effecteur, directement dans l'implémentation.

```
<?xml version="1.0" encoding="UTF-8"?>

  <rules description="Behavioral Rules" sensor="sensorRulePiloteClass">

    <ruleGroup description="Growing Policy" weight="1">

       <rule description="Growing from Youth to Teenage" mandatory="1" weight="1">
          <when description="Growing Conditions" sensor="GrowingSensor"
          params="unitaryEvaluationNumber=2" result="$evaluationsNumber1" operator="EQ"
          value="3">
           <do description="Grow to teenage" effector="Grow"
             params="from=youth|to=teenage" />
          </when>
       </rule>

       <rule description="Growing from Teenage to Mature" mandatory="1" weight="1">
          <when description="Growing Conditions" sensor="GrowingSensor"
          params="unitaryEvaluationNumber=2" result="$evaluationsNumber2" operator="EQ"
          value="10">
             <do description="Grow to mature" effector="Grow"
             params="from=teenage|to=mature" />
          </when>
       </rule>

    </ruleGroup>

     <ruleGroup description="Periodic Rules" weight="1">

        <rule description="Find Agent Links" mandatory="1" weight="1">

           <do effector="AgentLinkCreator" request="SELECT agent.* FROM agent,
          agt_objective, objective, obj_domain WHERE
          text(obj_domain.code)=text(objective.domain_objective_id) AND
          text(agt_objective.agent_id)=text(agent.code) AND
          text(objective.code)=text(agt_objective.objective_id) AND
          text(obj_domain.descriptiondomaineobj)=text(&amp;apos;Générique&amp;apos;)
          AND agent.code&amp;lt;&amp;gt;1508"/>

       </rule>

    </ruleGroup>

  </rules>

</xml>
```

Voici une seconde illustration, bien plus générique, du contenu possible de l'implémentation d'un modèle comportemental. Elle permet d'avoir un aperçu des codifications dans les règles des paramètres, tables et champs, etc. :

```
<ruleGroup description="fixedRules1" weight="1">
   <rule description="rule 1" mandatory="0" weight="1">
      <when operator="AND">
         <when description="if 1 of the 3 conditions" operator="OR">
            <when description="first condition" operator="EQ" params="p1=0|p2=1"
               result="$res" sensor="SensorDoNothing" value="0">
               <do description="action if condition 1" effector="EffectorDoNothing"
                params="p1=table1.field1" result="$var1"/>
            </when>
            <when description="second condition" field="field2" operator="EQ"
               table="table1" value="^1(\d+)$">
               <do description="action if condition 2" effector="EffectorDoNothing"
                params="p1=table1.field1|p2=$var1" result="$var2"/>
            </when>
            <when description="third condition with 2 actions" field="$var1"
               operator="EQ" value="$var2">
               <do effector="EffectorDoNothing" params="p1=$var1|p2=$var2"
                result="$var3"/>
               <do effector="EffectorDoNothing" params="p1=$var3"/>
            </when>
            <do effector="EffectorDoNothing" params="p1=$var3"/>
         </when>
         <otherwise>
           <do effector="EffectorDoNothing"/>
         </otherwise>
      </when>
   </rule>
   <rule description="fixedRules2" mandatory="-1" weight="1"/>
</ruleGroup>
```

2.5.7.4 Comportement N°1 : autodéfinition

Une compétence (donc un AC) possède des attributs internes et un ensemble de compétences élémentaires CE_t qui le caractérisent. Réciproquement, une compétence élémentaire peut caractériser une ou de plusieurs compétences. Ainsi, la définition d'une compétence évolue. Le pendant de cette affirmation est que la définition même de l'AC est dynamique. Par exemple, une nouvelle C_{et} peut apparaitre pour un projet P particulier, ce qui modifie donc la définition même de l'AC. Le constat est le même pour la disparition d'une C_{et} du système. Le comportement d'autodéfinition ne possède pas d'implémentation au sens code informatique du terme. Il s'agit plutôt d'un comportement que l'on peut considérer comme « émergent», et qui relève davantage de la conséquence des actions implémentées par deux autres comportements : celui de sélection des acteurs et celui d'établissement des liens inter-agents. La définition d'un AC découle ainsi de la liste des C_{et} et/ou de l'évolution des poids des C_{et} au fil des projets, ainsi que des liens que l'AC établit avec les autres agents.

2.5.7.5 Comportement N°2 : sélection d'acteurs

2.5.7.5.1 Description

Le comportement de sélection des acteurs est déclenché suite à réception d'une requête utilisateur, donc déclenché de manière évènementielle. Cette requête possède deux paramètres principaux que sont la compétence souhaitée et la liste (valeurs numériques) des niveaux d'importance accordés à chaque critère d'efficacité (ou compétence élémentaire) qui la définit. Pour l'illustration, considérons que chaque poste d'une équipe de rugby soit une compétence. Dans ce contexte, imaginons qu'un entraineur adresse la demande qui suit à notre système :

1. Je souhaite trouver quelqu'un qui puisse jouer au poste de « demi de mêlée »
2. Je précise le niveau d'importance (valeur de 1 à 4) que j'attache à chaque critère d'efficacité:
   - Points marqués au pied : 3
   - Touches trouvées : 1
   - Mêlées perdues : 4
   - Matchs déjà joués : 2

Tout l'enjeu consiste à fournir à l'utilisateur une liste d'acteurs humains pouvant concrétiser la compétence demandée. La constitution de cette liste repose sur des calculs, liés aux mécanismes d'apprentissage, et basés sur les valeurs des poids des critères.

2.5.7.5.2 Mécanismes d'apprentissage

Nous proposons un mécanisme d'apprentissage incrémental classique.
Hypothèses de départ :

1. Soit **t** le numéro de la requête de sélection d'acteurs adressée à un Agent Compétence.
2. Soit **k** un critère d'efficacité (une compétence élémentaire) pour évaluer AC.
3. Soit $\mathbf{W_{k(t)}}$ le poids associé au critère **k** pour la requête N° **t**.
4. Soit **f** le nombre de fois où un acteur **a** a concrétisé l'AC dans un projet.

L'apprentissage se déroule en 3 étapes :

Etape 1 : Pour toute nouvelle requête **t**, l'utilisateur précise les poids $\mathbf{W_{1(t)} ... W_{n(t)}}$ qu'il souhaite voir associés à chacune des compétences élémentaires $\mathbf{C_{1t}..C_{nt}}$. Toute valeur $\mathbf{W_{k(t)}}$ non précisée par l'utilisateur sera considérée comme étant égale à **1**.

Etape 2 : Le système recalcule les poids associés à chaque critère.

Si t = 1, le poids moyen de chaque critère est positionné à $\mathbf{W_{k(1)}}$, ce qui correspond aux poids fournis par l'utilisateur pour chacun des critères **k**.

Si t > 1, le nouveau poids moyen de chaque critère **k** est calculé de manière incrémentale à partir des poids de chaque critère dans les précédentes itérations qui ont été mémorisées et validées en utilisant l'équation (4).

$$\overline{W}_{k(t+1)} = \frac{(t \times \overline{W}_{k(t)}) + W_{k(t+1)}}{t+1} \qquad (4)$$

Etape 3 : Le système construit une liste d'acteurs à renvoyer à l'utilisateur en évaluant chacun d'eux. Pour ce faire, nous proposons d'utiliser les nouveaux poids moyens $\mathbf{W_{k(t)}}$, calculés pour chaque critère **k**, ainsi que les résultats des évaluations de l'acteur ayant concrétisé l'AC dans des projets passés. Pour commencer, on utilise l'équation (5) pour calculer une valeur partielle.

$$\overline{E}_{k(a,t)} = \frac{\sum_{j=1}^{f} E_{k(a,j)}}{f} \qquad (5)$$

Ensuite, l'évaluation de chaque acteur est faite en utilisant l'équation (6):

$$Eval(a) = \sum_{k=1}^{t} \overline{W}_{k(t)} \times \overline{E}_{k(a,t)} \qquad (6)$$

Nota Bene : si l'utilisateur a fourni des valeurs spécifiques pour les poids des critères **k**, nous aurons $\overline{W}_{k(t+1)}$ à la place de $\overline{W}_{k(t)}$.

Dans le but de donner une chance à chaque acteur, nous proposons d'alterner entre la recherche de performance (liée aux évaluations) et le souhait de promouvoir le partage (processus de sélection aléatoire) avec une probabilité de 0,5. Ainsi, un acteur qui aurait systématiquement une mauvaise évaluation ne serait pas pénalisé. De plus, dans tous les cas, l'utilisateur peut toujours sélectionner un acteur qui ne se situe pas en haut de la liste. En considérant le cycle de vie des ACs[30], on rappelle qu'une validation humaine des décisions prises par le système est toujours nécessaire, hormis à l'âge de la maturité si on l'a spécifié explicitement par paramétrage. Le processus de validation humaine

[30] Cf. paragraphe « *2.5.6 Cycle de vie* » plus haut

consiste concrètement à mettre à jour un booléen de validation dans un souvenir (un enregistrement dans une table) de l'AC. Cet enregistrement conserve la trace des $\overline{W}_{k(t+1)}$ calculés, au fil des requêtes, pour chaque critère **k**. Lorsque la validation humaine est prononcée, et le booléen de validation positionné à « vrai », ces poids sont considérés comme devenant des $\overline{W}_{k(t)}$ validés. Ils seront alors intégrés lors du calcul des futurs $\overline{W}_{k(t+1)}$.

Cette manière de procéder assure l'apprentissage des AC et n'est pas sans rappeler les mécanismes de Case-Based Learning. On notera également qu'elle présente l'avantage non négligeable de refléter une certaine « réalité », représentative de l'univers professionnel dans lequel le SMA est utilisé. L'inconvénient de la méthode proposée réside sans doute dans la prise en compte à égalité de toutes les expériences, aussi lointaines soient-elles dans le temps, ce qui n'est pas nécessairement pertinent.

2.5.7.6 Comportement N°3 : liens inter-agents

2.5.7.6.1 Description

Généralement, la notion de compétence liée est rattachée à une notion juridique (Montaignac, 1999). Dans nos travaux, elle a bien entendu un autre sens et s'appuie sur un indicateur que nous nommons « *coefficient de proximité* » entre 2 compétences. Cet indicateur est exprimé sous la forme d'un pourcentage et est calculé à partir de données présentes dans l'environnement de l'agent compétence qui cherche à établir des liens. Sa valeur est fonction de 2 variables :

- Le nombre de données communes entre 2 compétences dans l'environnement.
- Le niveau d'éloignement[31] de chacune de ces données communes (voir paragraphe suivant).

Un utilisateur peut rechercher les compétences potentiellement liées à celle qu'il souhaite mettre en œuvre. C'est souvent le cas lorsqu'on ne connaît pas la liste exhaustive des actions à réaliser pour mener à bien un projet complet et que ledit projet n'est pas décrit[32]. Dans notre proposition, le comportement d'établissement de liens inter-agents peut être déclenché de 2 manières :

1. Suite à réception d'un évènement : requête utilisateur.
2. Suite à déclenchement d'une règle comportementale, donc de manière automatique & autonome.

[31] Voir paragraphe « *2.5.7.6..2 mécanismes d'apprentissage* » plus bas
[32] Dans le cas où un projet existe, l'utilisateur préfèrera déclencher le quatrième comportement de l'AC

Illustrons le premier cas par un exemple. Un utilisateur, qui travaille dans une mairie, souhaite mener une campagne de sensibilisation de la population sur le sujet des éoliennes. Ce sujet n'a pas, par le passé, déjà fait l'objet d'un projet. En dehors de compétences purement techniques sur les éoliennes, notre utilisateur sait qu'il a besoin d'une compétence en organisation d'évènements. Il va chercher à identifier celles qui lui manquent pour organiser son projet. Il adresse une requête à l'AC « organisation d'évènement » qui lui renverra la liste des compétences qui lui sont liés, comme par exemple « réaliser campagne de communication auprès des citoyens » ou bien « réserver la salle polyvalente ».

Le second cas, quand à lui, est déclenché de manière périodique et autonome par l'exécution d'une règle comportementale. Chaque AC, au minimum à l'âge d'adolescence, tente d'établir des relations avec les autres au sein du SMA. L'identification des liens met en œuvre des mécanismes d'apprentissage décrits au paragraphe suivant.

2.5.7.6.2 Mécanismes d'apprentissage

L'apprentissage, pour chaque AC, repose sur la constitution de l'indicateur appelé « *coefficient de proximité* ». Il s'agit d'un pourcentage dont le calcul est basé sur la recherche de similarités entre compétences. On utilise pour ce faire des mécanismes de génération dynamique de requêtes sur la base de données.

Calcul du coefficient de proximité :
Deux compétences sont relativement proches l'une de l'autre, si elles ont en commun des compétences élémentaires, des mots-clés, si elles sont dans les mêmes domaines de compétence, ou si elles ont été mises en œuvre dans des projets communs. Il est difficile de pondérer chaque critère. Nous proposons une méthode simple, basée sur le calcul du nombre de descripteurs communs aux 2 compétences, chaque descripteur appartenant à une des listes suivantes :

1. Les mots-clés des ACs[33].
2. Les domaines de compétence auxquels elles appartiennent.
3. Les compétences élémentaires qui les caractérisent.
4. Les projets passés dans lesquels elles ont été impliquées.
5. Les domaines fonctionnels de ces projets.
6. Les types de ces domaines de projet.

La liste fournie ci-dessus peut être obtenue de manière automatique en parcourant dynamiquement et récursivement (en suivant les clés étrangères) l'arbre de base de données, à partir d'une racine précise : la table des agents. En parcourant cet arbre, intuitivement, on s'aperçoit que le graphique de la figure 12 tend bien à montrer des

[33] Cf. paragraphe « *2.5.4.2Attributs internes* » plus haut pour la liste exhaustive de ces attributs

niveaux d'importance allant du plus faible (à gauche) au plus fort (à droite). Effectivement, plus la profondeur est faible, plus les similitudes peuvent être considérées comme « significatives ».

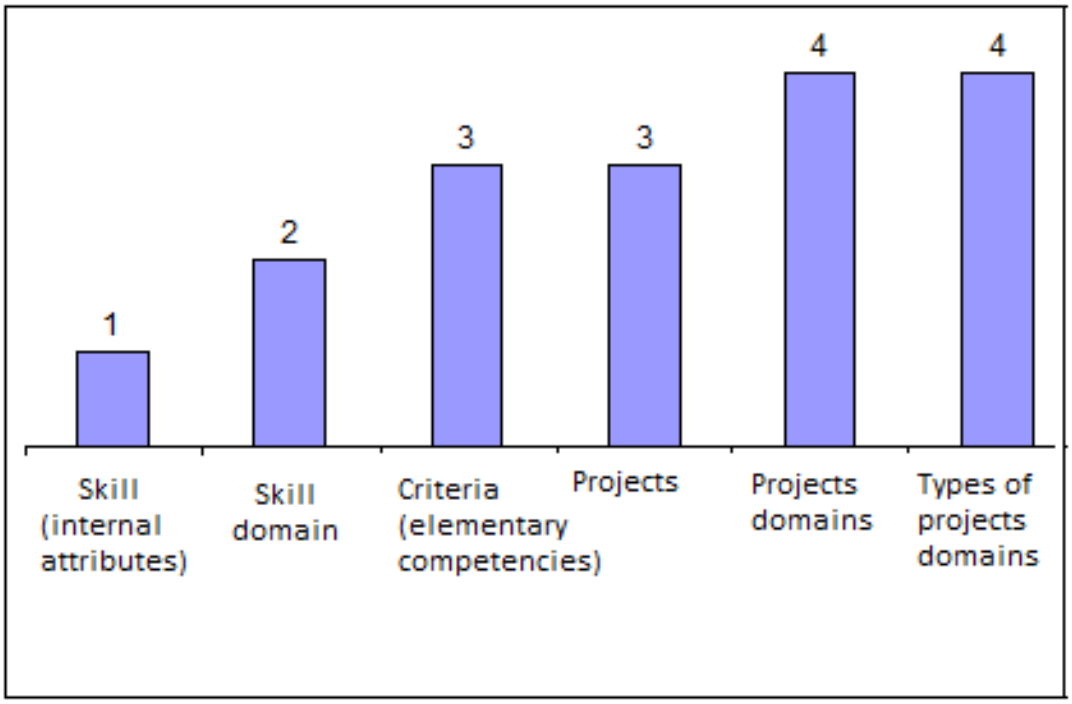


Figure 12 Etablissement de liens inter-agents : profondeur & similarités

Si l'on considère que **D** est la profondeur maximale de l'arbre, cela signifie que toute compétence peut être « rejointe » à partir d'une autre en suivant au maximum **D** liens. **D=4** dans l'exemple de la figure 12. Au début de l'exploration de l'arbre, la profondeur de la table courante est **d=1**.

L'algorithme d'apprentissage se compose de 6 étapes.

Etape 1 : pour chaque champ de la table courante, une requête SQL est générée dynamiquement dans le but d'identifier les autres compétences partageant les mêmes valeurs de champs (descripteur).

Etape 2 : si une compétence S_i est trouvée la première fois, son coefficient de proximité courant est positionné à $\mathbf{P_{Si} = 0}$.

Etape 3 : si **n** est le nombre de valeurs communes de champs dans la table courante, $\mathbf{P_{Si}}$ est mis à jour en utilisant l'équation (7).

$$P_{Si} = P_{Si} + (n \times (D - d)) \quad (7)$$

Etape 4 : Les clés étrangères et tables de jointure sont utilisées pour parcourir l'arbre jusqu'à la prochaine table. **d** est alors incrémenté : **d=d+1**.

Etape 5 : si **d** n'est pas supérieur à **D**, aller à l'étape 3.

Etape 6 : finalement, tous les $\mathbf{P_{Si}}$ sont normalisés en utilisant l'équation (8):

$$P_{Si} = \frac{P_{Si}}{P_{Si\,\max}} \times 100 \qquad (8)$$

Plus le coefficient de proximité est élevé, plus les compétences sont considérées comme proches. Une valeur de seuil (e.g. 65% par défaut) est appliquée sur les **Psi**, et une liste d'ACs potentiellement liés est retournée à l'utilisateur pour validation. Cette dernière est demandée de 2 manières différentes à l'âge de l'adolescence:

1. Directement à l'écran si c'est l'utilisateur qui effectue la recherche de lien de manière active.
2. Par mail à l'administrateur de la Bourse aux Compétences si la recherche de liens est déclenchée par l'agent de manière autonome.

Un exemple, illustré avec des copies d'écran, est présenté à la section « *3.11 Comportement N°3 : identifier des liens inter ACs* ». Dans les 2 cas, la validation est associée à la création dynamique d'une règle comportementale, ce qui implique une évolution du comportement[34] même de l'agent. En considérant le cycle de vie des ACs[35], on rappelle qu'une validation humaine des décisions prises par le système est toujours nécessaire, hormis à l'âge de la maturité si on l'a spécifié explicitement par paramétrage. L'AC crée alors ses propres liens avec les autres et les règles comportementales associées sans demander de validation à quiconque. Nous sommes bien ici dans un système apprenant, qu'il y ait passage ou non par une validation humaine intermédiaire. L'une des pistes de travail à explorer est l'optimisation du calcul de la valeur de seuil du coefficient de proximité. Actuellement, l'utilisateur peut le définir, s'il le souhaite, et transmettre sa valeur au système parmi les paramètres qui lui sont envoyés. Dans le cas contraire, c'est la valeur par défaut (positionnée arbitrairement à 65%) qui s'applique.

[34] Voir plus haut la règle `="Find Agent Links"` dans le paragraphe d'illustration des règles comportementales

[35] Cf. paragraphe « *2.5.6 Cycle de vie* » plus haut

2.5.7.7 Comportement N°4 : candidater à un nouveau projet

2.5.7.7.1 Description

De manière générale, lorsqu'un utilisateur souhaite créer un nouveau projet, deux situations peuvent se présenter :

1. Un projet similaire existe dans le système et reflète finalement les attentes. Il devient possible de construire le nouveau projet en utilisant tout ou partie des compétences mises en œuvre dans le projet existant.
2. Il n'y a pas de projet existant dans le système qui reflète les attentes. Compte tenu d'une certaine cohérence fonctionnelle, nous proposons de construire une liste globale de compétences à partir de toutes celles qui sont réparties dans tous les projets du même domaine d'objectif.

Il ne s'agit pas ici de rechercher simplement en base de données la liste des compétences mais bien de rendre les ACs « moteur » de leur proposition de participation au nouvel objectif.

En termes de traitement, 6 étapes sont enchainées :

1. La requête utilisateur est adressée à un agent particulier nommé « *ObjectiveAgent* ».
2. Ce dernier effectue un broadcast de la requête vers tous les ACs du SMA.
3. L'AC réalise le calcul d'un indicateur appelé « *coefficient de pertinence* », exprimé en pourcentage. Ce coefficient s'obtient en divisant le nombre de projets antérieurs du domaine fonctionnel auxquels l'AC a participé par le nombre total de projets réalisés dans ce même domaine. Si la valeur de ce coefficient est supérieure à un certain seuil, l'AC renvoie à l'agent *ObjectiveAgent* un booléen indiquant sa volonté de participation au projet.
4. *ObjectiveAgent* consolide les réponses en provenance des ACs du SMA et renvoie finalement à l'utilisateur la liste de ceux qui souhaitent participer à la réalisation du nouveau projet.
5. A réception de la liste proposée, l'utilisateur valide tout ou partie des compétences proposées pour construire son projet.
6. Le système mémorise cette action de sélection dans le but d'optimiser le calcul du coefficient de proximité pour les futures itérations de calcul. Bien entendu, l'utilisateur peut toujours ajouter et/ou retrancher des compétences à partir de la liste globale si cela s'avère nécessaire.

Pour illustrer concrètement le propos, considérons 3 projets existants appartenant tous au domaine d'objectif « *Nouveaux moyens de production d'Energie* ».

Le projet N°1, « *Installation d'une éolienne* », a 5 phases :

- Gestion de projet / réalisation.
- Définition des pré-requis de puissance électrique.

- Sélection de la meilleure technologie d'éolienne parmi celles disponibles sur le marché.
- Définition de la logistique (organisation du transport).
- Installation de l'éolienne.

Le projet N°2, « *Installation d'une microcentrale hydraulique* », a 8 phases dont 3 communes (en italique) avec le projet « *Installation d'une éolienne* » :

- *Gestion de projet / réalisation*.
- *Définition des pré-requis de puissance électrique*.
- Sélection de la meilleure technologie de microcentrale hydraulique parmi celles disponibles sur le marché.
- *Définition de la logistique (organisation du transport).*
- Installation de la microcentrale.
- Transfert technologique pour développer le tissu industriel sur ce type d'activités.
- Formulation et évaluation des projets pour les organisations humanitaires aux niveaux national et international.
- Formation sur la technologie et le modèle économique des petites centrales hydroélectriques.

Le projet N°3, « *installation d'un système de production de biodiesel pour alimenter un chantier de construction de route* », a 7 phases dont 3 également communes (en italique) avec le projet « *Installation d'une éolienne* » :

- Identifier le meilleur site.
- Evaluation des bénéfices environnementaux.
- *Gestion de projet / réalisation*.
- *Définition des pré-requis de puissance électrique*.
- Sélection de la meilleure technologie de production de biodiesel parmi celles disponibles sur le marché.
- *Définition de la logistique (organisation du transport).*
- Installation du système.

Supposons à présent que l'utilisateur souhaite réaliser un nouveau projet « *installation de panneaux solaires* » qui appartienne au même domaine d'objectif « *Nouveaux moyens de production d'Energie* ». Le coefficient de pertinence est fixé à 75%. Ainsi, en accord avec l'algorithme proposé :

1. *ObjectiveAgent* effectue un broadcast de la requête utilisateur à tous les ACs dans le SMA.

2. Chaque AC calcule son coefficient de pertinence. Si sa valeur est supérieure à 75%, l'AC renvoie à *ObjectiveAgent* la valeur booléenne « *vrai* » indiquant son souhait de participer au nouveau projet.
3. *ObjectiveAgent* consolide les réponses contenant la valeur « vrai » dans un flux XML retourné à l'utilisateur. Ce flux indique que les 3 compétences candidates sont:
   - *Gestion de projet / réalisation*.
   - *Définition des pré-requis de puissance électrique*.
   - *Définition de la logistique (organisation du transport).*
4. En utilisant l'interface graphique en front-office, l'utilisateur sélectionne (valide) globalement ou partiellement les compétences de la liste qui participeront à son projet.

2.5.7.7.2 Mécanismes d'apprentissage

A ce jour, concrètement, il n'y a pas de mécanismes d'apprentissage implémentés pour ce comportement. En particulier, le coefficient de pertinence n'est pas déterminé par calcul, il est simplement contrôlé par l'utilisateur. Le système se contente à l'heure actuelle de mémoriser, comme pour les autres comportements déjà décrits, l'action de validation humaine concernant l'intégration de tout ou partie de la liste des compétences proposées. De futurs travaux, en lien avec les techniques de Case Based Learning, devront permettre d'améliorer le calcul du coefficient de proximité par les ACs et/ou la pertinence globale de la liste des compétences candidates à un nouveau projet. Une autre piste pourrait être l'utilisation d'une méthode d'apprentissage progressif où, par exemple, nous pourrions vouloir optimiser le rapport entre le nombre de compétences effectivement choisies par l'utilisateur versus le nombre global de compétences candidates.

2.5.7.8 Conclusion

Nous venons de voir que le modèle comportemental des ACs se compose de 4 comportements fondamentaux et possède également une représentation qui lui est propre, sous la forme d'un arbre de règles XML. L'un des points notables de ce que nous venons de présenter est le dynamisme de ce modèle. Si l'on raisonne par similitude avec les capacités humaines, on peut dire qu'un AC apprend de ses expériences et qu'il est capable de reproduire un comportement appris par la suite. Cette affirmation est avérée dans le cadre de la modification dynamique des règles comportementales d'établissement des liens inter-ACs.

CHAPITRE 3 – Implémentation & Résultats

« La théorie, c'est quand on sait tout et que rien ne fonctionne. La pratique, c'est quand tout fonctionne et que personne ne sait pourquoi. Ici, nous avons réuni théorie et pratique : Rien ne fonctionne... et personne ne sait pourquoi ! »

Albert Einstein

« Je n'ai pas échoué. J'ai eu 10 000 idées qui n'ont pas marché »

Benjamin Franklin

3.1 Introduction

Le lecteur trouvera dans ce chapitre de nombreux éléments descriptifs de l'implémentation concrète des principes que nous avons exposés dans le chapitre 2 de modélisation des ACs. L'une des forces de ce chapitre est de décrire fidèlement un outil que nous avons implanté de manière opérationnelle au CG33. L'outil informatique BCDD est en effet disponible (première version en décembre 2013). 61 collectivités territoriales, membres du réseau des Agendas 21 de la Gironde l'utilisent. Ceci montre que notre modèle théorique a été implémenté et fournit des résultats concrets. Nous décrirons dans ce chapitre :

- L'architecture globale du système.
- L'interface home machine (IHM) gérant les échanges entre utilisateur et SMA.
- Le modèle de données qui traduit l'environnement des ACs dans le SMA.
- Le framework de développement que nous avons conçu et qui propose une évolutivité du système via l'utilisation de modules (plugins) indépendants.
- Un focus sur l'implémentation des mécanismes relatifs à la gestion du au cycle de vie des ACs, à la mémorisation des décisions qu'ils prennent.

Pour chacun des comportements d'un AC, nous décrirons les mécanismes d'apprentissage et d'échange avec l'utilisateur du système. Nous proposons également de faire un focus sur les résultats que nous avons obtenus suite à la simulation automatisée du comportement de sélection des acteurs pouvant exercer une compétence donnée.

3.2 Architecture globale

Il existe de très nombreux outils et frameworks pour développer des SMAs, que ces derniers soient distribués sous licence propriétaire ou libre. Ils s'appuient sur divers langages de programmation. Puisqu'il est impossible de les citer tous, nous nous contenterons de quelques exemples dont la documentation est facilement accessible sur Internet : *Swarm, Repeast, Mason, NetLogo, MadKit, Zeus*, etc. Le lecteur pourra se référer au site http://en.wikipedia.org/wiki/Comparison_of_agent-based_modeling_software pour obtenir un comparatif synthétique de nombre de ces outils. Notre choix s'est porté sur JADE compte tenu de son langage de programmation, Java, qui est cohérent avec la technologie GRAILS (Rocher, 2006) de la couche 1. De même, sa simplicité de déploiement (1 fichier jar unique), ses outils intégrés de manière native (débugging, traçage), et sa compatibilité avec les normes FIPA-ACL (FIPA, 2002) nous ont conduits à l'utiliser. JADE est distribué sous licence LGPL, donc entièrement Open Source et gratuit.

D'un point de vue concret, et plus qu'expérimental puisqu'en production au CG33, l'architecture logicielle de notre système de Bourse aux Compétences suit une décomposition en couches logicielles standardisées (voir figure 13) :

1. Une couche Modèle Vue Contrôleur (MVC) GRAILS, en front-office, dans le but de permettre aux utilisateurs d'échanger avec le système via une Interface Homme Machine (IHM) aux standards du Web.
2. Une couche de données qui héberge un Système de Gestion de Base de Données (SGBD) PostgreSQL (Stinson, 2006).
3. Une couche Système multi-agents (SMA), s'appuyant sur le Framework de développement JADE (Bellifemine, 1999), qui héberge nos agents compétence et que nous nommons « *moteur de R&D* ».

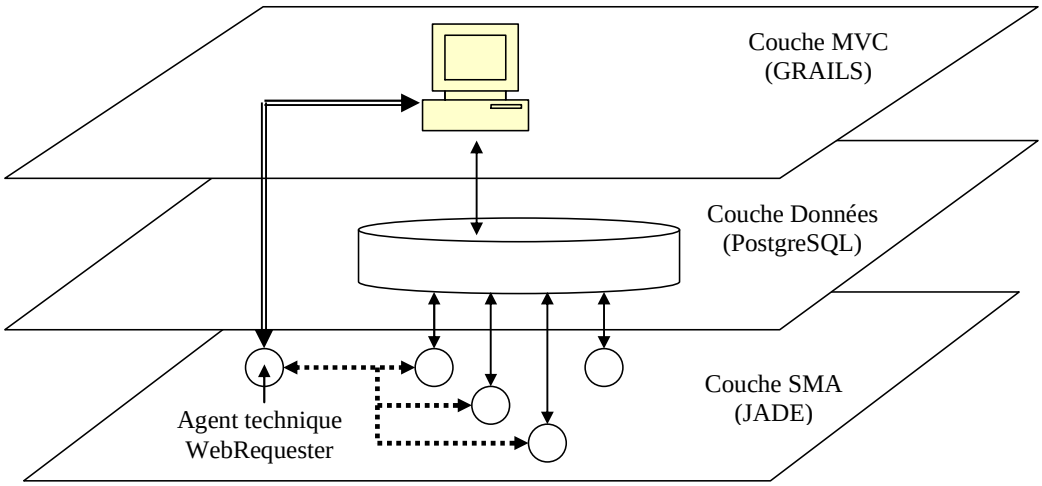


Figure 13 Architecture globale du système

La couche **N° 1** (MVC) se traduit par des pages Web dans un navigateur Internet. Elles assurent l'interface entre les utilisateurs (standards et/ou administrateurs) du système et la couche de données. Les échanges avec les agents compétences de la couche SMA s'effectuent par l'intermédiaire d'un agent technique dédié : « *WebRequester* ». Pour cette première couche, nous avons utilisé le framework de développement GRAILS en vigueur au CG33.

La couche **N° 2** (SGBD) assure la conservation physique des données tout au long du cycle de vie des objets (agents compétences, tables de paramétrage, etc.). Le modèle de données reste générique et adaptable à toute autre implémentation pour assurer la généricité de modélisation du domaine fonctionnel cible.

La couche **N° 3** (Moteur de R&D) se situe en « couche basse » (bottom) si l'on considère la partie MVC comme étant la partie haute (top) de l'architecture. S'appuyant sur le framework de développement JADE, la définition et la modélisation du moteur de R&D constituent l'essentiel des travaux de R&D faisant l'objet de cette thèse.

3.3 Couche N°1 : Application Web côté utilisateur

Lors de sa connexion à l'application Bourse aux Compétences, l'utilisateur accède à un portail généré par l'application GRAILS. Voir figure 14.

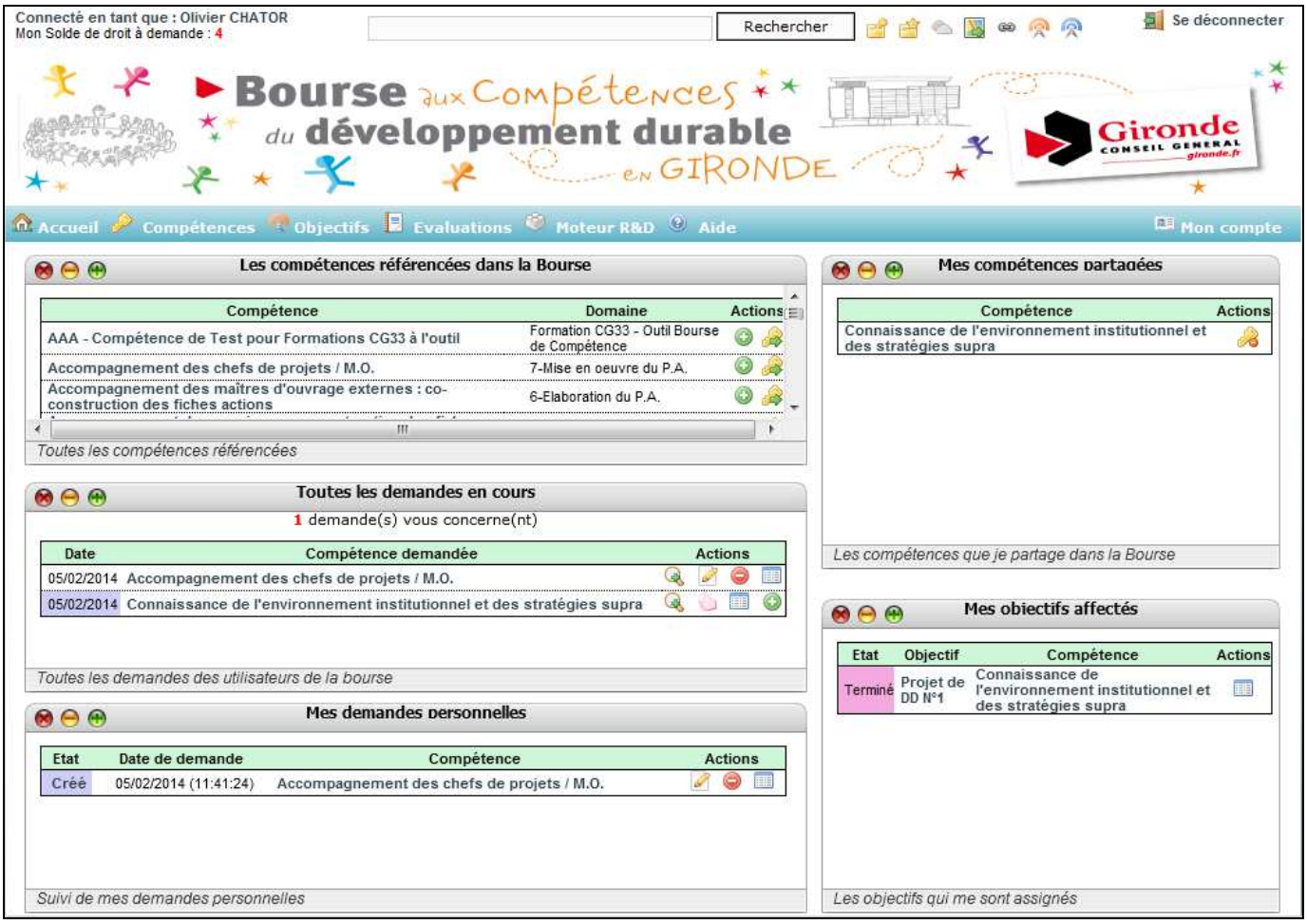


Figure 14 Portail utilisateur de la Bourse aux Compétences

De manière synthétique, cet écran se compose de 3 parties principales :

1. Un bandeau avec de gauche à droite : les informations de connexion utilisateur, une zone de recherche textuelle libre, des icônes de recherche par typologie (compétence, projets, identification de liens inter-compétences, etc.), puis un bouton de déconnexion.
2. Une barre de menus.
3. Une partie principale hébergeant 5 fenêtres :
   a) ***Les compétences référencées dans la bourse***. C'est le catalogue global de toutes les compétences qui sont disponibles et partageables. A titre d'information, il y en a 110 au CG33, ce qui signifie 110 Agents Compétence (ACs) dans le SMA en couche basse de l'architecture.
   b) ***Toutes les demandes en cours.*** Il s'agit de restituer la liste globale et exhaustive des demandes de compétence en cours, que l'utilisateur connecté puisse (veuille) ou non y répondre. L'intérêt de tout afficher est de montrer le dynamisme global du système et de stimuler les envies de l'utiliser.

c) ***Mes demandes personnelles***. L'ensemble des demandes de compétence effectuées par l'utilisateur connecté, avec leur statut de traitement courant (en cours, terminé, etc.) est disponible dans cette fenêtre.
d) ***Mes compétences partagées***. On retrouve ici la liste des compétences partagées par l'utilisateur connecté. Il s'agit de celles qu'il met à disposition des autres et pour lesquelles il est prêt à apporter sa participation.
e) ***Mes objectifs affectés***. Cette fenêtre est le pendant de celle intitulée « *Mes demandes personnelles* ». On trouve aussi une liste de demandes de compétence avec leur statut de traitement courant. Cependant, cette fois, il s'agit de celles que l'utilisateur connecté exerce pour le compte des autres et non des siennes. Le statut de traitement (en cours, suspendue, terminée, etc.) de chaque demande peut être mis à jour par l'utilisateur connecté.

D'un point de vue packaging technique, comme toute application Web GRAILS, cette composante de notre système se présente sous la forme d'un simple fichier war. Au CG33, nous l'avons déployé sur un serveur Apache Tomcat v 6.0.36, hébergé sur une machine virtuelle Linux RHEL-5 Entreprise 64bits.

3.4 Couche N°2 : Base de données

D'un point de vue purement technique, nous avons utilisé au CG33 un serveur de base de données PostgreSQL v 9.2.2. Ce serveur est hébergé sur une machine virtuelle Linux RHEL-5 Entreprise 64 bits. Le modèle de données a été présenté en détail plus haut dans ce mémoire (Voir figure 10[36]). L'implémentation physique est entièrement conforme au modèle proposé, il n'y a pas de particularités à signaler.

3.5 Couche N°3 : Le SMA (Moteur de R&D)

3.5.1 Généralités

L'hypothèse que nous cherchons à vérifier dans cette thèse est qu'une modélisation du concept de compétence sous la forme d'un agent logiciel autonome peut avoir de l'intérêt. Il s'agit réellement de construire un système apprenant pouvant déboucher sur une réelle gouvernance par les compétences et non plus par les acteurs au sein des projets. Nous cherchons à bâtir un réel modèle comportemental des agents compétence, basé sur différentes règles de fonctionnement qui ont été identifiées au travers de plusieurs documents d'étude et d'échanges fructueux entre les parties prenantes du projet (ENSC, Conseil Général, etc.). L'implémentation que nous proposons ici réside sur la constitution d'un moteur de R&D dont l'architecture doit être assez générique pour ne pas être cloisonnée à un domaine fonctionnel particulier (Développement Durable dans le cas de notre expérience au CG33).

[36] Voir paragraphe « *2.5.3 Modèle de données* » dans le chapitre 2

D'un point de vue purement technique, le moteur de R&D est une application écrite en langage Java, avec l'IDE Eclipse, reposant sur les librairies JADE (Bellifemine, 1999), et s'exécutant sur une machine virtuelle Linux RHEL-5 Entreprise 64 bits. On notera également que les comportements des agents sont implémentés de manière totalement normalisée au sein de plugins ***capteurs*** et ***effecteurs***. Ils sont décrits dans un paragraphe dédié plus loin.

3.5.2 Arborescence des composants logiciels

3.5.2.1 Vue global depuis la racine applicative ${SMA_ROOT}

Arborescence		Contenu des répertoires
smaServeur	→	Racine applicative ${SMA_ROOT}
.settings	→	Répertoire spécifique IDE Eclipse
classes	→	Classes Java compilées dans des packages sous *org.cg33.sma.**
conf	→	Fichiers globaux de configuration du moteur, des ACs et des logs
i18n	→	Fichiers de messages *fichier_xx.properties* avec *xx* code pays (fr, en…)
lib	→	Librairies Java utilisées dont *jade.jar*
logs	→	Fichiers de log générés: *SMALog.log* et *SMAErrorLog.log*
plugins	→	Racine des plugins comportementaux des ACs selon 2 typologies
effectors	→	Racine des effecteurs mis en œuvre par les agents
sensors	→	Racine des capteurs utilisés par les agents
script	→	Scripts de démarrage, arrêt, etc. du moteur, par système d'exploitation
simulations	→	Programmes de simulation
src	→	Fichiers source java du moteur dans des packages sous *org.cg33.sma.**
xml	→	Sauvegardes des fichiers XML du modèle comportemental

Tableau 5 : Structure arborescente de plus haut niveau du moteur de R&D

3.5.2.2 Focus sur le répertoire « ${SMA_ROOT}/classes »

Il contient toutes les classes Java compilées, packagées et visibles dans le tableau 6.

Arborescence		Contenu des répertoires
classes	→	Racine des classes Java compilées
org		
cg33		
sma	→	Racine du package du moteur
core	→	Classes **génériques** d'agent : *compétence*, *WebRequester*, etc.
data	→	Classes d'interfaçage DAO avec la base de données. Pour l'illustration, la table *agent*, se décline à travers les quatre fichiers *dao/AgentDAO.class*, *exception/Agent_agentException.class*, *impl/Agent_agentDAOImpl.class* et vo/*Agent.class*. factory contient des classes globales (*DAOFactory*.class*)
dao		
exception		
factory		
impl		
vo		
interfaces	→	Classe **générique** *SensorEffectorInterface.class*
lib	→	Librairies spécifiques au moteur : *SMAAgent*, *SMADaoInterface*, etc.
tools	→	Programmes de démarrage et d'arrêt global du moteur : *StartupAgents.class* et *StopAgents.class*

Tableau 6 : Structure arborescente du répertoire « classes »

Il est important de noter ici que :
Les classes dans ***core*** ont bien été conçues et implémentées de manière **générique**. Cela signifie qu'il n'y a pas autant d'implémentation différente que d'AC mais bien une généricité dans l'implémentation, par typologie d'agent. Pour être tout à fait transparent, ***core*** contient les éléments décrits dans le tableau 7.

Fichier compilé Java	Commentaire
BroadcastAgent$RequestPerformer.class BroadcastAgent.class Competence.class ObjectiveAgent.class SimulatorAgent.class TechnicalAgent.class WebRequesterAgent$TreatSignals.class WebRequesterAgent.class	On voit bien ici que l'on ne dispose que d'une seule et unique classe Java compilée par typologie d'agent logiciel dans notre moteur. Les classes dont le nom est de la forme *xxxx**$yyyy**.class* sont des classes embarquées dans les fichiers *xxxx.class*. C'est le cas de *BroadcastAgent$RequestPerformer.class* et *WebRequesterAgent$TreatSignals.class*

Tableau 7 : Contenu du répertoire « classes.org.cg33.sma.core »

3.5.2.3 Focus sur le répertoire « ${SMA_ROOT}/conf »

Ce répertoire contient tous les fichiers de configuration *globaux*, c'est-à-dire en dehors de ceux spécifiquement dédiés à chaque plugin capteur ou effecteur.

Fichiers de configuration	Commentaire
conf	
agents Competence.properties ObjectiveAgent.properties SimulatorAgent.properties TechnicalAgent.properties WebRequesterAgent.properties	On voit bien ici que l'on ne dispose que d'un seul et unique fichier de configuration par typologie d'agent. En parallèle du modèle comportemental implémenté en XML, c'est dans ces fichiers que peuvent être paramétrés les déclenchements de certains ***effecteurs***.
global AgentsMemoryTables.properties BehavioralRules.properties DatabaseServer.properties MailServer.properties Sma.properties Status.properties	Fichiers de configuration globaux hors agents dans le SMA. On y trouve par exemple la configuration des accès à la base de données (*DatabaseServer.properties*) ou du serveur de mail (*MailServer.properties*). Le paramétrage global du moteur s'effectue dans le fichier *sma.properties*.
log4j log4j.properties	Fichiers de configuration des logs

Tableau 8 : Structure arborescente du répertoire « conf »

3.5.2.4 Focus sur le répertoire « ${SMA_ROOT}/script »

Ce répertoire contient toutes les fichiers de configuration globaux, c'est-à-dire en dehors de ceux spécifiquement dédiés aux plugins capteurs et effecteurs (voir plus loin).

Scripts	Commentaire
script	
ux config.sh echoConfig.sh startAll.sh stopAll.sh	**<u>Sous Unix ou Linux</u>** Fichiers de paramétrage de l'environnement (config.sh), de lancement (startAll.sh) et d'arrêt (stopAll.sh) du moteur de R&D.
win32 config.bat startAll.bat stopAll.bat	**<u>Sous Windows</u>** Fichiers de paramétrage de l'environnement (*config.bat*), de lancement (*startAll.bat*) et d'arrêt (stopAll.bat) du moteur de R&D.

Tableau 9 : Structure arborescente du répertoire « script »

3.5.2.5 Focus sur le répertoire « ${SMA_ROOT}/src »

Ce répertoire contient toutes les classes Java sources. Selon leur typologie, elles sont réparties dans des packages visibles dans le tableau 10.

Arborescence	Contenu des répertoires	
classes	→	Racine des classes Java compilées
org		
cg33		
sma	→	Racine du package du moteur
core	→	Classes **<u>génériques</u>** d'agent : *compétence*, *WebRequester*, etc.
data	→	Classes d'interfaçage DAO avec la base de données
dao		
exception		
factory		
impl		
vo		
interfaces	→	Classe **<u>générique</u>** *SensorEffectorInterface.java*
lib	→	Librairies spécifiques au moteur : *SMAAgent*, *SMADaoInterface*, etc.
tools	→	Programmes de démarrage et d'arrêt global du moteur : *StartupAgents.java* et *StopAgents.java*

Tableau 10 : Structure arborescente du répertoire « src »

Rappel :
Comme nous l'avons déjà décrit au paragraphe « *3.5.2.2 Focus sur le répertoire ${SMA_ROOT}/classes* », les classes dans ***core*** ont bien été conçues et implémentées de manière **générique**. Cela signifie qu'il n'y a pas autant d'implémentation différente que d'AC mais bien une généricité dans l'implémentation, par typologie d'agent. Pour être tout à fait transparent, cette fois en tant que fichiers source, ***core*** contient les éléments décrits dans le tableau 11.

Fichier compilé Java	Commentaire
BroadcastAgent.java Competence.java ObjectiveAgent.java SimulatorAgent.java TechnicalAgent.java WebRequesterAgent.java	On voit bien ici que l'on ne dispose que d'une seule et unique classe Java par typologie d'agent logiciel dans notre moteur.

Tableau 11 : Contenu du répertoire « src.org.cg33.sma.core »

3.5.2.6 Focus sur le répertoire « ${SMA_ROOT}/plugins »

En référence du schéma présenté figure 1 (*Modèle multi-agent générique*), tout agent agit, induit des comportements, via la mise en œuvre de *capteurs* et d'*effecteurs*. Nous avons procédé par analogie dans notre implémentation et décliné chacun de ces éléments sous la forme de ***plugins comportementaux*** dans le répertoire ***${SMA_ROOT}/plugins*** du moteur de R&D. Le tableau 12 fournit la liste de ceux disponibles dans la version actuelle de nos développements.

Arborescence		Contenu des répertoires
plugins	→	**Racine des plugins comportementaux**
effectors	→	Racine des effecteurs des Agents
AgentLinkCreator	→	Effecteur de création de lien avec un autre AC
FindAgentsLinks	→	Effecteur de recherche de liens avec d'autres AC
FindBestActor	→	Effecteur de recherche d'acteur(s) pour un AC
FindObjectives	→	Effecteur de recherche d'objectifs
FindObjectiveSkillContribution	→	Effecteur de candidature à un projet pour un AC
FindObjectiveSkills	→	Effecteur de recherche des ACs pour un agent objectif
GenericEffector	→	Effecteur générique (squelette pour programmation)
GlobalShutdown	→	Effecteur d'arrêt global du SMA
Grow	→	Effecteur de croissance d'âge pour un AC
RulesCreator	→	Effecteur de génération nouvelle règle comportementale
Simulator	→	Effecteur de simulation
sensors	→	Racine des capteurs des Agents
DatabaseRuleChecker	→	Capteur de condition remplie en base de données
FindAgentsLinkSensor	→	Capteur de déclenchement de recherche de liens
GenericSensor	→	Capteur générique (squelette pour programmation)
GrowingSensor	→	Capteur de déclenchement de croissance d'âge d'AC
RulesEngine	→	Capteur d'analyse globale des règles comportementales

Tableau 12 : Structure arborescente du répertoire « plugins »

Tout plugin comportemental peut alors être codé, au sens informatique, selon un framework de développement normalisé que nous décrivons en détail au paragraphe suivant.

3.5.3 Framework de développement des plugins comportementaux

Chaque plugin comportemental, sous ***${SMA_ROOT}/plugins***, possède une arborescence normalisée. Exemple illustratif avec le plugin *AgentLinkCreator* :

Composants d'un plugin	Commentaire
AgentLinkCreator	
classes AgentLinkCreator.class ALC_UserParameter.class	Classes compilées du Plugin
conf AgentLinkCreator.properties	Fichier de configuration spécifique du plugin
i18n AgentLinkCreator_fr.properties	Fichier de langue spécifique du plugin
src AgentLinkCreator.java ALC_UserParameter.java	Fichiers source du plugin

Tableau 13 : Structure arborescente d'un plugin capteur ou effecteur

Tout plugin capteur ou effecteur implémente une interface commune (au sens java du terme): ***SMASensorEffector***. Pour pouvoir utiliser ensuite un plugin, il suffit d'agir sur, **et uniquement sur**, du paramétrage :

1. Rajouter le chemin d'accès à son répertoire classes dans la variable **SMA_CLASSPATH** positionnée dans le fichier ***${SMA_ROOT}/scripts/ux/config.sh*** ou ***${SMA_ROOT}/scripts/win32/config.bat*** selon le système d'exploitation sur lequel tourne le moteur de R&D
2. Pour l'agent ***xxxx*** qui va l'utiliser, il convient :
   - Soit de créer une nouvelle règle XML associée dans son modèle comportemental.
   - Soit de l'intégrer dans l'une des variables ***xxxxSensorsList*** ou ***xxxxEffectorsList***[37] du fichier de configuration global de l'agent ***${SMA_ROOT}/conf/agents/xxxx.properties***.

Par exemple, dans le fichier ***${SMA_ROOT}/conf/agents/competences.properties***, on trouve :

```
competenceSensorsList=RulesEngine;FindAgentsLinkSensor
competenceEffectorsList=102-FindObjectiveSkillContribution;200-FindBestActor;201-FindAgentsLinks
```

[37] Format de ce paramètre : signalNumber1-javaClassEffector1;...;signalNumberN-javaClassEffectorN

Le gros avantage d'une telle architecture à base de plugins est de pouvoir procéder aux développements des comportements des agents de manière externalisée, et de plus sous une forme normalisée. Tous les plugins doivent implémenter l'interface commune *SMASensorEffector*. Il n'y a jamais à intervenir sur le code interne des agents eux-mêmes. L'ajout de comportement s'effectue par simple adaptation de fichiers de configuration ou de règles XML comportementales dans le cas des ACs. On rappelle également que le code informatique de tout agent, par typologie, est générique (cf. paragraphe dédié sur le contenu du répertoire ***${SMA_ROOT}/src***).

3.5.4 Lancement, arrêt et mécanisme de traces du moteur de R&D

3.5.4.1 Lancement

Le démarrage du moteur de R&D s'effectue par exécution du script ***${SMA_ROOT}/script/ux/startAll.sh*** ou ***${SMA_ROOT}/win32.startAll.bat*** selon le système d'exploitation sur lequel il s'exécute. Le fichier de log ***${SMA_ROOT}/logs/SMALog.log*** contient alors des lignes telles que[38]:

```
09:37:07,658 INFO  INFO - StartupAgents - setup : Debut
09:37:07,724 INFO  INFO - StartupAgents - setup : Trying to start all Agents...
09:37:07,728 INFO   INFO - StartupAgents - setup : Starting Agent (1496 ->
Sensibilisation des cadres)
09:37:07,932 INFO   INFO - StartupAgents - setup : Starting Agent (1508 ->
Connaissance de l'environnement institutionnel et des stratégies supra)
……………………………………………………
09:37:11,919 INFO   INFO - StartupAgents - setup : Started Skill Agents number
(108)
09:37:11,923 INFO     INFO - StartupAgents - setup : Starting Agent
(WebRequesterAgent)
09:37:11,956 INFO     INFO - StartupAgents - setup : Starting Agent
(ObjectiveAgent)
09:37:11,988 INFO     INFO - StartupAgents - setup : Starting Agent
(BroadcastAgent)
09:37:11,999 INFO     INFO - StartupAgents - setup : Starting Agent
(SimulatorAgent)
09:37:12,040 INFO     INFO - StartupAgents - setup : Starting Agent
(TechnicalAgent)
09:37:12,073 INFO  INFO - StartupAgents - setup : Lancement de (113) Agent : OK
09:37:12,076 INFO  INFO - StartupAgents - setup : Fin
```

[38] En enlevant la date au début de chaque ligne (ex : `2014/02/07`)

3.5.4.2 Arrêt

L'arrêt du moteur s'effectue par exécution du script ***${SMA_ROOT}/script/ux/startAll.sh*** ou ***${SMA_ROOT}/win32.startAll.bat*** selon le système d'exploitation sur lequel il s'exécute. Le fichier de log ***${SMA_ROOT}/logs/SMALog.log*** contient alors des lignes telles que :

```
10:16:56,383 INFO  INFO - GlobalShutdown - execute : Global Shtudown in
progress...
&0:16:59,311 INFO  INFO - StopAgents - doStop : Résultat du shutdown plate-forme
(Shutdown de la plate-forme : OK)
10:16:59,324 INFO  INFO - SMAAgent - takeDown : Arret de l' Agent (Elaboration
méthodes d'animation)
10:16:59,329 INFO  INFO - SMAAgent - takeDown : Arret de l' Agent
(Sensibilisation / motivation des acteurs sur l'évaluation)
……................................
10:16:59,897 INFO  INFO - SMAAgent - takeDown : Arret de l' Agent
(Sensibilisation des agents)
```

3.5.5 Traces & fichiers de log

Tout au long de l'exécution du moteur de R&D, de son démarrage à son arrêt, les JADE et les agents de notre SMA fournissent des traces d'exécution au format standard log4j Java dans 2 fichiers :

- Pour les erreurs seules dans ***${SMA_ROOT}/logs/SMAErrorLog.log***.
- Pour toutes les traces dans ***${SMA_ROOT}/logs/SMALog.log***.

3.6 Mise en œuvre du cycle de vie

Dans le paragraphe dédié à la présentation des ACs, nous avons vu que ces derniers peuvent avoir 3 âges distincts : *enfance, adolescence* ou *maturité*. Lors du passage d'un âge à l'autre, on dit que l'AC « *grandit* » (ou évolue). Les transitions entre les âges sont assurées de manière automatique par des règles comportementales que l'on qualifiera de « *règles de croissance* ». Ces dernières se basent sur l'analyse de conditions environnementales propre à l'AC.

Par exemple, une des règles de croissance peut déclencher le passage automatique d'un âge à l'autre si l'on dispose de n évaluations de projets dans lesquels l'AC a été impliqué. Chaque AC étant autonome, ces règles peuvent bien entendu varier et être adaptées pour chacun d'eux. Dans tous les cas, elles font partie de l'ensemble des règles, dites « *éducatives de base* » que l'on fournira à l'AC au moment de sa création dans le système. Le capteur générique *RulesEngine* est chargé de l'interprétation de ces

règles. Concernant spécifiquement la gestion du cycle de vie, et dans l'état actuel de nos travaux, les règles de croissance sont les suivantes :

```
<ruleGroup description="Growing Policy" weight="1">

   <rule description="Growing from Youth to Teenage" mandatory="1" weight="1">
      <when description="Growing Conditions" sensor="GrowingSensor"
          params="unitaryEvaluationNumber=2" result="$evaluationsNumber1" operator="EQ"
          value="3">
          <do description="Grow to teenage" effector="Grow"
             params="from=youth|to=teenage" />
      </when>
   </rule>

   <rule description="Growing from Teenage to Mature" mandatory="1" weight="1">
      <when description="Growing Conditions" sensor="GrowingSensor"
          params="unitaryEvaluationNumber=2" result="$evaluationsNumber2" operator="EQ"
          value="10">
          <do description="Grow to mature" effector="Grow"
             params="from=teenage|to=mature" />
      </when>
   </rule>

</ruleGroup>
```

Tout au long de son cycle de vie, et via le capteur *GrowingSensor*, l'AC recherche dans son environnement le nombre d'évaluations disponibles pour les projets auxquels il a participé. Si les conditions de transition d'un âge à l'autre sont remplies, l'effecteur *Grow* est alors appelé avec les paramètres adéquats et modifie en conséquence la valeur de la colonne *age_id* de la table *agent* pour l'**Enr$_i$** de l'AC courant. Bien entendu, les conditions de croissance peuvent être adaptées à la demande et au contexte, voir même personnalisées pour chaque AC grâce à la mise à niveau du champ *rules* de l'**Enr$_i$** courant dans la table *agent*. Les modifications portent alors sur le paramétrage des règles de croissance existantes, ou bien s'effectuent grâce à la création de nouvelles règles et de capteurs afférents. Il s'agit toujours, au final, de vérifier via les capteurs utilisés si les conditions environnementales de déclenchement de l'effecteur *Grow* sont réunies.

3.7 Mise en œuvre du modèle comportemental

Compte tenu de leur nature différente, nous avons fait le choix dans l'implémentation de notre système, de bien distinguer deux types de situation induisant un déclenchement de comportements. Pour tout AC :

1. Tout comportement à déclencher suite à la survenue d'un évènement dont l'utilisateur humain du système est à l'origine via l'utilisation de l'IHM GRAILS, sera déclaré sous la forme d'un couple *signal-effecteur* dans le tag de configuration *competenceEffectorsList* du fichier de configuration global ***${SMA_ROOT}/conf/agents/competence.properties*** des ACs.

2. Tout comportement à déclencher suite à la survenue d'une modification de la situation environnementale de l'AC lui-même au sein du SMA, fera l'objet de la création d'une règle associée dans le modèle comportemental XML.

Par exemple, dans le premier cas, le déclenchement de la restitution d'une liste d'acteurs ou de compétences à embarquer dans un nouveau projet n'est à réaliser effectivement que suite à une action volontaire de demande de ces listes de la part de l'utilisateur humain du système. Dans le second cas, le fait de faire grandir un AC de l'*enfance* à l'*adolescence* est lié à une évolution de son univers dans le SMA et non issu d'une demande de l'utilisateur humain du système. En conséquence, dans ce dernier cas, on a privilégié l'utilisation de règles de croissance dans le modèle comportemental XML, analysées de manière périodique et récurrente par l'effecteur générique *RulesEngine* de l'AC tout au long de sa vie dans le SMA. Le choix de séparer les deux mécanismes de lancement d'exécution de comportements se justifie pleinement compte tenu de la nature différente des évènements à traiter et de la non périodicité de ceux appartenant au premier cas de figure.

Cependant, il existe un cas particulier que l'on pourrait qualifier de « réparti » sur les deux configurations précédentes. Il s'agit de la recherche de la liste des liens qu'un AC peut avoir avec les autres. La création de cette liste peut effectivement :

1. Etre issue d'une demande active d'un utilisateur humain du système via l'IHM GRAILS.
2. Etre issue du déclenchement d'un comportement récurrent permettant à l'AC de réaliser l'un de ses buts. Pour rappel, comme mentionné au paragraphe « *définition* » des agents compétence au chapitre 2, les 4 buts d'un AC sont :
   a) Constituer sa propre définition.
   b) Restituer une liste d'acteurs humains capables de le concrétiser au sein d'un projet.
   c) Etablir des liens (relations) avec d'autres agents.
   d) Se porter candidat à une participation à de nouveaux projets.

Les détails d'implémentation du cas particulier de création des liens inter AC sont abordés dans un paragraphe dédié plus bas dans ce document. Ce cas est d'importance compte tenu du fait que ses mécanismes d'apprentissage conduisent à la mise à jour dynamique du modèle comportemental XML de l'AC lui-même. De manière classique dans nos travaux, ces mises à jour sont soumises à validation humaine ou non selon l'âge de l'AC.

3.8 Mémorisation

En référence de ce que nous avons vu dans le paragraphe « *RAPC dynamique dans un contexte SMA*», la littérature propose différents mécanismes de mémorisation. Entre autre, les travaux de Shank (Shank, 1982) proposent une « théorie de la mémoire

dynamique ». Les hypothèses principales formulées par cet auteur sont que certains processus sont indissociables (se souvenir, comprendre, expérimenter et apprendre) et que l'expérience influe sur l'évolution de la mémoire. Les mécanismes de mémorisation des ACs suivent cette affirmation. L'évolution de la mémoire, et en particulier des souvenirs qui y sont stockés, est liée au stockage des opérations de validation. Cette validation, pour rappel, est effectuée par les acteurs humains si l'AC est *enfant* ou *adolescent* ou bien par l'AC lui-même à l'âge de la *maturité*. Il existe dans le modèle une table *memory_Ci* pour chaque AC. Ces *memory_Ci* assurent l'étanchéité des souvenirs de chaque AC et concrétisent le souhait de n'offrir qu'une vue partiel de l'environnement global à chacun d'eux. On notera qu'au moment de sa « *naissance* » dans le SMA, chaque AC créé sa table *memory_Ci* dédiée si elle n'existe pas. Reprenons le tableau 4 déjà vu plus haut dans le tableau 14.

Champ	Description	Exemple de contenu
code	Identifiant unique	1
evt_date	Date de création	2013-09-30
evt_id	Type d'évènement sur le souvenir (1 = création, 2 = validation humaine)	1
agent_id	Identifiant de l'agent	3192
parameters_in	Paramètres en entrée	stepNumber=1;useOnlyResponses=false;controller=RDEngine;resultsNumber=1;signalCode=200;useMemory=true;action=callRdEngineForBestActor;agentId=3192
decision_string	Décision prise	<reponses><reponse><num>1</num><memoryCode>#</memoryCode><userid>3152</userid><firstname>Xavier</firstname><lastname>PASTEAU</lastname><service>XPS</service></reponse></reponses>
human_validated	***Validation humaine***	***TRUE or FALSE***
comment	Commentaire décrivant à quel type de comportement de l'agent s'applique le souvenir	Trouver meilleur acteur

Tableau 14 : Contenu d'un enregistrement de la table mémoire d'un agent

Dans notre implémentation, les mécanismes de mémorisation interviennent lors de l'exercice de 3 comportements de sélection des acteurs, création de liens inter ACs et création d'un nouveau projet par l'utilisateur. Informatiquement parlant, au sein du code Java d'un effecteur étendant l'interface *SMASensorEffector*, c'est la méthode *storeInMemory()* avec ses paramètres qui est utilisée pour stocker et mettre à jour les validations de souvenirs dans la table *memory_Ci* de l'AC courant.

3.9 Comportement N°1 : s'auto définir

Pour s'auto définir, une compétence (donc un AC) identifie l'ensemble des compétences élémentaires CE_t qui la composent. Réciproquement, une compétence élémentaire peut participer à la définition d'une ou de plusieurs compétences. Ainsi, la définition d'une compétence évolue. Le pendant de cette affirmation est que la définition même de l'AC est dynamique. L'auto définition est un comportement « simulé artificiellement». Il n'y a pas de code informatique, ni de règle comportementale XML associées. Toutefois, d'un point de vue théorique, on peut dire qu'il y a bien apprentissage, même si celui-ci est réparti. Il est effectivement, dans les faits, issu des apprentissages réalisés respectivement par le comportement de sélection des acteurs et celui d'établissement de liens avec les autres ACs.

3.10 Comportement N°2 : sélectionner des acteurs

3.10.1 Introduction

Ce comportement n'est pas traduit sous la forme d'une règle dans le modèle comportemental XML global de l'AC. Il l'est dans le tag de configuration *competenceEffectorsList* du fichier de configuration globale ***${SMA_ROOT}/conf/agents/competence.properties*** des agents compétence. Il s'agit en effet d'un comportement provoqué suite à demande active de la part de l'utilisateur humain du système à travers son IHM GRAILS.

3.10.2 Ecran de paramétrage de la demande utilisateur

La demande de l'utilisateur s'effectue via l'écran présenté figure 15. La compétence à interroger est sélectionnée dans une liste déroulante. Le positionnement de valeurs pour les autres paramètres (généraux ou compétences élémentaires) est facultatif. Des valeurs par défaut seront appliquées par le moteur de R&D si nécessaire.

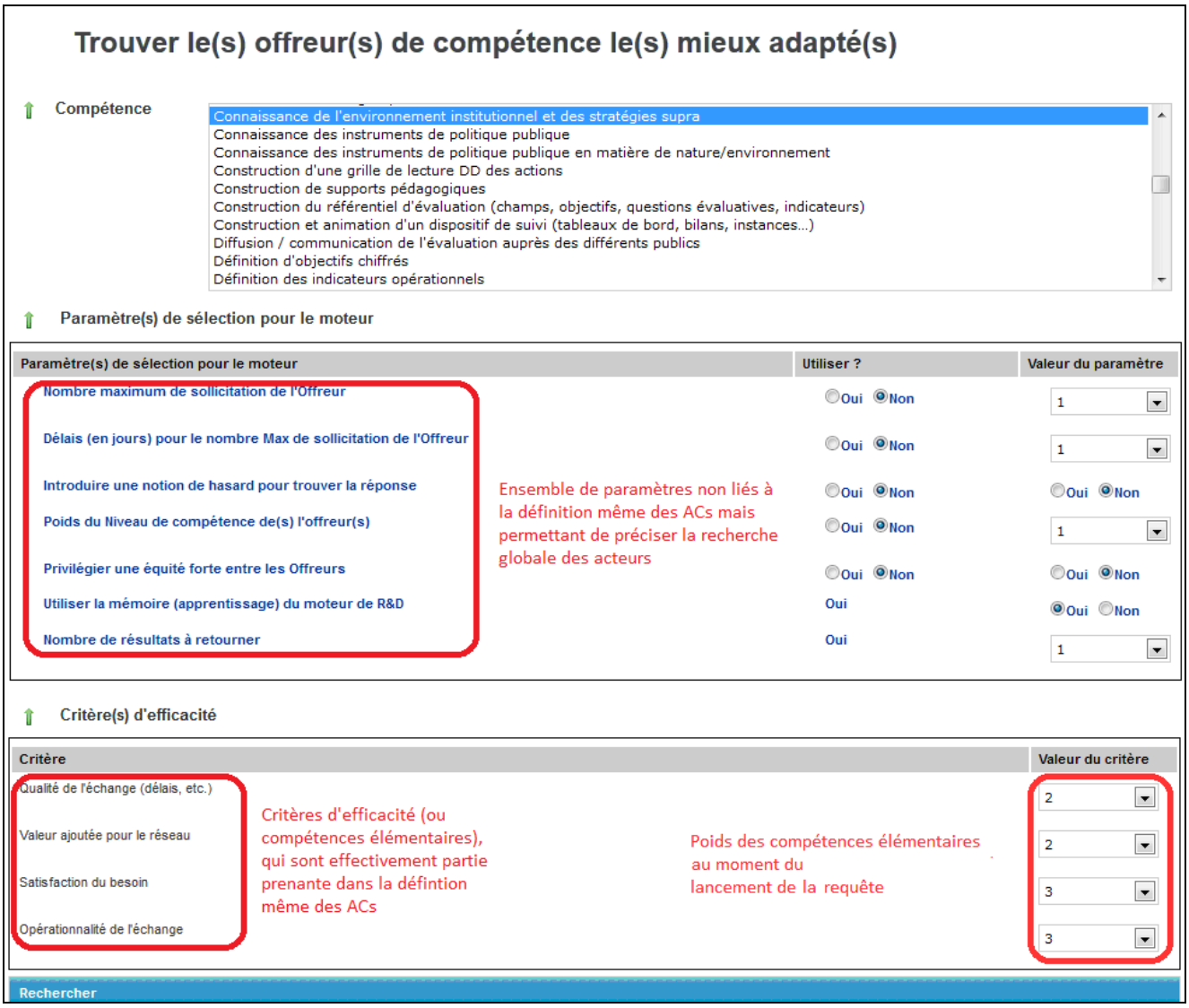


Figure 15 Sélection des acteurs : paramétrage de la demande utilisateur

3.10.3 Echanges de flux

La figure 16 propose une vue schématique des échanges de flux, en 2 étapes, pour ce comportement :

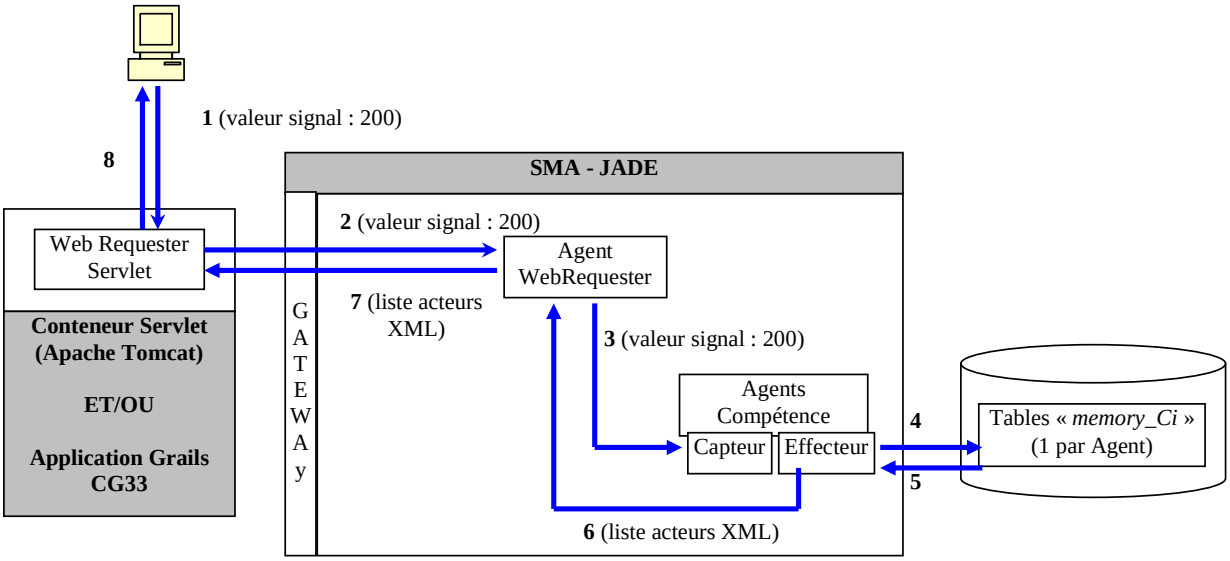


Figure 16 Sélection des acteurs : échanges de flux

Pour établir un parallèle avec les fichiers de configuration déjà détaillés plus haut, on notera que la valeur **200** du signal correspond au déclenchement de l'effecteur *FindBestActor* qui est déclaré dans le tag de configuration *competenceEffectorsList*. De manière plus détaillée, selon l'étape d'échange[39], les flux identifiés lors de la mise en œuvre de ce comportement sont présentés dans le tableau 15.

N°	Etape 1 : exécution de la demande de liste	Etape 2 : validation de la liste des acteurs
1	**Exemple de paramètres vers moteur de R&D** signalCode=200 stepNumber=1 agentId=1508 useMemory=true useOnlyResponses=false resultsNumber=1 criteriasWeights=305,0.3:269,0.3:2441,0.2:47,0.2	**Exemple de paramètres vers moteur de R&D** signalCode=200 stepNumber=2 agentId=1508 humanDecision=true memoryCode=1
2	Transmission (standard JADE) de la demande entre *WebRequester Servlet* et *WebRequester*	Transmission (standard JADE) de la validation entre *WebRequester Servlet* et *WebRequester*

[39] Transmission de la demande initiale de l'utilisateur (étape 1) ou de sa décision de validation (étape 2)

	Agent	*Agent*
3	Transmission de la demande à l'agent compétence interrogé	Transmission de la validation à l'agent compétence
4	Extraction des souvenirs antérieurs validés en mémoire Calcul des nouveaux poids des compétences élémentaires	Validation du souvenir
5	Extraction de la liste des acteurs Formatage de la liste des acteurs sous la forme d'un flux XML	Sans Objet
6	Renvoi du flux XML des acteurs vers *WebRequester Agent*	Renvoi du flux XML avec le résultat de validation vers *WebRequester Agent*
7	Transmission (standard JADE) de la liste dans le flux XML entre *WebRequester Agent* et *WebRequester Servlet*	Transmission (standard JADE) du flux XML avec le résultat de validation entre *WebRequester Agent* et *WebRequester Servlet*
8	Affichage de la liste des acteurs au format HTML	Affichage du résultat de validation au format HTML

Tableau 15 : Sélection des acteurs : tableau détaillé des flux échangés

On notera que le paramètre *criteriasWeights* contient la liste des couples (identifiant de compétence élémentaire, poids demandé pour cette demande) définis par l'utilisateur au moment de l'expression de sa requête. Les poids sont normalisés à la valeur **1**.

3.10.4 Apprentissage

Il ne s'agit pas ici de reprendre en détail tout l'algorithme qui a déjà été présenté dans le chapitre de modélisation. Le code informatique écrit en langage Java ne fait, en effet, que le traduire. On notera toutefois l'enchainement de certaines étapes de traitement nécessaires à l'apprentissage.

Etape 1 : Suite à la constitution de la liste d'acteurs répondants aux critères renseignés par l'utilisateur lors de l'émission de sa demande, l'AC qui est interrogé créé un nouvel enregistrement dans sa table mémoire *memory_Ci*. . Cet enregistrement correspond à un nouveau souvenir et sa colonne *human_validated* est positionnée à la valeur *faux*.

Etape 2 : La liste des acteurs fournis par l'AC est renvoyée à l'utilisateur (flux 8, étape 1). Cette liste est considérée comme étant la décision de l'AC suite à la demande utilisateur. Aux âges d'enfance et d'adolescence, une validation humaine doit être acquise[40]. C'est le rôle rempli par des écrans comme celui présenté figure 17.

[40] Rappel : elle est considérée comme acquise par défaut si l'AC est mature

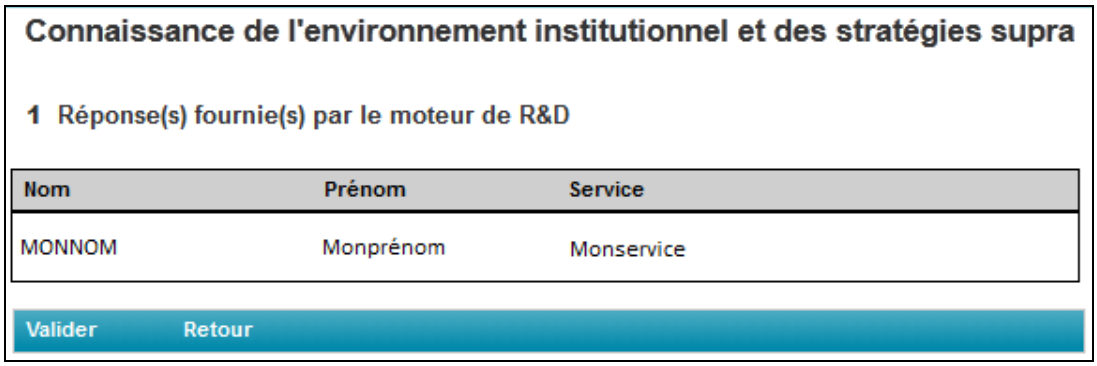


Figure 17 Sélection des acteurs : écran de validation de la décision de l'AC

<u>Etape 3</u> : Si la validation humaine est prononcée, l'AC met à jour le souvenir correspondant à la prise de décision courante. Dans la table mémoire *memory_Ci*, le champ *human_validated* passe alors de la valeur *faux* à la valeur *vrai*.

L'enchainement des 3 étapes que nous venons de décrire permet de donner un sens réel à l'apprentissage. Grâce aux mécanismes de validation humaine, il fait sens dans le contexte fonctionnel métier où évolue l'AC. Selon l'algorithme de sélection des acteurs, les poids de chaque compétence élémentaire sont recalculés, à chaque demande utilisateur, en tenant compte des souvenirs précédemment validés. D'un point de vue implémentation, on travaille à partir des valeurs des poids de chaque critère d'efficacité disponibles. Ces derniers sont pris dans les différentes valeurs du paramètre *criteriasWeights* des souvenirs qui ont été validés antérieurement, c'est-à-dire ayant une valeur du champ *human_validated* positionnée à *vrai*. Pour tout AC, grâce à l'acquisition de plus en plus de souvenirs validés, nous constatons que la valeur des poids recalculés de chaque compétence élémentaire converge. Ce phénomène sera également démontré plus loin via l'utilisation d'un simulateur comportemental dédié à cette sélection des acteurs.

3.11 Comportement N°3 : identifier des liens inter ACs

3.11.1 Introduction

Ce comportement permet de restituer des listes de liens qu'un AC peut avoir avec les autres au sein du SMA. Il s'agit de l'un des quatre buts que cherche à atteindre tout AC tout au long de sa vie. Ce comportement peut cependant être activé selon deux situations distinctes :

1. Suite à une demande explicite d'un utilisateur humain du système via l'IHM GRAILS.
2. Suite au déclenchement récurrent d'un comportement, interne à l'AC.

3.11.2 Cas N°1 : Demande explicite de l'utilisateur

3.11.2.1 Ecran de paramétrage de la demande utilisateur

La demande s'effectue de manière explicite via l'écran présenté figure 18. La compétence pour laquelle on souhaite rechercher les liens est sélectionnée dans une liste déroulante. Il est possible de modifier la valeur du seuil du coefficient de proximité minimal, la valeur de base étant celle considérée par défaut.

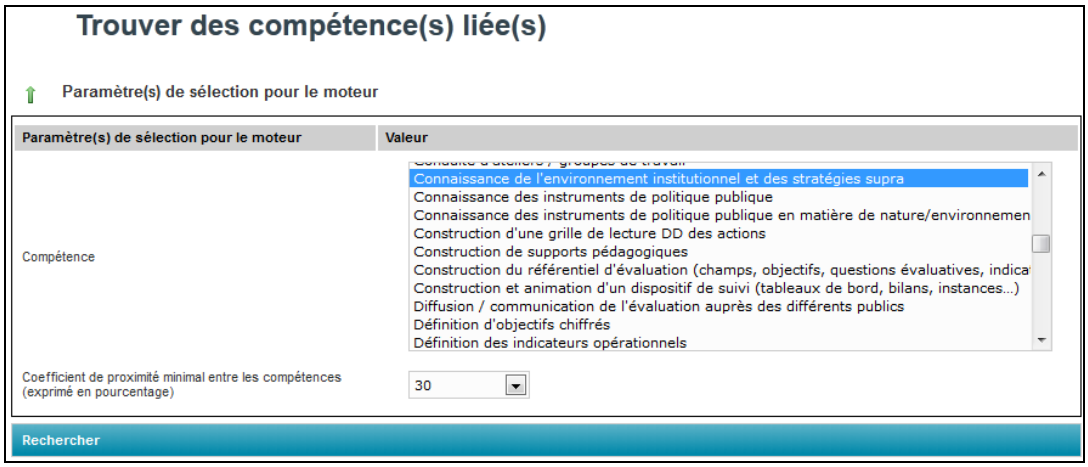


Figure 18 Recherche de liens inter ACs : cas 1 : paramétrage de la demande utilisateur

3.11.2.2 Echanges de flux

La figure 19 propose une vue schématique des échanges de flux, en 2 étapes :

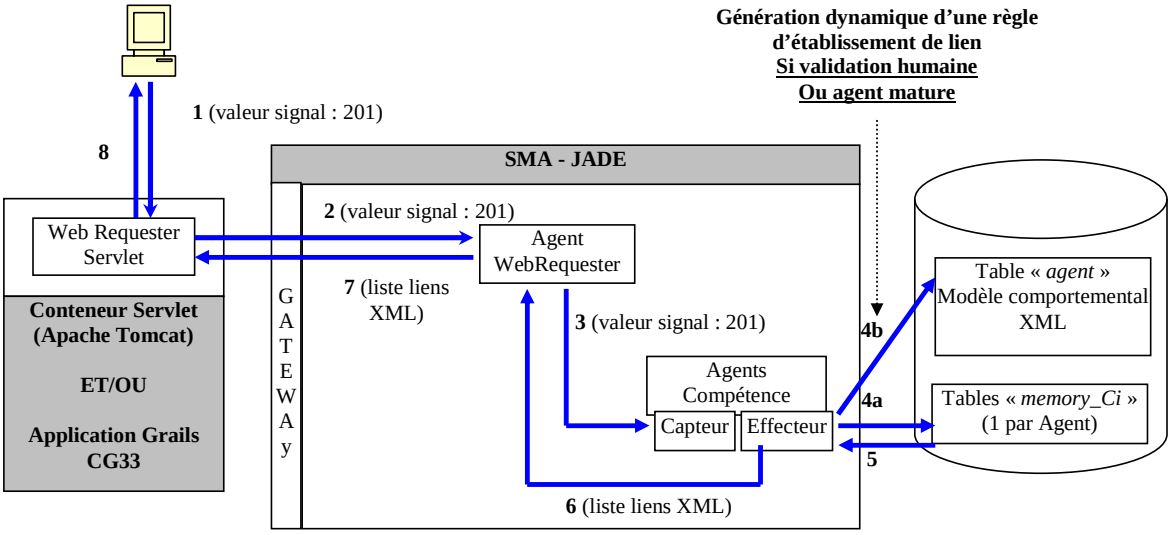


Figure 19 Recherche de liens inter ACs : cas 1 : échanges de flux

Pour établir un parallèle avec les fichiers de configuration déjà détaillés plus haut, on notera que la valeur **201** du signal correspond au déclenchement de l'effecteur *FindAgentsLinks* qui est déclaré dans le tag de configuration *competenceEffectorsList*. De manière plus détaillée, selon l'étape d'échange[41], les flux identifiés lors de la mise en œuvre de ce comportement sont présentés tableau 16.

N°	Etape 1 : exécution de la demande de liste	Etape 2 : validation de liens
1	**Exemple de paramètres vers moteur de R&D** signalCode=201 stepNumber=1 agentId=1508 minPercentValue=30 fromWebInterface =true	**Exemple de paramètres vers moteur de R&D** signalCode=201 stepNumber=2 agentId=1508 fromWebInterface =true keepMemoryValidated=true agentLevel=true linkedCodes=1507 weights=100.0 skillMemoryCodes=2 humanDecision=true memoryCodes=2
2	Transmission (standard JADE) de la demande entre *WebRequester Servlet* et *WebRequester Agent*	Transmission (standard JADE) de la validation entre *WebRequester Servlet* et *WebRequester Agent*
3	Transmission de la demande à l'agent compétence interrogé	Transmission de la validation à l'agent compétence
4a	Inscription en mémoire d'un nouveau souvenir avec la liste des paramètres d'entrée et la colonne *human_validated* positionnée à *faux*.	Validation du souvenir
4b	Sans Objet	Suite à validation humaine : génération et insertion dans le modèle comportemental global de l'AC d'une nouvelle règle de création de lien
5	Extraction de la liste des ACs liés Formatage de la liste des liens sous la forme d'un flux XML	Sans Objet
6	Renvoi du flux XML des liens vers *WebRequester Agent*	Renvoi du flux XML avec le résultat de validation vers *WebRequester Agent*
7	Transmission (standard JADE) de la liste dans le flux XML entre *WebRequester Agent* et *WebRequester Servlet*	Transmission (standard JADE) du flux XML avec le résultat de validation entre *WebRequester Agent* et *WebRequester Servlet*
8	Affichage de la liste des liens proposés au format HTML	Affichage du résultat de validation au format HTML

Tableau 16 : Recherche de liens inter ACs : cas 1 : échanges de flux

[41] Transmission de la demande initiale de l'utilisateur (étape 1) ou de sa décision de validation (étape 2)

On notera que pour l'étape N°2 concernant la validation des liens, de nombreux paramètres sont fournis. Ils permettent, via par exemple *linkedCodes*, de spécifier finement quels sont les liens à valider ou non parmi tous ceux qui sont proposés par l'AC. En particulier, une valeur à *faux* du paramètre *agentLevel* spécifie que les validations de lien prononcées ne se font pas agent par agent mais sont prononcées à un niveau global, c'est-à-dire tous les liens proposés. Cette décision est d'importance puisque c'est elle qui induit la génération automatique d'une règle dans le modèle comportemental de l'AC.

3.11.3 Cas N°2 : identification récurrente des liens

3.11.3.1 Introduction

Ici, la création des liens inter-agents utilise un capteur : *FindAgentsLinkSensor*. Ce dernier est déclaré dans le tag *competenceSensorsList* du fichier de configuration globale ***${SMA_ROOT}/conf/agents/competence.properties***. Un tel capteur analyse son environnement périodiquement et a pour objectif d'identifier des propositions de liens à établir. Ces propositions sont renvoyées vers l'animateur de la Bourse dans le but de recevoir une validation si l'AC est à l'âge d'*adolescence*.

3.11.3.2 Echanges de flux

De manière récurrente, l'AC recherche des liens inter agents. Lorsqu'il en trouve, il génère automatiquement des envois de mails contenant ses propositions à destination de l'animateur de la Bourse aux Compétences. Un exemple d'e-mail, pris dans le jeu de rugby, est fourni figure 20.

La compétence *Ailier* vous fait *9* propositions

- Lancer interface graphique

Proposition *1*

- **Sur la(les) similitude(s) suivante(s) :**
  - Domaine de compétence
- **1 compétence(s) semble(nt) pouvoir être liée(s) :**

  Centre 44,65 %

Proposition *2*

- **Sur la(les) similitude(s) suivante(s) :**
  - Lien compétence - critère d'efficacité : code de compétence
  - catégorie de critère d'efficacité : code
  - critère d'efficacité : code catégorie de critère
  - critère d'efficacité : code
  - Lien compétence - critère d'efficacité : code de critère
  - catégorie de critère d'efficacité : description
- **7 compétence(s) semble(nt) pouvoir être liée(s) :**

  Ouverture 33,56 %
  Demi de mélée 40,94 %
  Troisième Ligne 100,00 %
  Centre 44,65 %
  Pilier 95,08 %
  Talonneur 37,25 %
  Deuxième Ligne 64,32 %

Figure 20 Recherche de liens inter ACs : cas 2 : mail avec propositions

La figure 21 propose une vue schématique des échanges de flux, en 2 étapes :

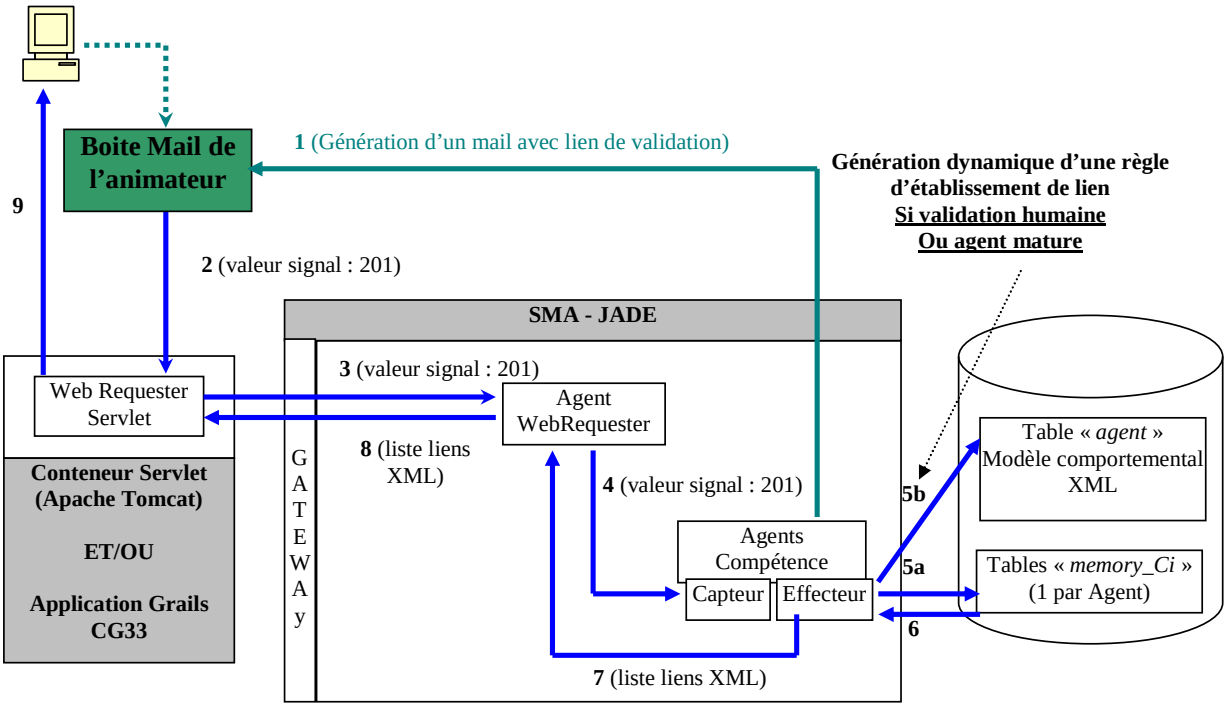


Figure 21 Recherche de liens inter ACs : cas 2 : échanges de flux

Comme dans le premier cas d'utilisation, pour établir un parallèle avec les fichiers de configuration déjà détaillés plus haut, on notera que la valeur **201** du signal correspond au déclenchement de l'effecteur *FindAgentsLinks* qui est déclaré dans le tag de configuration *competenceEffectorsList*. De manière plus détaillée, selon l'étape d'échange[42], les flux identifiés lors de la mise en œuvre de ce comportement sont présentés tableau 17.

N°	Etape 1 : accès écran de validation des liens	Etape 2 : validation de liens
1	Chaque AC, via son capteur *FindAgentsLinkSensor*, analyse son environnement et en déduit des propositions de lien qu'il expédie dans un mail vers l'animateur de la Bourse pour analyse et validations éventuelles.	Sans objet
2	**Exemple de paramètres vers moteur de R&D** signalCode=201 stepNumber=1 agentId=1508 minPercentValue=30	**Exemple de paramètres vers moteur de R&D** signalCode=201 stepNumber=2 agentId=1508 fromWebInterface =true

[42] Transmission de la demande initiale de l'utilisateur (étape 1) ou de sa décision de validation (étape 2)

	fromWebInterface =true	keepMemoryValidated=true agentLevel=true linkedCodes=1507 weights=100.0 skillMemoryCodes=2 humanDecision=true memoryCodes=2
3	Transmission (standard JADE) de la demande entre *WebRequester Servlet* et *WebRequester Agent*	Transmission (standard JADE) de la validation entre *WebRequester Servlet* et *WebRequester Agent*
4	Transmission de la demande à l'agent compétence interrogé	Transmission de la validation à l'agent compétence
5a	Inscription en mémoire d'un nouveau souvenir avec la liste des paramètres d'entrée et la colonne *human_validated* positionnée à *faux*.	Validation du souvenir
5b	Sans Objet	Suite à validation humaine : génération et insertion dans le modèle comportemental global de l'AC d'une nouvelle règle de création de lien
6	Extraction de la liste des ACs liés Formatage de la liste des liens sous la forme d'un flux XML	Sans Objet
7	Renvoi du flux XML des liens vers *WebRequester Agent*	Renvoi du flux XML avec le résultat de validation vers *WebRequester Agent*
8	Transmission (standard JADE) de la liste dans le flux XML entre *WebRequester Agent* et *WebRequester Servlet*	Transmission (standard JADE) du flux XML avec le résultat de validation entre *WebRequester Agent* et *WebRequester Servlet*
9	Affichage de la liste des liens proposés au format HTML	Affichage du résultat de validation au format HTML

Tableau 17 : Recherche de liens inter ACs : cas 2 : échanges de flux

On notera que pour l'étape N°2 concernant la validation des liens, de nombreux paramètres sont fournis. Ils permettent, via par exemple *linkedCodes*, de spécifier finement quels sont les liens à valider ou non parmi tous ceux qui sont proposés par l'AC. En particulier, une valeur à *faux* du paramètre *agentLevel* spécifie que les validations de lien prononcées ne se font pas agent par agent mais sont prononcées à un niveau global, c'est-à-dire tous les liens proposés. Cette décision est d'importance puisque c'est elle qui induit la génération automatique d'une règle dans le modèle comportemental de l'AC.

3.11.4 Apprentissage

Comme pour le comportement de sélection des acteurs que nous avons décrit plus haut, il ne s'agit pas ici de reprendre en détail tout l'algorithme déjà présenté dans le chapitre de modélisation. Le code informatique écrit en langage Java ne fait, en effet, que le

traduire. On notera toutefois l'enchainement de certaines étapes de traitement nécessaires à l'apprentissage.

Etape 1. Suite à la constitution de la liste de liens proposés, l'AC a créé un nouvel enregistrement dans sa table mémoire *memory_Ci*. . Cet enregistrement correspond à un nouveau souvenir et sa colonne *human_validated* est positionnée à la valeur *faux*.

Etape 2. Une liste des propositions de création de liens est fournie par l'AC puis renvoyée à l'utilisateur. Cette liste est considérée comme étant la décision de l'AC suite à la demande utilisateur ou bien au traitement récurrent de recherche. Aux âges d'enfance et d'adolescence, une validation humaine doit être acquise[43]. C'est le rôle rempli par l'écran présenté figure 22.

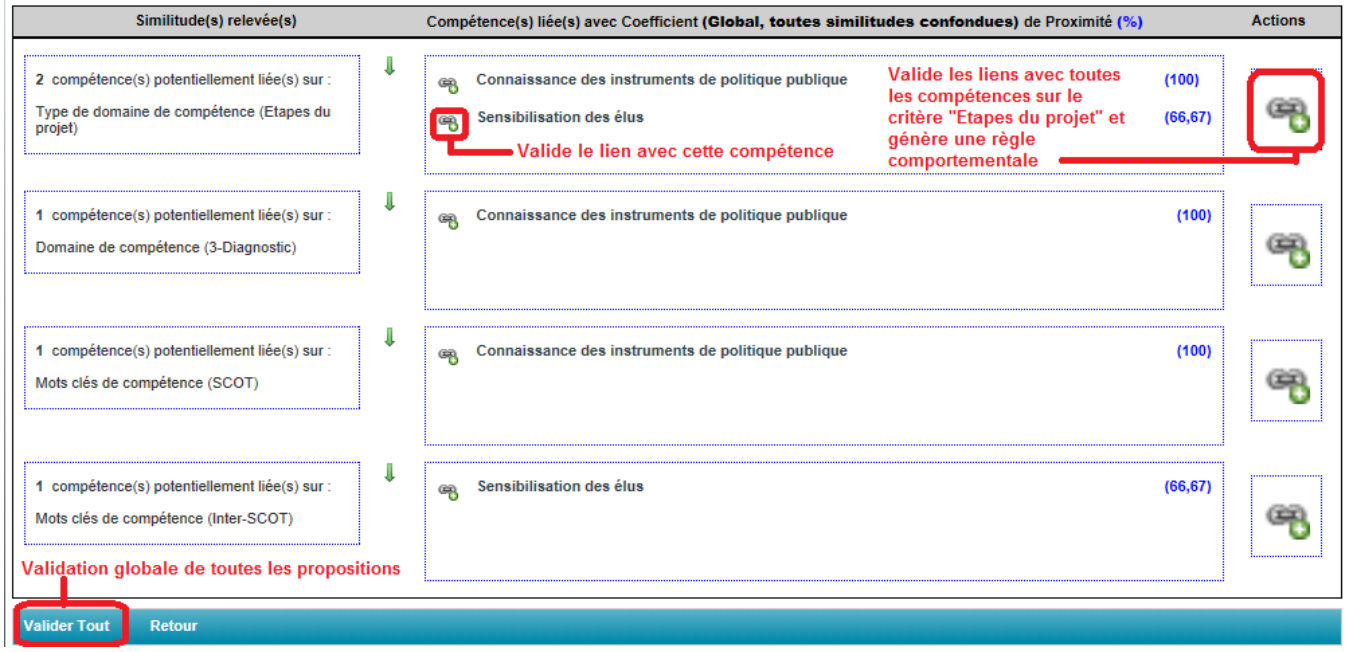


Figure 22 Recherche de liens inter ACs : écran de validation des propositions

La validation humaine intervient sur tout ou partie des propositions et peut se situer à 3 niveaux distincts :

1. ***Niveau compétence*** : on considère uniquement certains liens sur le critère de similitude donné comme pertinents.
2. ***Niveau proposition*** : on valide tous les liens de la proposition sur le critère de similitude donné.
3. ***Niveau global*** : on valide tous les liens de toutes les propositions.

Une validation au niveau proposition induit pour le moteur la génération d'une règle comportementale. Ainsi, de manière dynamique et récurrente, si de futures compétences apparaissent comme étant liées sur le critère de similitude et que l'on a validé

[43] Rappel : elle est considérée comme acquise par défaut si l'AC est mature

globalement la proposition, il y aura génération automatique de lien par le moteur (cf. étape 3). Il y a donc bien apprentissage par validation humaine tant que la compétence est à l'âge d'adolescence comme déjà décrit dans notre modèle théorique. A l'âge de maturité, les liens s'établissent de manière automatique, sans action de validation humaine.

Etape 3. L'apprentissage consiste à générer dynamiquement des règles d'établissement de lien avec les autres agents et à les « injecter » au sein même du modèle comportemental de l'AC. Il y a ici un point fondamental à retenir :

Toute règle permettant la génération de liens inter AC est elle-même générée et peut être considérée comme étant une décision de l'AC. Elle sera insérée de manière dynamique dans le modèle comportemental global de l'AC suite à sa validation par l'utilisateur du système si l'AC est *adolescent*[44]. En synthèse, l'AC voit son comportement même devenir dynamique et évolutif par l'ajout de ces règles générées.

Exemple de règle générée dans un *ruleGroup* global d'identification des liens :

```
<ruleGroup description="findAgentLinks" weight="1">
   <do effector="AgentLinkCreator"
      request="SELECT agent.* FROM agent, agt_objective, objective,
      obj_domain WHERE
      text(obj_domain.code)=text(objective.domain_objective_id) AND
      text(agt_objective.agent_id)=text(agent.code) AND
      text(objective.code)=text(agt_objective.objective_id) AND
      text(obj_domain.descriptiondomaineobj)=
      text(&amp;apos;Générique&amp;apos;) AND
      agent.code&amp;lt;&amp;gt;1508"
   />
</ruleGroup>
```

L'effecteur *AgentLinkCreator* prend en paramètre une requête SQL générée suite à l'analyse de l'environnement de l'AC comme cela est décrit dans le chapitre de modélisation. Suite à la création de la règle, l'AC établira des liens, à valider ou non en fonction de son âge, avec d'autres ACs qui seraient identifiés suite à exécution de cette requête dans l'effecteur.

Etape 4. Si la validation humaine est prononcée, l'AC met à finalement à jour le souvenir correspondant à la prise de décision courante. Dans la table mémoire *memory_Ci*, le champ *human_validated* passe alors de la valeur *faux* à la valeur *vrai*.

[44] Rappel : la validation est considérée comme acquise par défaut si l'AC est mature

3.12 Comportement N°4 : participer à un nouveau projet

3.12.1 Introduction

Ce comportement traduit la proactivité des ACs lors de la création de nouveaux projets par l'utilisateur dans notre système. On rappelle, selon notre modèle théorique, qu'un projet est vu en tant qu'instance d'un objectif plus général. Par exemple, considérons que l'utilisateur du système a pour projet de construire le collège de X, instance de l'objectif global « construire un collège ». Ici, on attend du système l'identification des compétences requises (maître d'ouvrage, maître d'œuvre, architecte, etc.) pour la création du nouveau projet. Pour réaliser informatiquement ce comportement, un nouveau projet sera instancié que :

1. Un projet similaire existe dans le système et reflète finalement les attentes. Il devient possible de construire le nouveau projet en utilisant tout ou partie des compétences mises en œuvre dans ce projet.
2. Il n'y a pas de projet existant dans le système qui reflète les attentes. Compte tenu d'une certaine cohérence fonctionnelle, nous proposons de construire une liste globale de compétences à partir de toutes celles qui sont réparties dans tous les projets du même domaine d'objectif.

Nous nous situons dans un environnement multi-agents où les agents eux-mêmes sont des compétences. Dans ce contexte, notre implémentation ne se contente pas de rechercher simplement en base de données la liste des compétences pour le nouveau projet. Elle place bien les ACs au cœur du système dans le but de les rendre « proactifs » quant à leur décision de proposer leur participation au nouveau projet.

3.12.2 Cas N°1 : utiliser un objectif existant

3.12.2.1 Ecran de paramétrage de la demande utilisateur

La demande s'effectue de manière explicite via l'écran présenté figure 23. Les paramètres de filtrage des projets existants que sont *type d'environnement*, *environnements*, *domaines d'objectif* et *objectifs* sont sélectionnables dans des listes déroulantes. Par défaut, deux choix sont offerts à l'utilisateur selon le type de demande qu'il formule :

- Sa demande concerne effectivement la création d'un nouvel objectif. Dans ce cas, la case à cocher « *Ma demande concerne un nouvel objectif avec les domaines, environnement et type d'environnement sélectionnés* » est positionnée à *oui* et une zone de texte lui permet de saisir un libellé pour le nouveau projet.
- Sa demande ne concerne pas la création d'un nouvel objectif mais plutôt uniquement une demande d'information sur les compétences ayant été utilisées pour tels ou tels projets antérieurs. ici, la case à cocher « *Ma demande concerne un*

nouvel objectif avec les domaines, environnement et type d'environnement sélectionnés » est positionnée à *non* et aucune zone de texte complémentaire n'est affichée.

L'utilisateur peut également, en fonction de son besoin, rendre les paramètres de filtrage *stricts* en positionnant la valeur de la case à cocher « *Ma recherche est exacte, les valeurs de chaque paramètre sont à prendre en compte* » à *oui*.

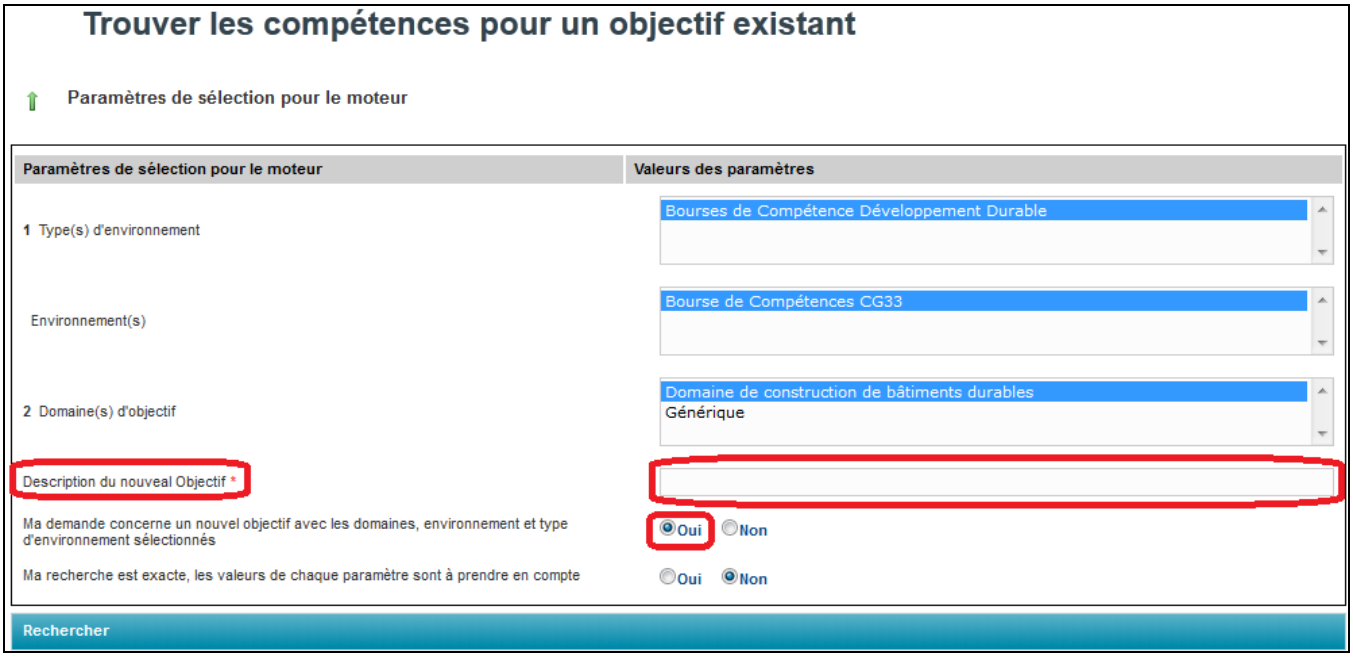


Figure 23 Créer un nouveau projet : cas 1 : paramétrage de la demande utilisateur

3.12.2.2 Echanges de flux

La figure 24 propose une vue schématique des échanges de flux avec ***ObjectiveAgent***.

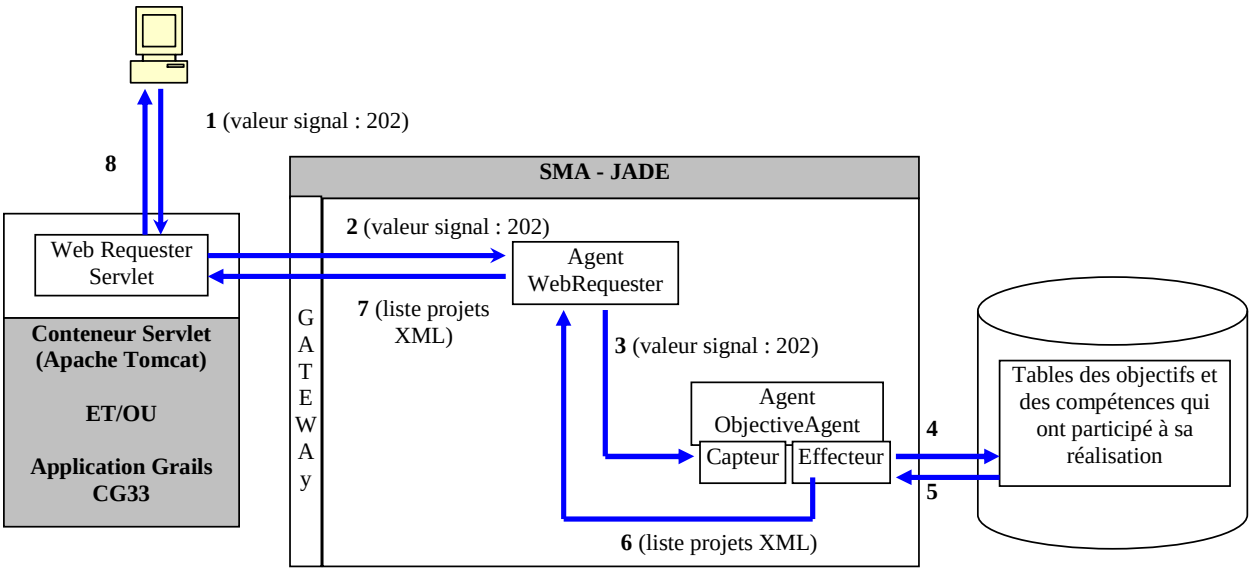


Figure 24 Créer un nouveau projet : cas 1 : échanges de flux

Pour établir un parallèle avec les fichiers de configuration déjà détaillés plus haut, on notera que la valeur **202** du signal correspond au déclenchement de l'effecteur *FindObjectives* qui est déclaré dans le tag *objectiveEffectorsList* du fichier global de configuration ***${SMA_ROOT}/conf/agents/objectiveAgent.properties***. De manière plus détaillée, selon l'étape d'échange, les flux identifiés lors de la mise en œuvre de ce comportement sont présentés tableau 18.

N°	Etape 1 : exécution de la demande de liste
1	**Exemple de paramètres vers moteur de R&D** signalCode=202 stepNumber=1 agentName=ObjectiveAgent exactSearch=false agentCodes= newObjective=true envTypeId=312 environementId=64 objectiveDomainId=1908 objectiveId=1909 newObjectiveDescription=Nouveau_Collège newObjectiveCheckBox=true

	exactSearchCheckBox=false
2	Transmission (standard JADE) de la demande entre *WebRequester Servlet* et *WebRequester Agent*
3	Transmission de la demande à l'agent objectif
4	Recherche en base de la liste des projets, avec les compétences associées, correspondant aux critères définis par l'utilisateur.
5	Extraction de la liste des projets et formatage global sous la forme d'un flux XML
6	Renvoi du flux XML des liens vers *WebRequester Agent*
7	Transmission (standard JADE) de la liste dans le flux XML entre *WebRequester Agent* et *WebRequester Servlet*
8	Affichage de la liste des projets avec les compétences associées au format HTML

Tableau 18 : Créer un nouveau projet : cas 1 : échanges de flux

3.12.2.3 Apprentissage

Dans ce contexte, il n'y a pas d'apprentissage et il ne se justifie pas. On ne cherche qu'à retrouver la liste des anciens projets, avec leurs compétences associées. Il n'y a donc pas d'intelligence particulière derrière ce traitement. Le nouveau projet est simplement instancié à partir de tout ou partie de la liste des compétences utilisées dans le cadre d'un projet antérieur.

3.12.2.4 Cinématique de création du nouveau projet

La figure 25 montre une illustration affichant les résultats (ici un seul projet) de la recherche des projets existants. Pour créer son nouveau projet, l'utilisateur sélectionne tout ou partie des compétences réparties sur un ou plusieurs projets existants. Il peut également décider d'utiliser simplement un projet existant, c'est-à-dire que le nouveau projet sera constitué par clonage du projet désigné.

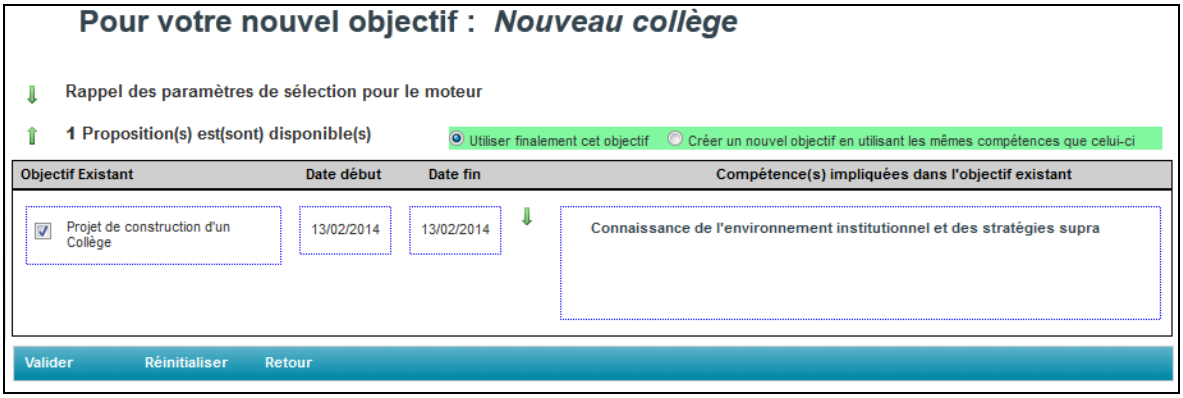


Figure 25 Créer un nouveau projet : cas 1 : constitution interactive du nouveau projet

En cliquant sur le bouton « ***Valider*** », le système génère automatiquement le nouveau projet ainsi que toutes les demandes de compétence qui lui sont associées.

3.12.3 Cas N°2 : proactivité des Agents Compétence

3.12.3.1 Ecran de paramétrage de la demande utilisateur

La demande s'effectue de manière explicite via l'écran présenté figure 26. Les paramètres de filtrage des projets existants que sont *type d'environnement*, *environnements*, et *domaines d'objectif* sont sélectionnables dans des listes déroulantes. Deux paramètres sont positionnés par défaut :

1. Sa demande est *stricte* sur les critères de filtrage : la case à cocher « *Ma recherche est exacte, les valeurs de chaque paramètre sont à prendre en compte* » a pour valeur *oui*.
2. Le coefficient de pertinence a pour valeur 50 par défaut.

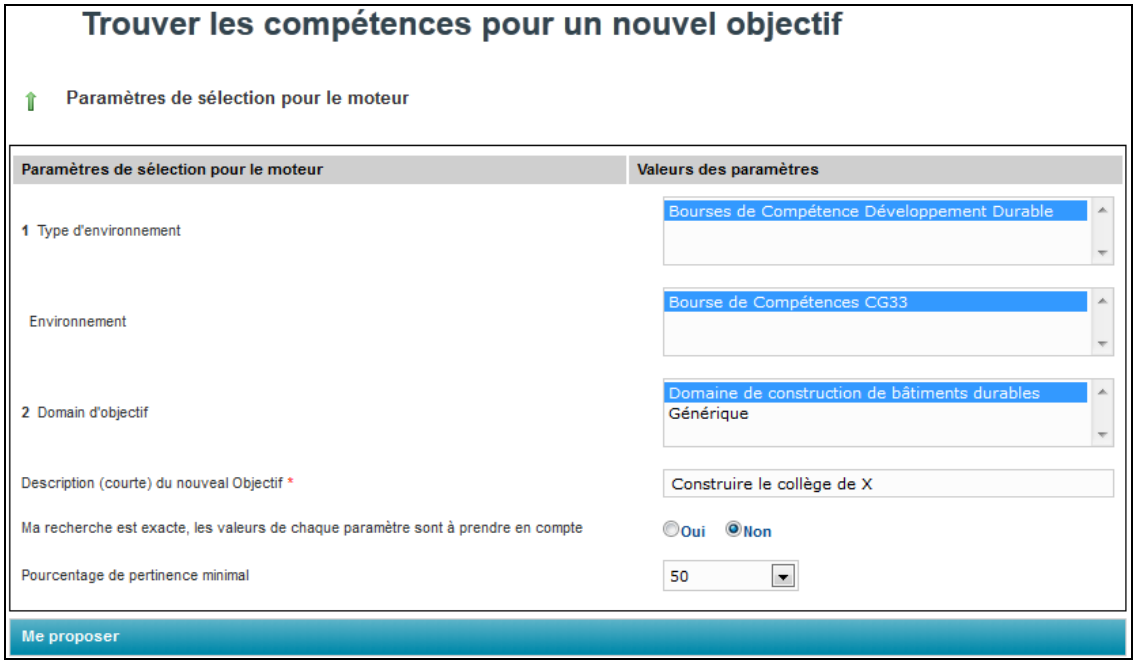


Figure 26 Créer un nouveau projet : cas 2 : paramétrage de la demande utilisateur

3.12.3.2 Echanges de flux

La figure 27 propose une vue schématique des échanges de flux avec ***ObjectiveAgent***.

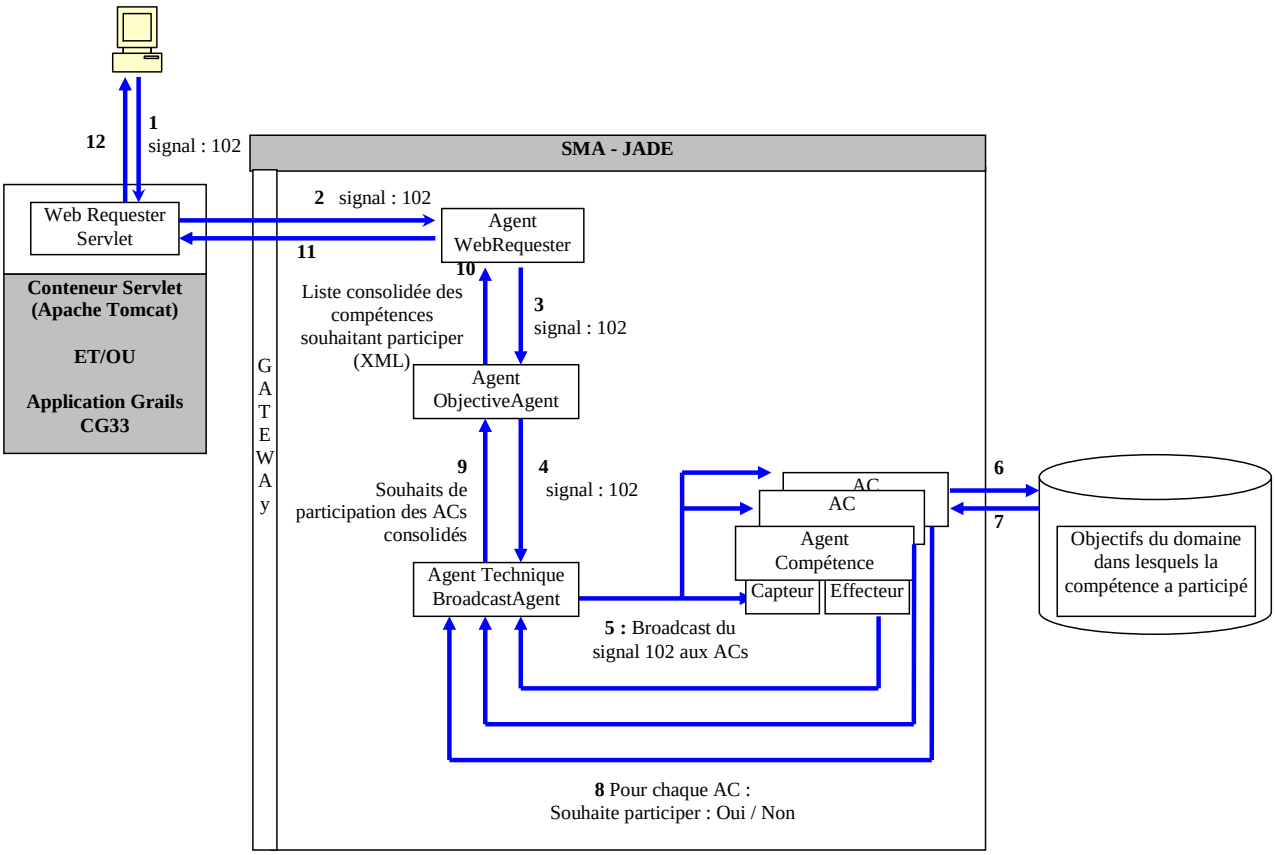


Figure 27 Créer un nouveau projet : cas 2 : échanges de flux

Pour établir un parallèle avec les fichiers de configuration déjà détaillés plus haut, on notera que la valeur **102** du signal correspond au déclenchement de l'effecteur FindObjectiveSkills qui est déclaré dans le tag *objectiveEffectorsList* du fichier global de configuration ***${SMA_ROOT}/conf/agents/objectiveAgent.properties***. De manière plus détaillée, selon l'étape d'échange, les flux identifiés lors de la mise en œuvre de ce comportement sont présentés tableau 19.

N°	Etape 1 : exécution de la demande de liste
1	**Exemple de paramètres vers moteur de R&D** signalCode=102 stepNumber=1 agentName=ObjectiveAgent exactSearch=false doBroadcast=true newObjective=true

	newObjectiveDescription=Nouveau_Collège humanDecision=false envTypeId=312 environementId=64 objectiveDomainId=1908 exactSearchCheckBox=false minPercentValue=50
2	Transmission (standard JADE) de la demande entre *WebRequester Servlet* et *WebRequester Agent*
3	Transmission de la demande à l'agent objectif
4	Transmission de la demande à l'agent technique *BroadcastAgent* chargé de la diffuser à tous les ACs
5	Diffusion de la demande à tous les ACs présents dans le SMA, sans aucune distinction
6	Recherche de la liste des projets du domaine auxquels l'AC courant a participé
7	Extraction de la liste des projets du domaine auxquels l'AC courant a participé et calcul du coefficient de pertinence
8	Renvoi, dans un flux XML, de la décision de participation au nouvel objectif, en fonction de la valeur du coefficient de pertinence calculé et du seuil demandé pour ce dernier (*minPercentValue*). La décision revêt la forme d'une valeur booléenne
9	Renvoi vers *ObjectiveAgent*, dans un flux XML, des réponses de chaque AC
10	Renvoi, dans un flux XML, de la liste des compétences ayant répondu oui à une participation au nouveau projet
11	Transmission (standard JADE) de la liste dans le flux XML entre *WebRequester Agent* et *WebRequester Servlet*
12	Affichage HTML de la liste des compétences se portant candidates à une participation au nouvel objectif.

Tableau 19 : Créer un nouveau projet : cas 2 : échanges de flux

3.12.3.3 Cinématique de création du nouveau projet

La figure 28 (basée sur le jeu de rugby par exemples), montre une illustration affichant les résultats. Il existe une colonne « *En se basant sur le(s) élément(s) historique(s)* ». Celle-ci est destinée à fournir des explications à l'utilisateur quand aux éléments d'historique qui ont permis à l'AC de se porter candidat. Pour créer son nouveau projet, l'utilisateur peut sélectionner tout ou partie des compétences proposées.

Suggestion de compétence(s) pour construire un nouvel objectif

Rappel des paramètres de sélection pour le moteur

Liste des compétences candidates à la réalisation de votre objectif, avec un niveau minimal de pertinence de 50 %

Compétence	Veut participer	Pertinence (%)	En se basant sur le(s) élément(s) historique(s)	Choisir
Troisième Ligne	Oui	100 %	A déjà été impliquée dans 1 objectif(s) sur 1 du domaine, avec l'historique : • 26/04/2012 : 100% des critères sélectionnés, objectif : Match Fictif (initialisation du système)	
Centre	Oui	100 %	A déjà été impliquée dans 1 objectif(s) sur 1 du domaine, avec l'historique : • 26/04/2012 : 100% des critères sélectionnés, objectif : Match Fictif (initialisation du système)	
Arrière	Oui	100 %	A déjà été impliquée dans 1 objectif(s) sur 1 du domaine, avec l'historique : • 26/04/2012 : 100% des critères sélectionnés, objectif : Match Fictif (initialisation du système)	
Ouverture	Oui	100 %	A déjà été impliquée dans 1 objectif(s) sur 1 du domaine, avec l'historique : • 26/04/2012 : 100% des critères sélectionnés, objectif : Match Fictif (initialisation du système)	
Pilier	Oui	100 %	A déjà été impliquée dans 1 objectif(s) sur 1 du domaine, avec l'historique : • 26/04/2012 : 100% des critères sélectionnés, objectif : Match Fictif (initialisation du système)	
Talonneur	Oui	100 %	A déjà été impliquée dans 1 objectif(s) sur 1 du domaine, avec l'historique : • 26/04/2012 : 100% des critères sélectionnés, objectif : Match Fictif (initialisation du système)	
Deuxième Ligne	Oui	100 %	A déjà été impliquée dans 1 objectif(s) sur 1 du domaine, avec l'historique : • 26/04/2012 : 100% des critères sélectionnés, objectif : Match Fictif (initialisation du système)	
Demi de mêlée	Oui	100 %	A déjà été impliquée dans 1 objectif(s) sur 1 du domaine, avec l'historique : • 26/04/2012 : 100% des critères sélectionnés, objectif : Match Fictif (initialisation du système)	
Ailier	Oui	100 %	A déjà été impliquée dans 1 objectif(s) sur 1 du domaine, avec l'historique : • 26/04/2012 : 100% des critères sélectionnés, objectif : Match Fictif (initialisation du système)	

Valider Réinitialiser Retour

Figure 28 Créer un nouveau projet : cas 2 : constitution interactive du nouveau projet

En cliquant sur le bouton « ***Valider*** », le système génère automatiquement le nouveau projet ainsi que toutes les demandes de compétence qui lui sont associées.

3.13 Simulateur automatisé de sélection des acteurs

3.13.1 Introduction

Une expérimentation grandeur nature de la Bourse aux Compétences du Développement Durable est bien en cours au Conseil Général de la Gironde au moment de la rédaction de ce mémoire. Cependant, nous ne disposons pas encore de données suffisamment fournies pour nous permettre de tirer des conclusions sur la pertinence du modèle proposé dans le réel. Pour vérifier nos hypothèses et pouvoir nous projeter, nous avons alors créé un « automate simulateur » pour le comportement d'identification des acteurs. Son écran de paramétrage est proposé figure 29.

Figure 29 Simulateur de sélection d'acteurs : paramétrage de la demande utilisateur

L'objectif de chaque simulation est de reproduire le comportement et les actions des utilisateurs humains de la Bourse. Chacun d'eux effectuerait des demandes de compétence au système en affectant des poids aux compétences élémentaires puis procèderait à une (in)validation des résultats retournés par le moteur de R&D.

3.13.2 Scénarios de simulation

Pour que la simulation soit pertinente, nous avons choisi de faire en sorte que nos utilisateurs automatisés :

1. Sélectionnent toujours la même compétence (« *connaissance de l'environnement institutionnel et des stratégies supra* » dans notre cas). Ceci n'est pas rédhibitoire puisque :
   - Le comportement du modèle est identique et applicable quelle que soit la compétence.
   - La notion de liens entre nos compétences n'est pas pertinente pour la sélection des acteurs.

- Nous focaliser sur une seule compétence nous permet de lui adresser un maximum de requêtes (demandes) afin de pouvoir observer les tendances des variations des poids des critères dans le temps.

2. Affectent des poids de manière aléatoire à chacun des 4 critères d'efficacité que sont « *Satisfaction du besoin* », « *Valeur ajoutée pour le réseau* », « *Opérationnalité de l'échange* » et « *Qualité de l'échange* ». Le choix d'affecter des valeurs aléatoirement (tout de même bornées entre 1 et 10) n'est pas non plus dénué de sens. Il nous permet de refléter une variabilité forte des demandes et donc des contextes utilisateur très variés. Nous simulons finalement un grand nombre de contextes, représentés par des poids de critères très variables.

3. Valident ou invalident les résultats fournis par le moteur (i.e. l'agent compétence choisi) de manière aléatoire. Bien entendu, nous sommes conscients que cela n'a pas de sens si l'on raisonne de manière rationnelle. La(l') (in)validation est prononcée, dans la vie réelle, uniquement après analyse humaine de pertinence du résultat présenté à l'utilisateur. Ici encore, le choix de valider aléatoirement les résultats fournis, nous offre une grande variabilité du comportement utilisateur en face des résultats présentés. Sans aller plus loin, notons toutefois toute la difficulté qui consisterait à réaliser un système de validation automatique pertinent. Comment le réaliser ? Quelles en sont les bases concrètes, quel raisonnement appliquer, quelle proportion de subjectivité, etc. ? Ceci ouvre certes une perspective forte intéressante sur d'autres champs d'étude scientifiques mais nous ne les prendrons pas en compte dans notre simulation. De plus, nous ne prenons pas dans notre simulation les évaluations post-réalisation concrète de la compétence qui participent au choix effectué par l'humain. En mode simulé, ce n'est toutefois pas impactant puisqu'on rappelle que nous cherchons à valider le modèle de mémorisation et de calcul des poids que nous proposons. Sachant qu'une demande utilisateur est égale à une simulation, nous avons réalisé 3 scénarios de simulation, contenant chacun 5 tirs (= occurrences des scénarios). Le tableau 20 nous fournit des informations sur ces simulations pour chacun des 3 scénarios.

Scénario	Nombres de simulations	Nombre de tirs (occurrences de scénario)
1	100	100
2	200	200
3	300	300

Tableau 20 : Simulateur de sélection d'acteurs : détail des scénarios

Bien que nous l'ayons expliqué plus haut, on identifie les limites suivantes par rapport à des cas d'utilisation réelles pour notre simulateur :

1. Attribution de poids aléatoires.
2. (In)Validations aléatoires des résultats retournés.
3. Absence de spécifications d'évaluation.

3.13.3 Eléments clés de simulation

Afin de constater d'éventuels problèmes (ou non homogénéité des résultats entre chaque scénario), la mémoire de l'agent compétence est purgée avant tout lancement d'un tir. Les poids des critères d'efficacité, quant à eux, sont valorisés de manière aléatoire entre 1 et 10 tout au long de la simulation de demande utilisateur. Par exemple, pour le **3**^{ème} tir du scénario **3** à **300** simulations, nous avons observé, pour « *Qualité de l'échange* » et « *Opérationnalité de l'échange* », les variations aléatoires présentées à la figure 30.

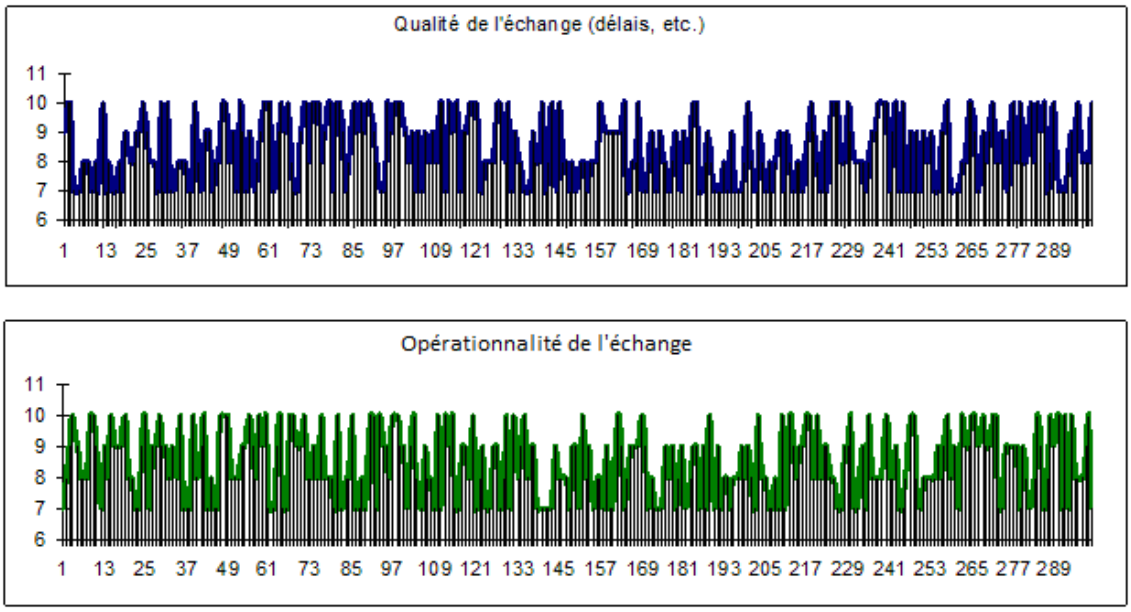


Figure 30 Simulateur de sélection d'acteurs : variation des poids des critères d'efficacité

De manière synthétique (moyennée), sur les 5 tirs des 3 scénarios, on notera également que le pourcentage de validation humaine aléatoire prononcé est globalement de 48% par scénario, avec une valeur minimale de 46% et maximale de 51%.

3.13.4 Résultats obtenus

L'utilisation du simulateur nous a conduit à vérifier les hypothèses formulées pour la méthode de calcul des pondérations des C_{et} et observer leur convergence. Par exemple, pour les critères « *Qualité de l'échange* » et « *Satisfaction du besoin* », nous obtenons le graphique présenté figure 31 qui montre cette convergence.

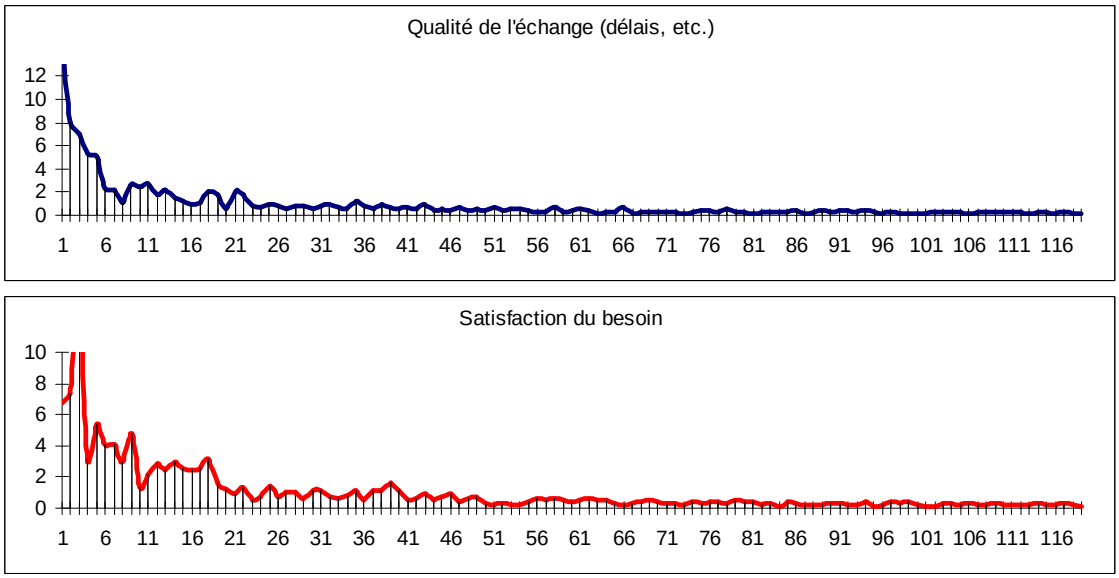


Figure 31 Simulateur de sélection d'acteurs : convergence des valeurs de poids

Nous pouvons également vérifier l'efficacité du mécanisme d'apprentissage pour les calculs de pondération des C_{et} via l'atteinte d'une certaine stabilité, observable autour des 120 requêtes et validations prononcées par les utilisateurs simulés. Ce résultat est bien entendu à considérer dans le contexte de nos travaux de simulation. La figure 32 nous montre ce phénomène de stabilisation.

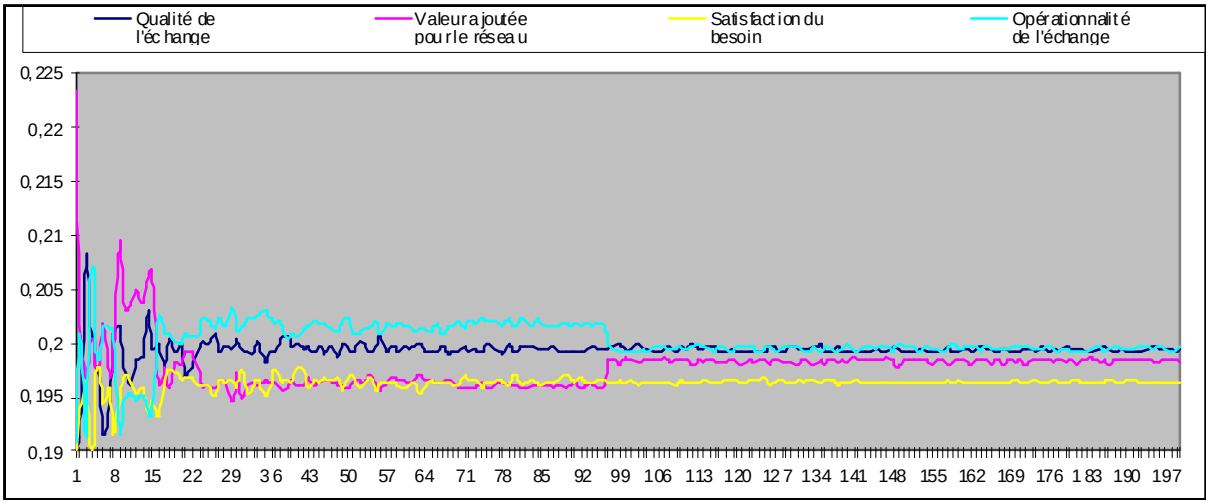


Figure 32 Simulateur de sélection d'acteurs : stabilité des pondérations calculées

3.13.5 Proposition d'interprétation

Il n'est pas dénué de sens de considérer d'un point de vue théorique que plus un critère (une compétence élémentaire) est « importante » pour la réalisation d'une compétence

demandée, plus son poids calculé sera élevé. La courbe ci-dessus tendrait à montrer « visuellement » ce phénomène. Par exemple, la compétence élémentaire « *Opérationnalité de l'échange* » pourrait être considérée comme prévalant sur « *Valeur ajoutée pour le réseau* ». Même si nous sommes bien conscients ici d'être uniquement dans le domaine de la simulation, nous pouvons cependant penser avec une certaine pertinence que dans le cadre d'une mise en œuvre dans la réalité, nous serons capables d'identifier avec une certaine exactitude quelles sont les compétences élémentaires les plus « importantes » pour permettre la réalisation d'une compétence de niveau plus global. Il suffirait en effet pour cela de construire de manière automatique un graphique à partir des poids des critères figurant dans les souvenirs validés de chaque AC. En allant plus loin, supposons que nous puissions constater que le niveau d'importance d'une compétence élémentaire décroisse de manière significative par rapport aux autres. Des actions pourraient alors être engagées de manière automatisée. Par exemple, doit-on conserver ce critère de jugement d'efficacité dans le système ? Est-il toujours d'actualité ? Est-il englobé par un autre critère ? Que doit alors exécuter comme actions l'agent compétence ?

3.14 Exercice d'application

3.14.1 Contexte

Comme nous l'avons déjà vu plus haut[45], une expérimentation grandeur nature de la Bourse aux Compétences du Développement Durable (BCDD) est bien en cours au Conseil Général de la Gironde au moment de la rédaction de ce mémoire. Cependant, nous ne disposons pas encore de données suffisamment fournies pour nous permettre de tirer des conclusions sur la pertinence du modèle proposé dans le réel. Dans ce cas, comment asseoir nos assertions ?

Pour vérifier nos travaux[46], et en même temps la généricité de notre proposition, formulons l'hypothèse que nous mettons notre outil à disposition de l'entraineur de rugby de l'équipe de l'ASM Clermont Auvergne, M. Vernon COTTER (néo-zélandais). Ce dernier initialise les informations de la base de données[47] de notre outil dans le contexte du TOP 14, championnat Français de rugby, pour la saison 2013-214. Il apparait évident que ce secteur d'activité professionnel n'a réellement rien à voir avec celui des projets de Développement Durable traités dans BCDD du CG33.

[45] Cf. paragraphe « *3.13 Simulateur automatisé de sélection des acteurs* » plus haut

[46] Toutes les illustrations fournies (bandeau, etc.) proviennent du site officiel : http://www.asm-rugby.com/

[47] Le SMA lui-même n'est pas lancé car on part d'un système vide, i.e. qu'aucun AC n'est encore créé

L'interface graphique que manipule M. COTTER pourrait être celle proposée à la figure 33.

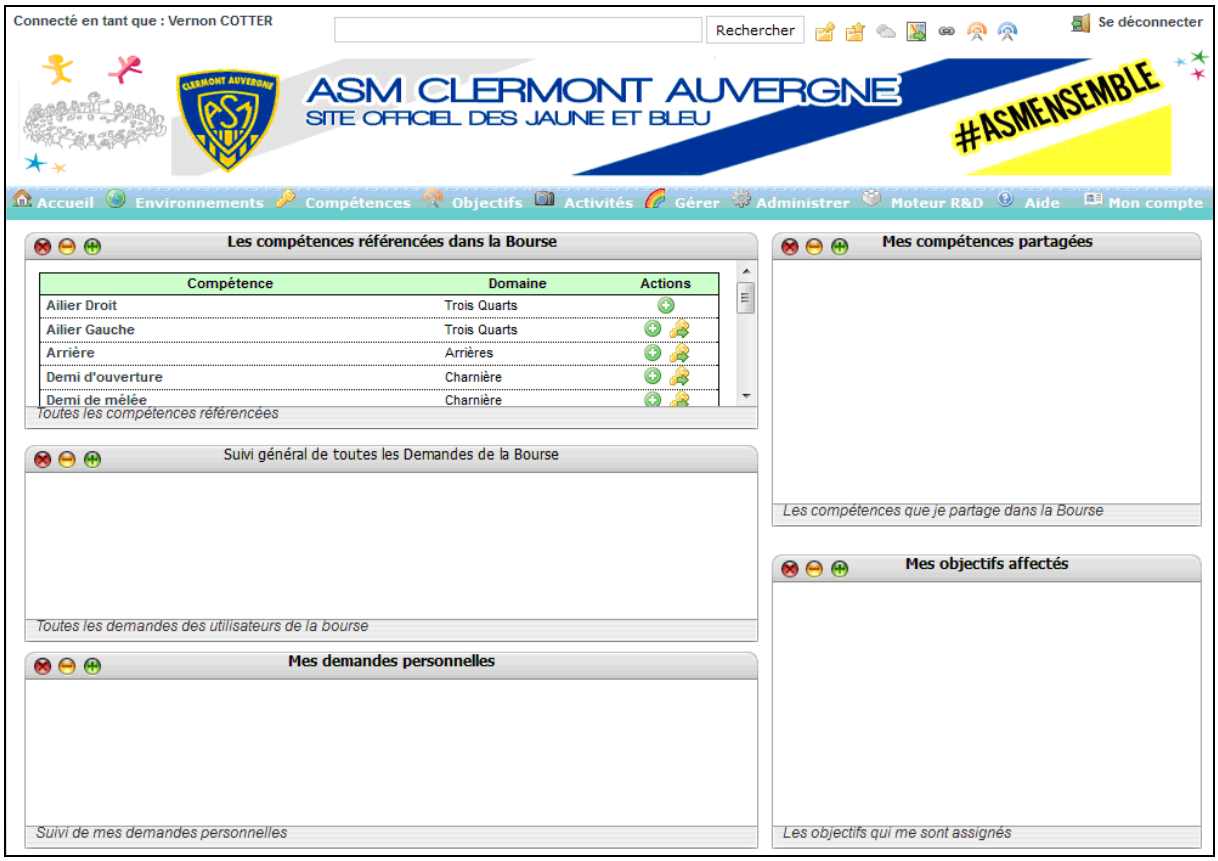
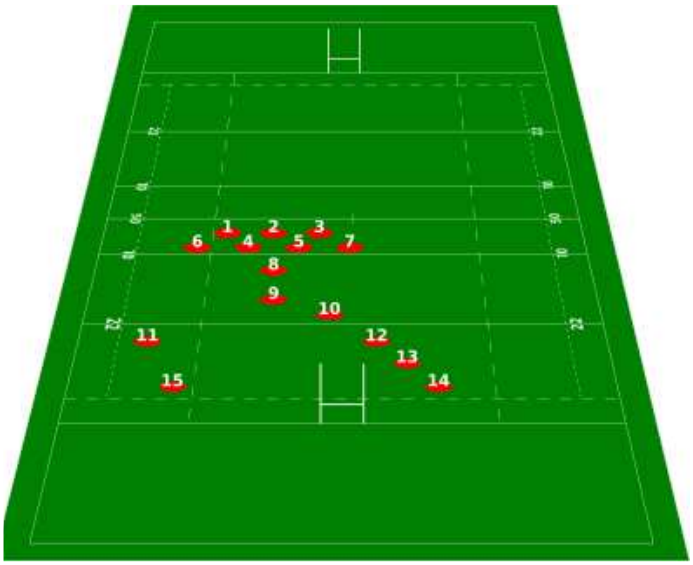


Figure 33 Interface de l'outil pour Vernon COTTER, entraineur de l'ASM

Pour initialiser le système (à partir d'une base de données vide), M. COTTER commence par créer un type d'environnement ***TOP 14*** puis un environnement ***TOP 14 – Saison 2013/2014*** qui lui est rattaché. Il aurait pu également créer un type d'environnement ***H-Cup*** pour représenter le championnat Européen de rugby auquel participe son équipe depuis de nombreuses années. Une équipe de rugby est composée de joueurs (i.e. acteurs humains) évoluant à 15 postes différents, nécessitant l'exercice de compétences spécifiques (voir figure 34).



1 → Pilier Gauche
2 → Talonneur
3 → Pilier droit
4 → Seconde ligne gauche
5 → Second ligne droit
6 → Troisième ligne aile gauche
7 → Troisième ligne aile droit
8 → Troisième ligne centre
9 → Demi de mêlée
10 → Demi d'ouverture
11 → Ailier gauche
12 → Premier centre
13 → Deuxième centre
14 → Ailier droit
15 → Arrière

Figure 34 Les 15 postes composant une équipe de rugby[48]

M. COTTER va créer 15 compétences à l'intérieur du système, une compétence correspondant à un poste sur le terrain. Il spécifie également, pour chacune d'elles, les compétences élémentaires[49] qui la caractérisent et permettront d'évaluer son exercice, par un joueur, et pour un match donné. Par exemple (à adapter bien entendu), on peut avoir la décomposition présentée au tableau 21.

Compétences élémentaires	Postes occupés														
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Pourcentage de passes réussies	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Pourcentage de plaquages réussis	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Nombre de ballons pris à l'adversaire	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Nombre de ballons touchés	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Capacité de pénétration (nombre de mètres avancés avec le ballon en mains)	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Vitesse pure (pour les trois-quarts et arrières)									✓	✓	✓	✓	✓	✓	✓
Détente pure (pour les sauteurs)				✓	✓	✓	✓	✓							
Poussée maximale (pour les mêlées)	✓	✓	✓	✓	✓	✓	✓	✓							
Nombre de kilomètres parcourus rapporté au temps de jeu	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Pourcentage de réussite de pénalités et drops									✓	✓	✓	✓	✓	✓	✓

Tableau 21 : Compétences élémentaires pour chaque poste de rugby

[48] Ref. image : http://fr.wikipedia.org/wiki/Composition_d%27une_%C3%A9quipe_de_rugby_%C3%A0_XV
[49] Fixées arbitrairement pour cette simulation

M. COTTER doit ensuite préciser qui (joueur ou acteur humain) occupe quel poste dans son équipe. Une fois ceci fait, à titre d'illustration, voici le détail de la compétence « *Ailier droit* » dans notre système. Voir tout particulièrement la partie « *Acteurs partageant cette compétence* » dans la figure 35.

Date de création	12 / 06 / 2014 - 11:40:11
Compétence	Ailier Droit
Mot(s) clé(s)	attaquer, rapidité, agilité, marquer, contre-attaquer
Domaines de rattachement	• **Trois Quarts**
Critères / compétences élémentaires	• **Nombre de kilomètres parcourus rapporté au temps de jeu** • **Pourcentage de passes réussies** • **Capacité de pénétration (nombre de mètres avancés avec le ballon en mains)** • **Nombre de ballons touchés** • **Pourcentage de plaquages réussis** • **Vitesse pure (pour les trois-quarts et arrières)** • **Nombre de ballons pris à l'adversaire** • **Pourcentage de réussite de pénalités et drops**
Infos	• Les ailiers gauche (11) et droit (14) sont en bout de ligne d'attaque. Ils sont souvent d'un gabarit plus petit, leur vitesse doit leur permettre de contourner la défense adverse. Ce sont aussi des finisseurs, généralement les meilleurs marqueurs d'essais. En défense, ils constituent le troisième rideau et aident l'"arrière" à contre-attaquer. Depuis le phénomène Jonah Lomu, il arrive qu'un joueur de grand gabarit (de plus d'1m90 et d'environ 100kg)mais ayant tout de même une bonne vitesse, soit positionné au poste d'ailier.
Mode apprentissage ?	Oui
Age courant	**Enfance**
Nombre d'utilisations	0
Acteurs partageant cette compétence	• **Aurelien ROUGERIE** • **Naipolioni VONOWALE NALAGA** • **Julien MALZIEU** • **Sitiveni waica SIVIVATU** • **Noa seru NAKAITACI** • **Jean marcellin BUTTIN** • **Admin Administrateur**

Figure 35 Détails d'un poste de joueur

La saison 2013/2014 du TOP 14 de rugby[50] possède un calendrier des matchs défini à l'avance par la Ligue Nationale de Rugby (LNR), avant le démarrage de la saison. Ce calendrier est divisé en 5 parties, appelées phases du championnat, et au sein desquelles les matchs sont répartis. Dans le cadre de notre système, ces phases constituent les « *Domaines d'objectifs* » que M. COTTER va créer :

1. Matchs Aller.
2. Matchs Retour.
3. Matchs de barrage.
4. Demi-finales.
5. Finale.

[50] Données prises sur le site web de la LNR (http://www.lnr.fr/calendrier-resultats-top-14.html)

Comme toutes les équipes impliquées, tout au long de la saison 2013/2014 du TOP 14, l'ASM a cherché à répondre à plusieurs reprises au même objectif : gagner match après match. Considérons que « *gagner un match* » est un objectif global à réaliser plusieurs fois. Dans ce cas, chacun des matchs joués contre une équipe adverse peut être vu comme une instance (une nouvelle occurrence) de cet objectif. Finalement, un match peut être vu comme un projet à concrétiser à une date définie par le calendrier officiel des matchs. Pour la saison 2013/2014 du top 14, M. COTTER a créé chacun des matchs (projets) dans le système en le reliant à la phase du championnat à laquelle il est rattaché. En fin de saison, cela permet d'obtenir la copie d'écran de la figure 36.

Environnement	Top 14 - Saison 2013 / 2014
Date Début	17/08/2013
Date Fin	10/05/2014
Type d'environnement	**Top 14**
Objectifs dans le domaine	• Barrage - Clermont / Castres (16-22) • Clermont / Perpignan (25-22) • Racing Metro / Clermont (22-6) • Clermont / Castres (23-11) • Brive / Clermont (26-24) • Clermont / Toulon (22-16) • Bordeaux / Clermont (26-16) • Clermont / Montpellier (42-16) • Grenoble / Clermont (16-13) • Clermont / Paris (25-13) • Bayonne / Clermont (18-9) • Toulouse / Clermont (19-12) • Clermont / Oyonnax (33-19) • Clermont / Biarritz (35-6) • Perpignan / Clermont (23-30) • Clermont / Racing Metro (47-14) • Castres / Clermont (22-22) • Clermont / Brive (36-29) • Toulon / Clermont (25-19) • Clermont / Bordeaux (40-11) • Montpellier / Clermont (43-3) • Clermont / Grenoble (27-13) • Paris / Clermont (23-16) • Clermont / Bayonne (55-0) • Clermont / Toulouse (38-19) • Oyonnax / Clermont (30-19) • Biarritz / Clermont (18-22) • Sans objectif Particulier

Figure 36 Liste des matchs de l'ASM[51] pour la saison 2013/2014

Pour chacun des matchs (projet), M. COTTER a bien entendu attribué l'exercice de chaque compétence (poste sur le terrain) à un joueur. Ceci est bien illustré par la copie d'écran présentée dans la figure 37. On y observe que le match Clermont / Toulouse, de la phase « *Matchs Aller* » du championnat, a utilisé les compétences recensées dans la zone « *Compétences utilisées* » de la copie d'écran, avec les acteurs ayant débuté le

[51] Format : *Equipe qui reçoit / Equipe qui se déplace (score équipe qui reçoit – score équipe qui se déplace)*

match[52] mentionnés en face de chacune d'elles. Ce match c'est déroulé le 31 Août 2013. Pour information, c'est l'ASM qui recevait à son stade Marcel Michelin de Clermont-Ferrand.

Objectif	Clermont / Toulouse (38-19)
Date Début	31/08/2013
Date Fin	31/08/2013
Environnement	Top 14 - Saison 2013 / 2014
Domaine	Matchs Aller
Compétences utilisées	• Pilier Gauche : Thomas DOMINGO • Talonneur : Benjamin KAYSER • Troisième ligne aile Gauche : Julien BONNAIRE • Troisième ligne Centre : Julien BARDY • Troisième ligne aile Droit : Damien CHOULY • Premier Centre : Wesley FOFANA • Demi de mêlée : Ludovic RADOSAVLJEVIC • Deuxième Centre : Benson WILLIAMS STANLEY • Seconde ligne Gauche : Julien PIERRE • Seconde ligne Droit : Nathan HINES • Demi d'ouverture : Brock JAMES • Ailier Gauche : Naipolioni VONOWALE NALAGA • Arrière : Lee BYRNE • Pilier Droit : Daniel KOTZE • Ailier Droit : Sitiveni waica SIVIVATU

Figure 37 Détails d'un match (projet)

Chaque poste occupé pendant un match est caractérisée par une liste d'une ou plusieurs (ici prises parmi 10) compétences élémentaires. A l'issu de chaque match, M. COTTER évalue l'exercice du poste, par le joueur et pour le match, sur la base de ces compétences élémentaires. Par exemple, considérons le poste de « Pilier Gauche » occupé par « Thomas DOMINGO ». Pour la rencontre Clermont – Toulouse du 31 Août 2013, on peut obtenir les évaluations présentées dans la figure 38.

Objectif & Compétence	Clermont / Toulouse (38-19) Pilier Gauche Thomas DOMINGO
Evaluateur	Vernon COTTER
Etat du formulaire	Validé
Date Début évaluation	31/08/2013
Date Fin évaluation	31/08/2013
Critére(s) à évaluer	• Capacité de pénétration (nombre de mètres avancés avec le ballon en mains) : 511 • Nombre de ballons pris à l'adversaire : 2 • Nombre de ballons touchés : 12 • Nombre de kilomètres parcourus rapporté au temps de jeu : 13 • Pourcentage de passes réussies : 24 • Pourcentage de plaquages réussis : 75 • Poussée maximale (pour les mêlées) : 228
Importance (sur 20)	1

Figure 38 Evaluation d'un poste, exercé par un joueur et pour un match

Tout comme les compétences élémentaires ont été fixées arbitrairement, et dans le but de gagner du temps, nous avons généré ces valeurs d'évaluation de la même manière, et

[52] Le système ne gère pas les notions de remplaçants en cours de partie dans sa version actuelle.

automatiquement. Nous avons toutefois tenté de garder une certaine cohérence compte tenu des caractéristiques des différents postes occupés. Notre méthodologie de génération automatique des valeurs est décrite dans le tableau 22.

Compétences élémentaires	Méthodologie de génération de la valeur
Pourcentage de passes réussies	Aléatoire
Pourcentage de plaquages réussis	Aléatoire
Nombre de ballons pris à l'adversaire	Aléatoire, inférieur à 10
Nombre de ballons touchés	Aléatoire, inférieur à 15
Capacité de pénétration (nombre de mètres avancés avec le ballon en mains)	Aléatoire, inférieur à 100 pour les Avants Aléatoire, entre 500 et 2000 pour demi de mêlée, demi d'ouverture et lignes arrières
Vitesse pure (pour les trois-quarts et arrières)	Usain BOLT court à 44,72 km/h[53] en vitesse de pointe On considère que la vitesse décroit de l'arrière vers le paquet d'avants On suppose que notre arrière court à 35 Km/h
Détente pure (pour les sauteurs)	Aléatoire, entre 50 et 100 cm
Poussée maximale (pour les avants)	Selon la publication (Kh'lifa, 1995), cette force varie entre 217% de leur poids pour les avants (moyenne 100 Kg plus ou moins 7,6 Kgs) et 219% de leur poids (81,2 Kg plus ou moins 7,1Kgs) pour les arrières. Dans notre simulation, proposons que seuls les avants aient cette capacité et que leur poids aléatoirement entre 100 et 107, 6 Kgs. On prendra donc une valeur aléatoire entre 217 et 233 pour valoriser cette compétence élémentaire.
Nombre de kilomètres parcourus rapporté au temps de jeu	Aléatoire entre 12 et 14 Kms[54]
Pourcentage de réussite de pénalités et drops	Considérons que c'est toujours le 10 qui botte Aléatoire, entre 30 et 100 %

Tableau 22 : Compétences élémentaires pour chaque poste de rugby

A ce stade, toutes les actions d'initialisation ont été accomplies. M. COTTER a certes renseigné des données statiques, mais cependant tout à fait contextuelles, à partir d'éléments réellement observés[55]. En conclusion, le système paramétré :

1. Est structuré et adapté à un domaine d'activité professionnel complètement différent de celui de la BCDD du CG33. On tend ainsi à prouver son adaptabilité à tout domaine professionnel, au moins pour la manipulation des concepts clés (compétences, domaines, environnement, etc.).
2. Dispose d'une liste de compétences nécessaires à l'accomplissement des objectifs fixés par le domaine professionnel. Ces compétences seront traduites (automatiquement) sous la forme d'ACs au moment du démarrage du SMA.
3. N'a pas encore mis en œuvre de mécanismes d'apprentissage, le SMA n'ayant pas encore été démarré.

[53] Cf. http://www.lemonde.fr/sport/portfolio/2009/08/17/usain-bolt-l-homme-qui-court-a-44-72-km-h_1229382_3242.html
[54] Cf. http://sante-medecine.lefigaro.fr/distance-match.html
[55] Si l'on veut bien considérer les valeurs d'évaluation générées des **Ce** comme « réelles » dans cette simulation

3.14.2 Scénarios de simulation globale

3.14.2.1 Spécifications

Le scénario de simulation proposé considère que l'équipe de l'ASM Clermont Auvergne reste dans une « situation constante ». Dans les faits, à l'intersaison, des joueurs et/ou des entraineurs changent de club. C'est le cas par exemple de M. COTTER qui quitte l'ASM pour devenir entraineur principal de l'équipe nationale d'Ecosse au moment de la rédaction de ce mémoire. De même, certains clubs quittent l'élite du rugby français et sont rétrogradés en seconde division (pro-D2) alors que d'autres montent en TOP 14. La liste des matchs à jouer varie donc chaque année. Dans tous les cas, ce type de mouvement de personnes ou de clubs entre les divisions n'a aucun impact sur le fonctionnement interne de l'outil et ses mécanismes d'apprentissage. Il convient juste d'accepter que la situation de notre simulation puisse diverger quelque peu de la réalité des faits observés dans le championnat de France de rugby. Ceci étant posé, nous proposons de réaliser par génération automatique les actions que mèneraient M. COTTER dans le contexte d'une nouvelle saison (2014/2015) du TOP 14 de rugby.

Nous créerons alors automatiquement les matchs au fur et à mesure de l'avancée simulée de la saison. Cette création sera faite à partir de la liste de ceux qui ont été joué pendant la saison 2013/2014. Pour chaque match créé, nous génèrerons :

1. Les demandes relatives à toutes celles de la liste des compétences (postes) à embarquer.
2. Les candidatures des acteurs possédant cette compétence pour leur réalisation.
3. L'opération d'affectation[56] de la compétence à un acteur en utilisant les mécanismes d'apprentissage.
4. Les évaluations des **Ce** pour chaque compétence exercée pendant le match.

L'algorithme d'apprentissage de sélection des acteurs (point 3) bénéficiera des évaluations portées au point 4 sur les matchs précédents.

A l'issu de la simulation, nous vérifierons si les objectifs de nos travaux sont atteints. Par exemple, par comparaison avec une situation réelle prise sur la saison 2013/204, nous devrions pouvoir constater factuellement que la répartition des affectations des postes parmi les joueurs répond mieux aux attentes de partage, montée en compétence réciproque, etc. Nous devrions également avoir un aperçu (et en vérifier la pertinence) des différents niveaux d'importance pris par chaque **Ce** dans la définition d'un poste.

3.14.2.2 Limites connues et identifiées

Dans le domaine fonctionnel choisi (équipe du TOP 14 de rugby), tous les matchs (i.e. les projets) embarquent la même liste de compétences. Cet état de fait, bien que

[56] Cf. paragraphe « *3.13 Simulateur automatisé de sélection des acteurs* » plus haut

complètement cohérent avec la réalité de la situation, induit qu'il ne peut y avoir d'évolution observable de la liste des compétences dans les projets. Nous devons alors accepter 2 limites importantes pour notre simulation automatisée. Elle ne permettra pas d'observer d'évolution :

1. Du coefficient de pertinence[57] pour la partie candidature des ACs à participation à un projet puisqu'elles participent toutes à tous les projets.
2. Du coefficient de proximité[58].pour la partie établissement des liens inter-agents. Pour rappel, ce coefficient entre 2 ACs est calculé à partir de l'observation de similarités prises dans :
   a) Les attributs internes des ACs.
   b) Les domaines de compétence auxquels elles appartiennent.
   c) Les compétences élémentaires qui les caractérisent.
   d) Les projets passés dans lesquels elles ont été impliquées.
   e) Les évaluations de ces projets.
   f) Les domaines des projets (e.g. TOP 14).
   g) Les types de ces domaines de projet (vides ici).

Le fait est que si la liste de compétence est unique dans tous les projets, les points d, e, f et g deviennent inutiles. En conséquence, ce coefficient est calculé à partir de données contextuelles « constantes » et non variables, d'où une valeur qui n'évolue pas.

A l'opposé, il convient de reconnaître que si cette liste évoluait, nous ne pourrions procéder à une simulation automatique complète sans passer par des phases de validation manuelles et successives tout au long de la génération des projets. Nous devrions alors analyser les listes de compétences candidates, valider ou non celles à embarquer dans les projets, etc. Ceci prend du temps et nécessite une connaissance approfondie du domaine. C'est en tous cas l'objectif recherché dans l'expérimentation terrain BCDD, engagée côté CG33. Si cette expérimentation s'avère prépondérante pour observer la variation des établissements de liens inter-ACs et la pertinence des candidatures à participation à projet, notre simulation automatique reste pertinente, d'autant plus dans un domaine professionnel comme celui que nous avons choisi, bien ancré dans la réalité.

3.14.3 Résultats expérimentaux

3.14.3.1 Lancement du SMA

Au lancement du système multi agents, les agents compétence représentant chacun des postes sont instanciés. La figure 39 montre les agents du SMA créés après démarrage dans JADE.

[57] Cf. paragraphe « *2.5.7.5 Comportement N°2 : sélection d'acteurs* » plus haut
[58] Cf. paragraphe « *2.5.7.6 Comportement N°3 : liens inter-agents* » plus haut

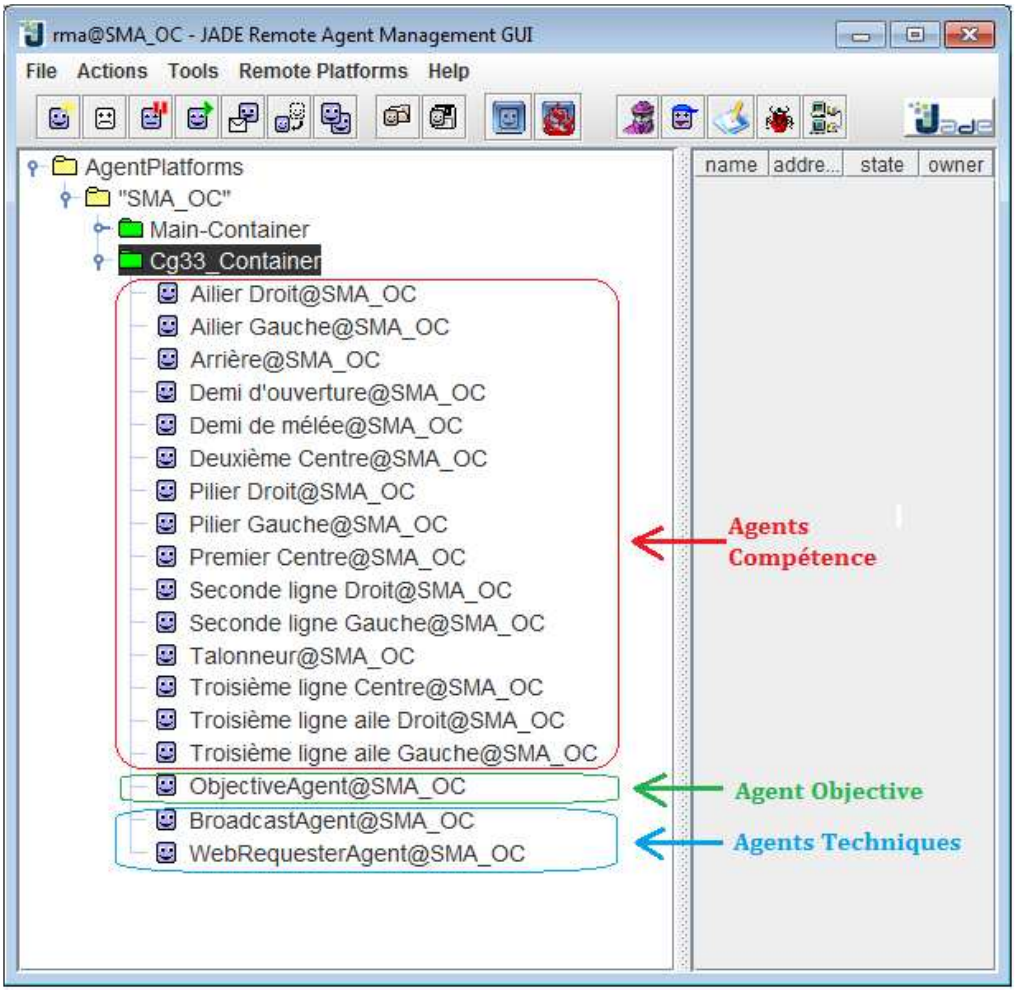


Figure 39 Simulation globale : agents présents dans JADE au lancement de simulation

3.14.3.2 Résultats d'établissement de liens inter-Acs

Nous avons déjà en termes de modélisation[59] et d'implémentation[60] que le comportement d'établissement de liens inter-agents peut être déclenché de 2 manières :

1. Suite à réception d'un évènement : requête utilisateur.
2. Suite à déclenchement d'une règle comportementale, donc de manière automatique & autonome.

Dans cette partie simulation (voir figure 40), dans le second cas, on constate l'arrivée de 15 mails de proposition d'établissement de lien, un par AC.

[59] Cf. paragraphe « *2.5.7.6 Comportement N°3 : liens inter-agents* » plus haut
[60] Cf. paragraphe « 3.11.3 Cas N°2 : identification récurrente des liens » plus haut

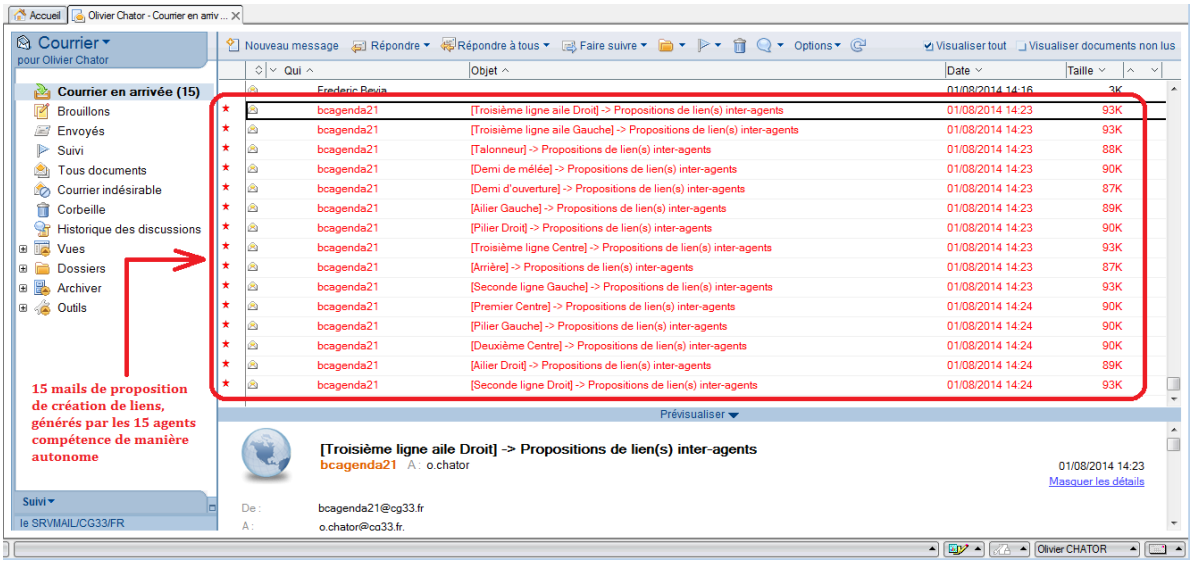


Figure 40 Simulation globale : mails générés automatiquement par les ACs

En créant une matrice de tous les coefficients de proximité reçus dans ces différents mails, on obtient le tableau 23.

IMPORTANT : Les valeurs fournies sont à considérer comme étant « vues depuis les colonnes vers les lignes », c'est-à-dire des compétences en colonne vers celles en ligne. Cet avertissement est essentiel à bien intégrer pour bien comprendre le point N°5 dans l'analyse qui suit le tableau.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	100	98,46	98,49	98	98	95,91	96,42	96,42	97,31	96,23	94,2	94,2	94,2	94,2	96,54
2	95,81	100	95,81	95,36	95,36	94,34	94,84	94,84	96,77	96,23	94,2	94,2	94,2	94,2	96,54
3	98,49	98,46	100	98	98	95,91	96,42	96,42	97,31	96,23	94,2	94,2	94,2	94,2	96,54
4	100	100	100	100	100	97,9	98,42	98,42	99,03	97,95	95,89	95,89	95,89	95,89	98,27
5	100	100	100	100	100	97,9	98,42	98,42	99,03	97,95	95,89	95,89	95,89	95,89	98,27
6	97,85	99,45	97,85	97,89	97,89	97,9	100	99,47	99,03	97,95	95,89	96,42	96,42	95,89	98,27
7	98,39	99,45	98,39	98,42	98,42	100	95,89	100	99,03	97,95	95,89	96,42	96,42	95,89	98,27
8	97,85	99,45	97,85	97,89	97,89	98,95	99,47	100	99,03	97,95	95,89	96,42	96,42	95,89	98,27
9	97,2	99,34	97,2	96,94	96,94	96,44	96,94	96,94	100	100	96,52	97,05	97,05	96,52	98,92
10	96,13	98,24	96,13	95,89	95,89	95,39	95,89	95,89	100	100	96,52	97,05	97,05	96,52	98,92
11	96,13	98,24	96,13	95,89	95,89	95,39	95,89	95,89	98,6	98,6	100	97,37	97,37	100	100
12	96,13	98,24	96,13	95,89	95,89	95,91	96,42	96,42	99,14	99,14	97,37	100	100	97,37	99,46
13	96,13	98,24	96,13	95,89	95,89	95,91	96,42	96,42	99,14	99,14	97,37	100	100	97,37	99,46
14	96,13	98,24	96,13	95,89	95,89	95,39	95,89	95,89	98,6	98,6	100	97,37	97,37	100	100
15	96,13	98,24	96,13	95,89	95,89	95,39	95,89	95,89	98,6	98,6	97,58	97,05	97,05	97,58	100

Tableau 23 : Simulation globale : coefficients de proximité

Tout joueur évoluant sur un poste (une compétence **Ci**) dispose, bien entendu de manière plus ou moins maitrisée, des savoir-faire élémentaires **Ce** qui permettent de la caractériser et de l'exercer (sur un terrain de rugby dans notre simulation). Un corolaire de cette affirmation est que plus la valeur du coefficient de proximité entre deux postes $\mathbf{Ci_1}$ et $\mathbf{Ci_2}$ sera élevée, plus les joueurs y évoluant seront interchangeables, c'est à dire pouvant aussi bien exercer $\mathbf{Ci_1}$ que $\mathbf{Ci_2}$. Ceci étant posé, regardons de plus prêt les chiffres fournis dans le tableau 23. Si l'on s'y connait un tout petit peu en rugby, on peut constater au moins de manière intuitive qu'il est globalement cohérent avec la réalité des faits. Cependant, plusieurs constats, illustrés par typologie ici, peuvent nous interroger :

1. La proximité des 3 premiers postes (2 piliers + 1 talonneur) avec les 2 postes de secondes lignes (n° 4 et 5) est de 100%. Cela voudrait dire que les joueurs évoluant à ces postes sont interchangeables. Il en est de même pour les postes de demi de mêlée (n° 9) et de demi d'ouverture (n° 10).
2. Si les ailiers (n° 11 et 14), la paire de centres (n° 12 et 13) ou bien celle de troisièmes lignes ailes (N° 6 et 7) semblent interchangeables, ce ne semble pas être le cas pour les 2 piliers. La valeur dans ce dernier cas est de 98,49%. De même, Pour la compétence troisième ligne centre (n° 8), l'interchangeabilité est effective avec le poste de troisième ligne aile droit (n° 7), mais pas avec celui de troisième ligne aile gauche (n° 6) où la valeur du coefficient est de 99,47 %.
3. La valeur du coefficient de proximité entre 2 postes peut varier selon « le point de vue où l'on se place ». Par exemple, la compétence de troisième ligne centre (n°8) a calculé un coefficient de proximité de 99,47% pour exprimer sa liaison avec le troisième ligne aile gauche (n°6). Inversement, la valeur calculée par le troisième ligne aile gauche (n°6), pour cette même relation, est de 98,95%. Le constat peut aussi être renouvelé en considérant les liens entre arrière (n° 15) et ailiers (n° 11 et 14), ainsi que la valeur calculée par les 3 premiers postes (N° 1,2 et 3) pour décrire leur liaison avec les secondes lignes (n° 4 et 5). De même, les secondes lignes ont calculé une valeur de 98% pour exprimer leur lien avec les piliers, et de 95,36% pour exprimer celui avec le talonneur (n° 2).

Analysons, à travers des exemples, les données environnementales de simulation, stockées en base de données, que nous avons créées pour les agents. Prenons aussi en compte les éléments fournis au tableau 21[61]. L'analyse nous montre que dans cet environnement de simulation :

1. Le poste 2 partage 3 mots clés sur 5 avec les postes 4 et 5.
2. Les postes 1, 3 ont 5 mots clés identiques aux postes 4 et 5.
3. Les compétences élémentaires **Ce** exercées dans les postes 1,2 et 3 sont toutes inclues dans la liste de celles exercées pour les postes 4 et 5.
4. Les postes 4 et 5 disposent de la **Ce** « *Détente pure (pour les sauteurs)* » en plus de celles partagées avec les postes 1,2 et 3.

[61] Cf. paragraphe « *3.14.1 Contexte* » plus haut

5. Le calcul du coefficient de proximité possède un bais lié aux arrondis de valeurs calculées.

Le constat n° 1 s'explique par le fait que nous avons créé des données environnementales où les postes 1,2 et 3 ont quasiment les mêmes caractéristiques (mots clés, attributs internes, **Ce**) que les postes 3 et 4. C'est le cas en particulier au niveau des mots clés et des **Ce**, où ceux des postes 1 et 3 sont inclus dans celles des postes 4 et 5. En conclusion : il convient de prêter une attention toute particulière lors de la création des compétences et de leur description, ce qui nécessite une expertise particulière et collant au plus prêt du contexte métier d'utilisation.

Le constat n°2 s'explique par la méthodologie du calcul du coefficient de proximité. En se référant à la figure 12[62], on constate que le parcours récursif de l'arbre de base de données inclue la recherche de similarités sur la valeur même des données d'évaluation des **Ce** au sein des projets dans son algorithme. On a vu plus haut que les postes 1 et 2 possèdent des caractéristiques et des **Ce** identiques. Dans le cas spécifique de notre simulation, les données d'évaluation pendant les projets ont été générées aléatoirement[63]. C'est cette génération aléatoire qui introduit un biais dans la valeur finale obtenue pour le coefficient, puisque selon le poste qui fait le calcul, le nombre de données d'évaluation identiques des **Ce** entre les postes varie. En conclusion : il conviendra probablement d'exclure ces données d'évaluation de la méthodologie globale de calcul du coefficient de proximité.

Le constat n° 3 est très proche et s'explique finalement comme le premier puisque la seule différence entre les postes 4, 5 et les postes 1, 2, 3 est l'adjonction d'une **Ce** en plus pour les premiers. Nous obtenons logiquement une valeur du coefficient de proximité différente, selon que le calcul est effectué par les postes 4 et 5 ou bien par les postes 1,2 et 3. La conclusion est alors identique à celle présentée pour expliciter le constat n° 1.

L'analyse des résultats de cette partie de notre simulation nous montre tout l'intérêt, et les limites, de notre méthodologie de calcul du coefficient de proximité. Bien que ce ne soit pas implémenté au moment de la rédaction de ce mémoire, nous pouvons tout de même proposer une autre utilisation possible et concrète de ce coefficient. Il pourrait nous permettre d'identifier des acteurs pouvant exercer d'autres **Ci** que celles qu'ils maitrisent, moyennant par exemple le suivi de formations spécifiques. Ceci pourrait se concrétiser par analyse automatique (à déclenchement sur valeur de seuil de notre coefficient) des **Ce** communes et distinctes (donc à acquérir) entre deux **Ci.**

[62] Cf. *Mécanismes d'apprentissage* au paragraphe « *2.5.7.6 Comportement N°3 : liens inter-agents* » plus haut
[63] Cf. paragraphe « *3.14.1 Contexte* » plus haut

3.14.3.3 Résultats de simulation globale

Le scénario de simulation a été décrit en détail au paragraphe « *3.14.2.1 Spécifications* ». En synthèse, il s'agit d'automatiser les actions que mèneraient M. COTTER dans le contexte de une, puis finalement cinq, nouvelles saisons du TOP 14 de rugby. Pour chacune des saisons, la génération repose sur la duplication de la structure (liste de 27 matchs) de la saison 2013/2014 de référence. On rappelle que pour chaque match d'une saison, nous avons créé :

1. Les demandes relatives à l'exercice des 15 postes nécessaires.
2. Les candidatures de tous les joueurs capables d'exercer chacun des 15 postes.
3. L'opération d'affectation du poste à un joueur en utilisant l'apprentissage.
4. Les évaluations des **Ce** pour chaque poste exercé.

Elément	**Valeur**
Nombre total de saisons	5
Nombre total de matchs (projets)	135
Nombre total de postes (compétences) utilisés	2025
Nombre total de formulaires d'évaluation post-match générés	2025
Nombre de souvenirs générés par agent (poste), à raison de 2 par match : a) Candidature à participation à match b) Sélection joueur	270

Tableau 24 : Simulation globale : chiffres globaux

Le simulateur automatique que nous avons développé gère bien les 15 compétences rugbystiques nécessaires à la concrétisation de chaque projet (match). Cependant, pour faciliter la présentation des résultats dans ce mémoire de thèse, nous prendrons l'exemple de l'une d'elles, la **Ci** « *Pilier Gauche* » (joueur dossard N° 1). Les comportements décrits et résultats obtenus pour cette **Ci** particulière sont bien entendu généralisables aux 15 autres. Dans notre simulation, la compétence « *Pilier Gauche* » peut être offerte par 3 joueurs (Raphael CHAUME, Vincent DEBATY et Thomas DOMINGO. Notre **Ci** possède 7 **Ce** (cf. tableau 21[64]). Pour rappel :

1. Pourcentage de passes réussies.
2. Pourcentage de plaquages réussis.
3. Nombre de ballons pris à l'adversaire.
4. Nombre de ballons touchés.
5. Capacité de pénétration (nombre de mètres avancés avec le ballon en mains).
6. Poussée maximale (pour les mêlées).
7. Nombre de kilomètres parcourus rapporté au temps de jeu.

[64] Cf. paragraphe « *3.14.1 Contexte* » plus haut

Il est important de rappeler ici que lors de la simulation des demandes de **Ci** au système par M. COTTER, un poids est attribué à chacune de ces 7 **Ce**. Ce poids est généré essentiellement de manière aléatoire, selon la méthodologie déjà présentée au tableau 22[65]. Ces variations des poids simulent les évolutions de contexte des matchs, à travers des niveaux d'importance attribués par l'entraineur à chaque **Ce**. Par exemple, lors de la demande au système du nom d'un joueur pouvant exercer la compétence « *Pilier Gauche* », dans le contexte d'un match contre l'équipe du Racing Club Toulonnais (équipe très physique), il peut être judicieux d'attribuer un poids plus élevé à la **Ce** « *Poussée maximale (pour les mêlées)* » qu'à la Ce « *Nombre de ballons pris à l'adversaire* ».

Suite à une demande envoyée par M. COTTER au système, une proposition de joueurs pour l'exercice de la **Ci** « *Pilier Gauche* » est renvoyée. Contrairement à ce qu'il ne manquera pas de se passer dans le monde réel compte tenu des différents contextes des matchs et des facteurs d'appréciation humains, notre simulateur valide systématiquement la proposition de « meilleur candidat » du système. Nous sommes tout à fait conscients de la limite de l'opération, elle nous apparait évidente. Toutefois, nous devons bien constater concrètement que les résultats observés et présentés ici, concernant la simulation des activités de M. COTTER pour une puis 5 saisons, semblent répondre à nos objectifs de départ : réciprocité, équité, montée en compétence, etc. Effectuer une validation systématique des choix du système revient à observer les évolutions de son comportement, comme si nous lui faisions finalement entièrement « confiance » pour mener notre équipe de l'ASM à la victoire en TOP14 ! Plus sérieusement, nous verrons plus loin que les résultats obtenus sont significatifs en regard des buts que nous nous étions fixés au départ de ces travaux de thèse.

Comme nous l'avions proposé pour l'attribution de poids aléatoires aux **Ce** lors de l'émission de la requête vers le système, l'évaluation de l'exercice d'une **Ci** par un joueur, dans le contexte d'un match donné et à son issue, repose aussi sur la base de notes attribuées aux **Ce**. Dans notre simulateur, les évaluations sont également générées de manière aléatoire, en respectant la même méthodologie que celle que nous avons décrite pour l'attribution des poids initiaux. Ces évaluations post-réalisation seront consolidées dans le but de calculer une note globale, attribuée à chaque joueur. Cette note peut être vue comme étant une sorte « d'indicateur de performance » qui va évoluer. Sa valeur change en fonction des retours d'évaluation prononcés par M. COTTER suite à la participation de chaque joueur à un match (i.e. à l'exercice des **Ci** dans les projets).

Enfin, nous cherchons à vérifier finalement dans notre simulation automatique que l'algorithme proposé[66] pour optimiser la sélection des joueurs fait sens. 4 scénarios « possibles » sont explorés :

a) **Scénario 1** : La sélection du joueur repose une fois sur deux sur l'élection de celui qui a la note globale la plus faible. On procède, pour la seconde fois, à une sélection

[65] Cf. paragraphe « *3.14.1 Contexte* » plus haut
[66] Cf. *Mécanismes d'apprentissage* au paragraphe « *2.5.7.5 Comportement N°2 : sélection d'acteurs* »

aléatoire de joueur. Ce que l'on propose ici consiste à privilégier en partie la sélection d'un joueur ayant la note la plus faible. On souhaite finalement qu'il monte en compétence et « s'améliore » par une participation accrue aux matchs (projets).

b) **Scénario 2** : La sélection du joueur repose sur l'élection de celui qui a le moins participé aux matchs par le passé. On ne prend plus en compte les notes globales des joueurs. On cherche cependant à respecter la plus grande équité possible entre eux, en termes de nombre de sélections, donc de matchs joués par chacun d'eux.

c) **Scénario 3** : La sélection du joueur repose sur un choix entièrement aléatoire. Il n'y a dans ce scénario aucune considération du nombre de sélections antérieures ou de la note globale des joueurs. Concrètement, ce scénario « déresponsabilise » l'entraineur dans la mesure où il donne complètement la main au hasard. On voit déjà bien toutes les limites induites par ce scénario. Cependant, les résultats obtenus sont surprenants.

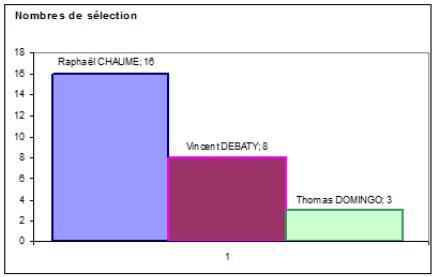
d) **Scénario 4** : C'est celui qui reprend notre proposition théorique. La sélection du joueur repose une fois sur deux sur l'élection de celui qui a la note globale la plus élevée. On procède, pour la seconde itération, à une sélection aléatoire de joueur. Ce scénario assure un équilibre entre performance et équité des chances.

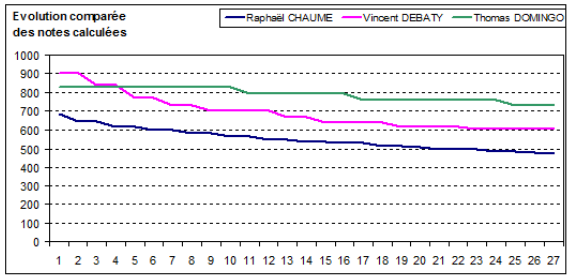
Au moment du lancement de chacun de nos scénarios, nos Agents Compétence n'ont pas encore de « souvenirs » mémorisés. Le contexte initial d'exécution est identique pour chacun d'eux. Il contient uniquement les données des 27 matchs du TOP 14 de la saison 2012/2013. Tout au long des traitements automatisés, notre simulateur fournit des traces d'exécution. Par exemple (version condensée) :

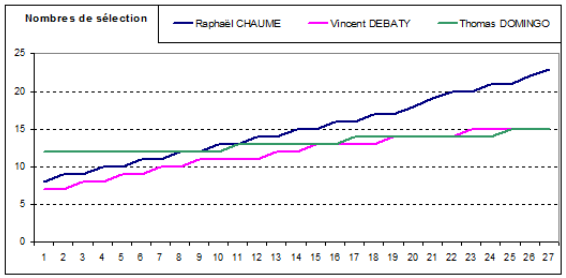
```
------------------Debut (Pilier Gauche) ---------------------
Trouvé (27) affectation(s) déjà faite(s) pour Agent (1)
Pour l'Agent (1), (Raphael CHAUME) a déjà été sélectionné (8) fois
Pour l'Agent (1), (Vincent DEBATY) a déjà été sélectionné (7) fois
Pour l'Agent (1), (Thomas DOMINGO) a déjà été sélectionné (12) fois
La liste des candidats contient (3) élément(s)
 (DEBUT) Liste d'acteurs Cible
    Acteur (Raphael CHAUME), note calculée (688.0)
    Acteur (Thomas DOMINGO), note calculée (826.4166666666666)
    Acteur (Vincent DEBATY), note calculée (903.0)
(FIN) Liste d'acteurs Cible
Selected Best User -> (Thomas DOMINGO)
Résultat                                                        final
(<reponses><reponse><num>1</num><memoryCode>2</memoryCode><userid>8</u
serid><firstname>Thomas</firstname><lastname>DOMINGO</lastname><servic
e>ASM</service></reponse></reponses>)
------------------Fin (Pilier Gauche) ---------------------
```

Les résultats d'exécution des différents scenarios de simulation sont présentés sous une forme simplifiée et schématique au sein des figures 41 et 42 qui suivent.

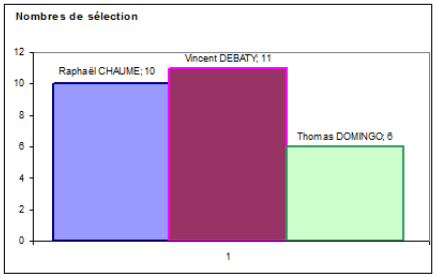
Scénario 1 : En privilégiant le joueur ayant la note la plus basse + hasard 1 fois sur 2

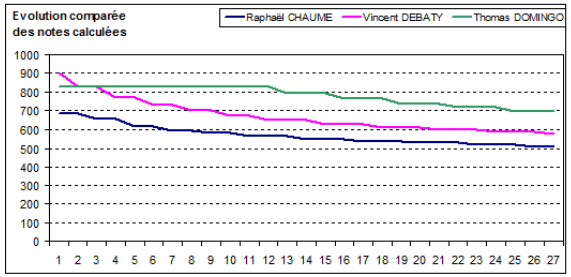


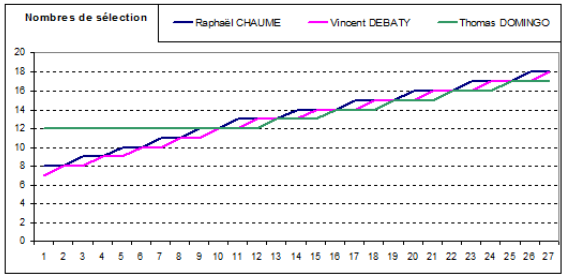




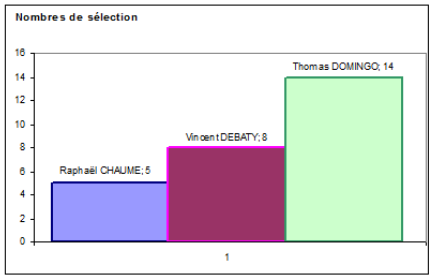
Scénario 2 : En privilégiant le joueur ayant été le moins sélectionné dans le passé

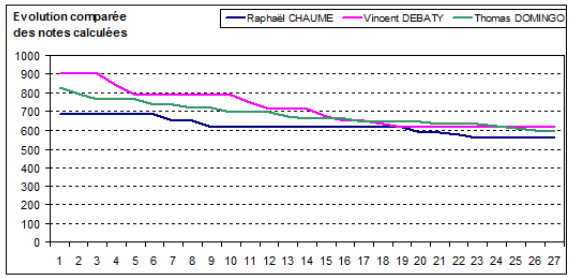


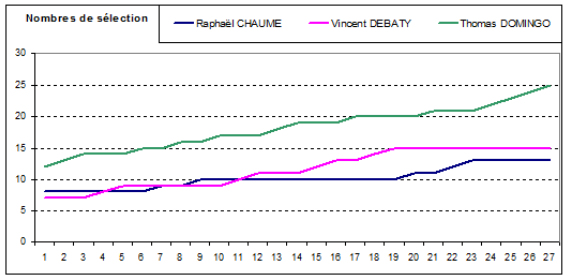




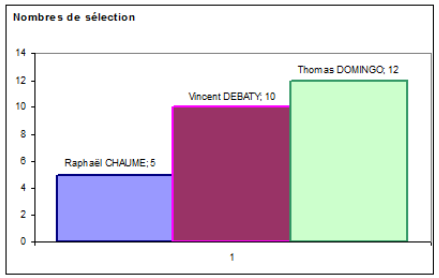
Scénario 3 : En choisissant le joueur entièrement au hasard

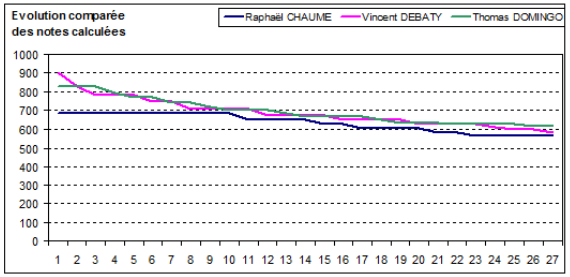






Scénario 4 : En privilégiant la note la plus élevée + hasard 1 fois sur 2





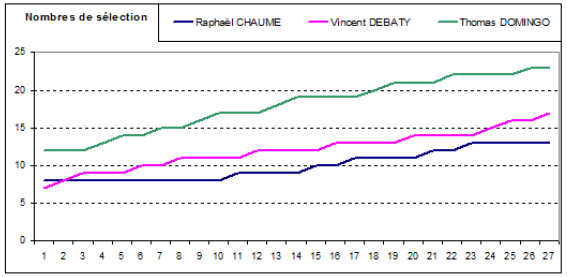
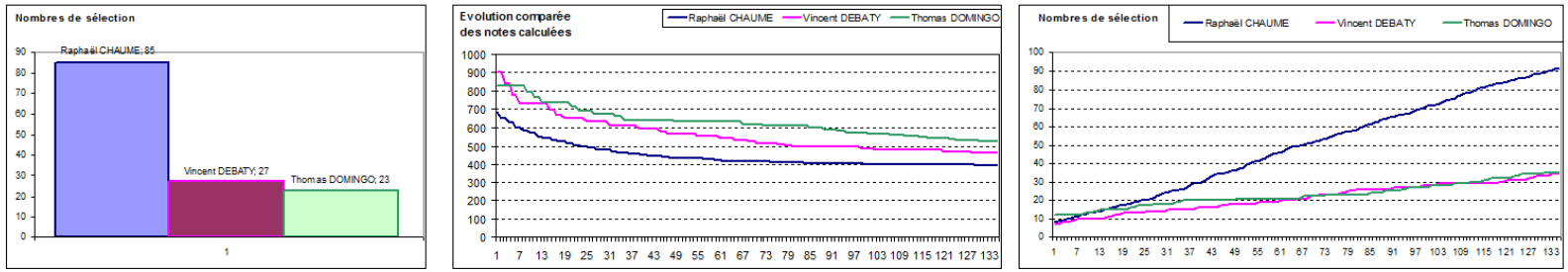
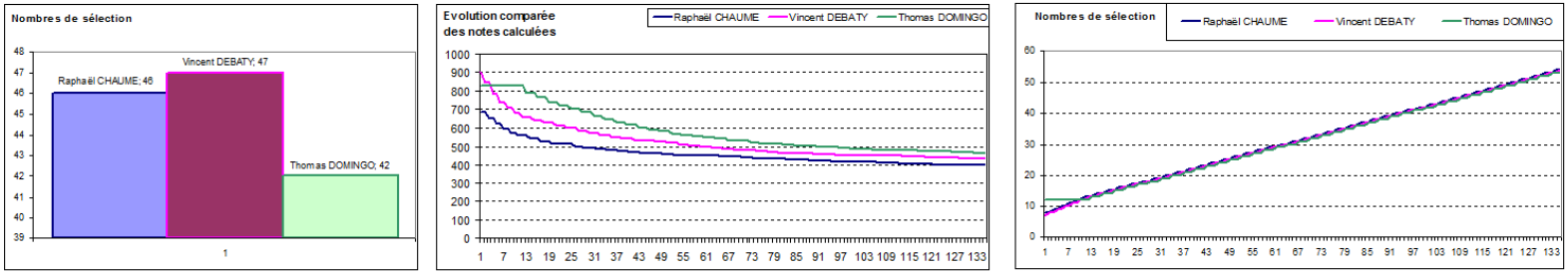


Figure 41 Résultats de simulation globale – 1 saison

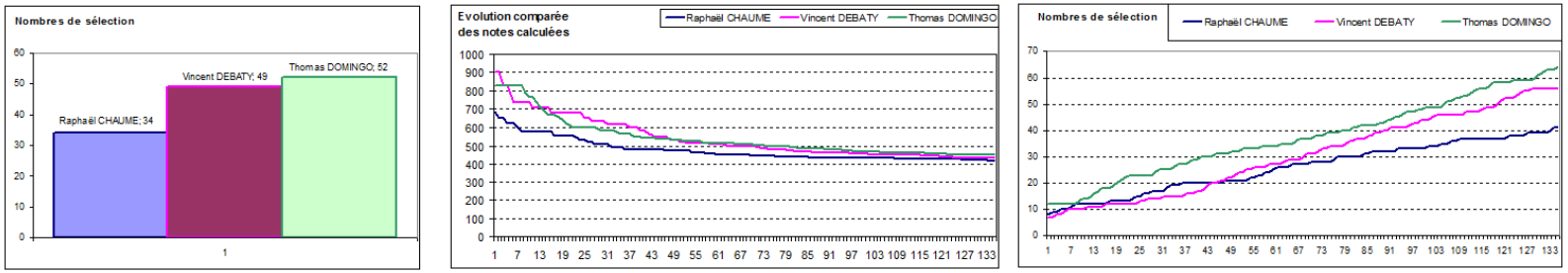
Scénario 1 : En privilégiant le joueur ayant la note la plus basse + hasard 1 fois sur 2



Scénario 2 : En privilégiant le joueur ayant été le moins sélectionné dans le passé



Scénario 3 : En choisissant le joueur entièrement au hasard



Scénario 4 : En privilégiant la note la plus élevée + hasard 1 fois sur 2

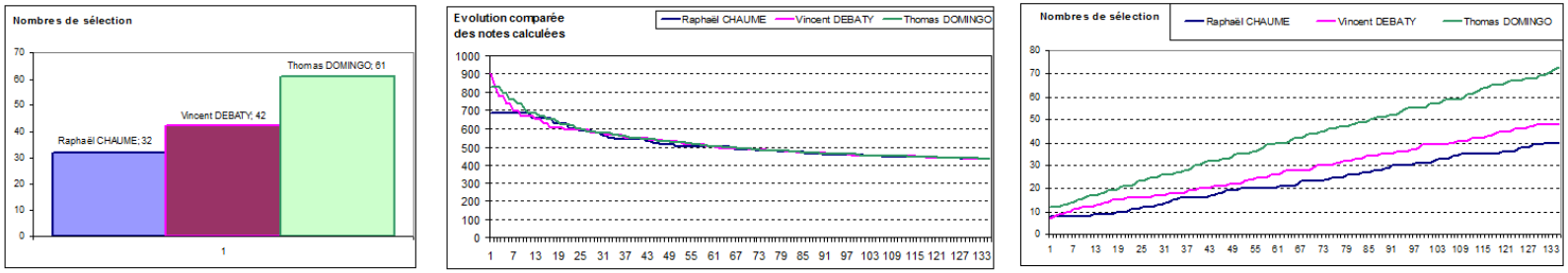


Figure 42 Résultats de simulation globale – 5 saisons

L’analyse des figures 41 (1 saison) et 42 (5 saisons) nous permettent de proposer les conclusions suivantes :

1. La simulation sur 5 saisons du TOP 14 vient corroborer et valider la tendance générale amorcée dans la simulation de génération des matchs pour une seule saison.
2. Les notes calculées (indicateurs de performance) des utilisateurs tendent à converger, ce qui nous conduit à dire que la montée progressive en compétence des joueurs est validée à travers le fonctionnement de l’outil.
3. La convergence des indicateurs de performances des joueurs est plus ou moins atteignable rapidement selon le scénario choisi.
4. Le scénario le plus performant révélé par la simulation est celui que nous avons proposé dans l’étude théorique de cette thèse. Il se compose d’une alternance entre sélection du joueur possédant la note globale la plus élevée et sélection aléatoire.
5. La mise en œuvre de la simulation globale (1 ou 5 saison) valide l’atteinte des objectifs attendus dans le cadre de nous travaux de thèse. Elle propose à la fois une restitution de joueurs ayant un niveau de compétence élevé, tout en offrant aux autres joueurs la capacité d’atteindre eux-mêmes ce niveau d’expertise.

On voudra bien noter que le scénario se rapprochant le plus de l’optimal, en matière de rapidité de convergence, est bien le N°3 de sélection complètement aléatoire des joueurs. Nous pourrions en tirer l’enseignement que la prise en compte des seuls critères de notation ou de nombre de sélection des joueurs ne sont pas suffisants compte tenu de nos objectifs. L’introduction d’une part de hasard dans la méthode de sélection proposée au scénario 4, semble d’ailleurs bien vérifier cette supposition.

Plus largement, le bon déroulement de cette phase de simulation globale valide la dimension de généralisation possible à tout domaine d’activité professionnel. On rappelle que notre système a été prévu, pensé et développé au départ pour répondre aux attentes du CG33 afin d’optimiser la gestion des projets de développement durable. Un environnement rugbystique n’a que peu de choses en commun, voir aucune, avec un environnement dans lequel on met en œuvre des projets de développement durable. Nous vérifions encore que notre système est bien apprenant de par les échanges interactifs simulés avec M. COTTER. Nos agents compétence apprennent des expériences (matchs) passés. Ils réutilisent ce qu’ils ont appris pour optimiser la génération de liens inter-agents, la création de nouveaux projets par candidature des compétences elles-mêmes, ou bien encore la sélection des acteurs.

Conclusion & perspectives

«Clore des cycles, fermer des portes, finir des chapitres - peu importe comment nous appelons cela, l'important est de laisser dans le passé les moments de la vie qui sont achevés»

Paulo Coelho

Discussion

Quel que soit le secteur d'activité concerné, l'objectif de ces travaux de thèse est de proposer la mise en place d'un véritable pilotage des projets par une vision complètement orientée compétences. Il s'agit de finalement de se recentrer réellement sur le besoin, notre problématique étant surtout de nous centrer davantage sur le « quoi », en nous débarrassant d'apriori sur le « qui » va le réaliser. Pour atteindre cet objectif, nous déclinons le concept même de compétence sous la forme d'agents logiciels autonomes, apprenants, et qui évoluent au sein d'un système multi-agents. Cette proposition est novatrice par rapport aux usages actuels où les compétences sont simplement décrites par des règles comportementales qui définissent les actions des agents.

Dans nos travaux, les compétences SONT des agents à part entière. Chacun d'eux cherche à réaliser 4 (à ce stade de nos travaux) buts principaux :

1. S'auto-définir via l'identification d'une liste pondérée de compétences élémentaires
2. Restituer une liste d'acteurs humains capable de le concrétiser au sein d'un projet
3. Etablir des liens (relations) avec d'autres agents compétence
4. Candidater à une participation à de nouveaux projets

Dans l'objectif d'autodéfinition, les pondérations des compétences élémentaires peuvent évoluer par apprentissage. Ces poids définissent le niveau d'importance accordé à chacune d'elles dans la définition de la compétence globale.

L'utilisation d'ACs palie à certaines lacunes pendant la phase d'expression du besoin lors de la création de nouveaux projets. Elle apporte (entre autres) une réponse à deux problématiques :

- Les utilisateurs n'ont souvent qu'une vision partielle de toutes les compétences dont ils ont besoin pour créer leurs projets.

- Les outils de gestion de projet fournissent une liste statique de compétences à mettre en œuvre. L'évolution de cette liste s'effectue suite à une action manuelle, donc active. C'est le cas, par exemple, d'une intervention sur le paramétrage de ces outils. On notera que ces actions de mise à jour peuvent être génératrices de coûts si l'on doit faire appel à des experts. Elles entrainent également une perte d'efficacité du processus global de production lors de ces phases de mise à jour : quelle stratégie de transition entre les deux versions mettre en œuvre ?

L'introduction des ACs propose une réelle amélioration compte tenu de la disponibilité de mécanismes d'évolution dynamique et d'apprentissage. Il n'y a plus, ici, de vision statique de la liste des compétences constituant les projets. A l'opposé, il ya génération dynamique de cette liste. Le système que nous avons réalisé colle toujours au plus près des évolutions du domaine professionnel pour le compte duquel il s'exécute. Si l'on raisonne à un niveau plus global, le système apprend des requêtes des utilisateurs, grâce au comportement proactif des agents qui le composent.

Considérons également le niveau de l'Entreprise au sens global du terme. L'utilisation d'ACs peut apporter son concours aux processus l'optimisation du management des compétences. Concrètement, l'observation des activités se déroulant à l'intérieur du système peut fournir des informations très utiles en s'appuyant sur des constats tels que:

- L'émergence de compétences « clé » sur lesquelles il vaudra mieux investir, par exemple en terme de formation, puisque souvent sollicitées. La réciproque est également vraie : l'identification de compétence peu (ou plus) utilisées.
- L'apparition de complexités jugées comme trop importantes pour certains projets.
- L'apparition de compétences transverses à de multiples domaines d'objectifs.

Poussons le raisonnement encore plus loin : ces constats peuvent être générateurs d'évolutions organisationnelles au sein même de l'Entreprise. On citera, par exemple, la création de services de pools d'expert dans tel ou tel domaine de compétences et/ou telle ou telle compétence partageable entre plusieurs unités de l'Entreprise ou d'un Groupe.

Conclusion

Le Conseil Général de la Gironde, à travers cette thèse de doctorat, a souhaité créer un outil informatique innovant dont l'objectif est d'accroitre l'efficacité du partage de compétences dans les projets relevant des thématiques du Développement Durable. La concrétisation de ce souhait est l'outil collaboratif *Bourse aux Compétences de Développement Durable*, opérationnel au moment de la rédaction de ce mémoire. Pour parvenir à ce résultat, d'un point de vue théorique, il a été nécessaire de proposer une modélisation originale du concept même de compétence. Pour ce qui concerne l'implémentation, le choix s'est porté sur l'utilisation des technologies multi-agents (Ferber, 1999) (Jennings, 1998). Ces dernières offrent de nombreux avantages reconnus lors de la constitution de systèmes autonomes, apprenants et évolutifs.

En synthèse[67], nous proposons d'un point de vue théorique que toute compétence puisse être représentée informatiquement sous la forme d'une entité logicielle unique, apprenante et autonome, nommée « Agent Compétence » (AC). Au sein d'un SMA de type évènementiel, cet agent possède un cycle de vie, un modèle comportemental, et des ressources. Il cherche à réaliser à minima 4 buts principaux :

- Constituer sa propre définition.
- Restituer une liste d'acteurs humains capable de le concrétiser au sein d'un projet.
- Etablir des liens (relations) avec d'autres agents.
- Se porter candidat à une participation à de nouveaux projets.

Dans nos travaux, les ACs ont été implémentés dans un SMA Open Source JADE (Bellifemine, 1999). Nos résultats expérimentaux tendent à démontrer une convergence vers un point d'équilibre pour les algorithmes d'apprentissage proposés. Cette affirmation est vérifiable pour les comportements visant à atteindre les buts de restitution de listes d'acteurs et de création de liens inter-ACs.

Nous démontrons dans cette thèse que la modélisation des compétences sous la forme d'AC fait sens et impacte les habitudes de définition des projets. Cette définition ne suit plus un raisonnement orienté acteur humain pour l'exercice de telle ou telle compétence. Un réel recentrage sur les compétences, lors de la définition du besoin, est mis en avant. Il n'y a aucun préjugé sur les acteurs qui vont les concrétiser.

Au fil du temps, l'observation et l'analyse des activités relatives à ces projets dans notre système peut induire des impacts sur le management des compétences au sein de l'Entreprise. Par exemple, il sera possible de constater l'émergence de compétences « clé », la survenue de « goulots d'étranglement » pour la couverture de certains besoins, l'apparition d'une transversalité de certaines compétences sur les domaines d'objectifs, etc.…Le lancement de plans de formation de personnels, voir le déclenchement de recrutements, peut être dicté par la nécessité des projets et devenir l'une des conséquences de ces observations.

Les ACs, à travers des outils tels que la BCDD, stimulent et encouragent la collaboration des acteurs. Toutes les compétences partagées sont mises en visibilité de tous et sont présentées sans qu'il soit fait mention de qui peut la concrétiser. D'un point de vue sociétal, il est évident dans un tel système que la notion de « *réciprocité des échanges* » devient alors prépondérante. Si l'on extrapole encore davantage, l'utilisation d'ACs peut permettre l'émergence d'un nouveau modèle organisationnel et relationnel dans le monde du travail. Imaginons, par exemple, un système fédéré — qu'il soit effectivement unique ou bien constitué concrètement de multiples systèmes unitaires reliés entre eux ou non — transverse à de multiples organisations, secteurs professionnelles et/ou branche d'activité.

[67] Cf. paragraphe « *2.5.2 Proposition* » définissant un AC au 2

Considérons, par exemple, une compétence de « demi de mêlée » pour revenir à notre jeu de rugby. L'observation de notre système fédéré peut nous montrer l'émergence d'une certaine transversalité de cette compétence dans un système fédéré et partagé par plusieurs clubs de rugby. En considérant nos travaux et en généralisant à tout secteur d'activité professionnel, il devient alors pertinent de se poser les questions suivantes :

- Peut-on, de nos jours, imaginer un modèle organisationnel et sociétal d'entraide, basé sur le volontariat, où les acteurs mettent en œuvre leur compétence dans de multiples Entreprises de secteurs d'activités aussi diverses que variés?
- Avant de le rejeter tout à fait, quels seraient les impacts d'un tel modèle :
    a. A quelle Entreprise et structure hiérarchique les acteurs seraient ils rattachés ?
    b. Comment les rémunérations seraient elles gérées ?
    c. Selon quels cadres juridiques ?
    d. Que les acteurs humains ont-ils à y gagner ?
    e. Etc.

Bien entendu, la liste des questions fournies ci-dessus est loin d'être exhaustive. Il nous apparait alors possible de pouvoir remettre en cause l'intégralité de nos modèles sociétaux et relationnels dans le monde du travail, tout en accroissant l'efficacité des échanges humains, et par la même celle des processus de production.

Perspectives

Si cette thèse pose les bases de ce que sont les ACs, le champ de recherche reste largement ouvert. Il concerne de nombreux aspects parmi lesquels on peut identifier certaines optimisations et/ou compléments à apporter. On citera, pour illustrer : la nature même et la définition de ce qu'est un AC, les comportements qui le caractérisent, les algorithmes d'apprentissage qu'il met en œuvre, son cycle de vie, son modèle comportemental…

L'optimisation de la méthodologie et du mode de calcul des coefficients de proximité et de pertinence, ainsi que leur intégration complète au sein de mécanismes d'apprentissage dédiés peut aussi être citée. Par exemple, l'optimisation de la méthodologie de construction du coefficient de proximité pourrait passer par l'intégration de techniques d'analyse sémantique (et/ou syntaxiques) sur les valeurs des attributs internes des agents, etc. A ce jour, ces coefficients sont seulement stockés en mémoire pour usage ultérieur alors que leur mode de calcul nous semble pouvoir être optimisé et affiné par apprentissage.

Un autre aspect est celui de la vérification en volume des résultats expérimentaux. Au moment de la rédaction de ce mémoire, la BCDD est concrètement en production avec 110 ACs et 61 utilisateurs (membres du réseau des Agendas 21). Cependant, nous ne disposons pas encore d'assez d'évaluations de projets et de « trafic » dans l'outil pour en tirer des conclusions définitives. On remarquera toutefois qu'une analyse des données déjà disponibles à ce jour semble montrer une convergence avec celles dont on dispose via les résultats expérimentaux obtenus par simulation automatisée en laboratoire.

Nous avons vu plus haut que *l'utilisation d'ACs peut permettre l'émergence d'un nouveau modèle organisationnel et relationnel dans le monde du travail*. Bien que cet aspect ne soit pas directement lié aux technologies de l'information, il ouvre un champ d'investigation terriblement vaste en sciences sociales. Une telle affirmation peut effectivement être à l'origine de bouleversements d'ampleur, tant organisationnels que comportementaux, au sein des Entreprises. Mais finalement, serait-ce une si mauvaise chose ?

Henri Laborit (Labori, 1985) ne dit-il pas : « *Ce n'est pas l'Utopie qui est dangereuse, car elle est indispensable à l'évolution. C'est le dogmatisme, que certains utilisent pour maintenir leur pouvoir, leurs prérogatives et leur dominance* » ?

Valorisation de la Recherche

Liste des publications effectuées pendant cette thèse

Modélisation Multi-Agents Centrée sur les Compétences pour la Collaboration des Acteurs dans les Projets de Développement Durable

Proceedings conférence AIM 2012
27 mai 2012
http://aim.asso.fr/index.php/mediatheque/summary/26-aim-2012/864-modelisation-multi-agents-centree-sur-les-competences-pour-la-collaboration-des-acteurs-dans-les-projets-de-developpement-durable

Multi-agent System for Skills Sharing in Sustainable Development Projects

Proceedings conférence COGNITIVE 2013
27 mars 2013
http://www.thinkmind.org/download.php?articleid=cognitive_2013_1_40_40129
IARA - Conférence Cognitive 2013 - Publication AWARD
This paper has been selected as one of the "Best Papers" based on the reviews of the original submission, the camera-ready version, and the presentation during the conference. For the awarded papers, a digital award has been issued in the name of the authors.
http://www.iaria.org/conferences2013/awardsCOGNITIVE13/cognitive2013_a1.pdf

Agents compétence autonomes pour l'amélioration de la collaboration entre acteurs humain

Proceedings conférence CIA 2014
23 Mai 2014
http://gccpa.espe-aquitaine.fr/images/interventioncia2014/CHATOR-SALOTTI-Axe2.pdf

Improving Skills Management using Objectives within a Multi-Agent System

Proceedings conférence COGNITIVE 2014
30 mai 2014
http://www.thinkmind.org/index.php?view=article&articleid=cognitive_2014_1_30_40050

Human cooperation improvement using autonomous skill agents

International Journal On Advances in Intelligent Systems, vol 7, n° 1&2, pp. 51-63
Catégorie des articles internationaux avec comité de lecture
15 Juin 2014-05-09
http://www.iariajournals.org/intelligent_systems/intsys_v7_n12_2014_paged.pdf

Références bibliographiques CG33

(CG33-1, 2013) CG33, “Connaître le Conseil général de la Gironde”, http://www.gironde.fr/upload/docs/application/octetstream/2013-04/guide2013bd.pdf, France, 2013

(CG33-2, 2013) CG33, “Plan Climat, Air, Energie de la Gironde 2013 – 2017”, http://www.gironde.fr/upload/docs/application/x-download/2012-12/pcet33.pdf, France, 2013

(CG33-3, 2012) CG33, “Inventer le futur économique et social”, http://www.gironde.fr/jcms/cgw_45748/inventer-le-futur-economique-et-social?hlText=%C3%A9conomie+sociale, France, 2012

(CG33-4, 2010) CG33, “Acte 2 de l’Agenda 21 de la Gironde 2010-2012 et gouvernance territoriale de développement durable“, du 25 Mars 2010, http://www.gironde.fr/upload/docs/application/x-download/2010-09/delib_acte_ii.pdf, Bordeaux, France, 2010

(CG33-5, 2010) CG33, “Comité de Pilotage des Agendas 21 Locaux“ du 12 Avril 2010, http://www.gironde.fr/upload/docs/application/x-download/2010-09/agenda21_reseau_2010-2012.pdf, Bordeaux, France, 2010

(CG33-6, 2012) CG33, “14ème Comité de pilotage du Conseil Départemental des Agenda 21 locaux en Gironde - Rapport d’activités 2011“, http://www.gironde.fr/upload/docs/application/octetstream/2012-08/rapport_activites_cda21_2012.pdf, France, 2012

(CG33-7, 2012) CG33, “Référentiel pour l’évaluation des projets d’aménagement durable“, http://www.gironde.fr/upload/docs/application/octetstream/2012-08/referentiel_amenagement_durable_cda21.pdf, France, 2012

(CG33-8, 2012) CG33, “les 9 SCoT Girondins”, http://www.gironde.fr/jcms/cgw_46649/carte-des-scot-de-la-gironde, France, 2012

(CG33-9, 2011) CG33, "Panorama des Agendas 21 locaux en Gironde", 2ème édition, avril 2011, document collectif du Conseil Général de la Gironde, disponible sur Internet au 1er avril 2012:

Références bibliographiques

(Agrawal, 1992) R. Agrawal, S. P. Ghosh, T. Imielinski, B. R. Iyer, and A. N. Swami, “An interval classifier for database mining applications”, In VLDB '92, pp. 560-573, USA, 1992.

(Dortier, 2008) J. F. Dortier, “la grande histoire de la psychologie”, Sciences humaines, hors série no 7, p. 44, France, 2008

(Aamodt, 1994) A. Aamodt, E. Plaza, “Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches”, AI Communications, Vol. 7, P 39-59, USA, 1994

(Académie Française, 1986) Académie Française, Dictionnaire, 9ème édition, Fayard, oparis, France, 1986

(ADEME, 2006) ADEME, “Bâtiment et démarche HQE” ("HQE building and methodology"), Collection Connaître pour agir, Paris, France, 2006

(AFNOR, 2011) AFNOR, “Responsabilité sociétale et ISO 26000 - Comprendre et mettre en oeuvre”, France, 2011

(Arkin, 2003) R.C. Arkin, Y. Endo, J.B. Lee, D.C. MacKenzie, and E. Martinson, “Multistrategy learning methods for multirobot systems”, Proceedings of the 2nd International Workshop on Multi-Robot Systems, pages 137-150, USA, 2003

(Aumann, 1974) R. Aumann, “Subjectivity and correlation in randomized strategies”, Journal of Mathematical Economics, Vol. 1, P 67–96, USA, 1974

(Bardou, 2011) M. Bardou, “De la stratégie à l’évaluation : des clés pour réussir un Agenda 21 local” ("From the strategy to the assessment: the keys to succeed in the elaboration of a local Agenda 21") , Collection “Références” du Service de l’Économie, de l’Évaluation et de l’Intégration du Commissariat Général au Développement Durable, Paris, France, 2011

(Bayes, 1763) T. Bayes, “An Essay towards solving a Problem in the Doctrine of Chances”, Philosophical Transactions of the Royal Society of London, Vol. 53, pp. 370-418, UK, 1763

(Becker, 2004) R. Becker, S. Zilberstein, V. Lesser, and C.V. Goldman, “Transition-independent decentralized markov decision processes”. In Journal of Artificial Intelligence Research, Vol. 22, p. 423–455, USA, 2004

(Bellifemine, 1999) F. Bellifemine, A. Poggi, and G. Rimassa, “JADE – A FIPA-compliant agent framework”, 4th International Conference on Practical Application of Intelligent Agents and Multi-Agent Technology (PAAM-99), London (UK), 1999

(Belman, 1957) R. Bellman, “Dynamic Programming”, Princeton University Press, USA, 1957

(Bernstein, 2002) D.S. Bernstein, R. Givan, N. Immerman, and S. Zilberstein, “The complexity of decentralized control of markov decision process”, In Mathematics of Operations Research, Vol. 27, Issue 4, P 819-840, USA, 2002

(Bertsekas, 1987) D.P. Bertsekas, “Dynamic Programming: Deterministic and Stochastic Models”, Prentice-Hall, USA, 1987

(Bertsekas, 1996) D.P. Bertsekas, and S. Ioffe, “Temporal differences-based policy iteration and applications in neuro-dynamic programming”, Report LIDS-P-2349, MIT, USA 1996

(Biemacki, 2003) C. Biernacki, G. Celeux, and G. Govaert, ”Choosing starting values for the EM algorithm for getting the highest likelihood in multivariate gaussian mixture models. Computational Statistics and Data Analysis 41”, 561–575, Elsevier, USA, 2003

(Borodin, 1998) A. Borodin, and R. El-Yaniv, “Online Computation and Competitive Analysis”, Cambridge University Press, UK, 1998

(Boutilier, 1995) C. Boutilier, R. Dearden, and M. Goldszmidt, “Exploiting structure in policy construction”, Proceedings of the 14th International Joint Conference on Artificial Intelligence, P. 1104–1111, USA 1995

(Boutilier, 1999) C. Boutilier, T. Dean, and S. Hanks, “Decision theoretic planning: Structural assumptions and computational leverage”, Journal Of Artificial Intelligence Research, Volume 11, pages 1-94, USA, 1999

(Boutte, 2009) J.L. Boutte, "De Taylor au K.M., quelle(s) approche(s) de la compétence ?" in Boutte J.L. & Mallet J. (dir) : La compétence et les nouveaux enjeux de la professionnalisation. "Questions Vives" n°10, Université de Provence – éditeur, France, 2009

(Bowling, 2000) M. Bowling, “Convergence problems of general-sum multi-agent reinforcement learning”, In proceedings of the 7th International Conference on Machine Learning, P 89–94, USA, 2000

(Bowling, 2001) M.H. Bowling, and M. Veloso, “Rational and convergent learning in stochastic games”, In proceedings of 7th International Joint Conference on Artificial Intelligence, P. 1021–1026, USA, 2001

(Bradtke, 1996) S.J. Bradtke, and A.G. Barto, “Linear least-squares algorithms for temporal difference learning”, Journal Machine Learning, Vol. 22, Issue 1-3 , P 33-57, USA 1996

(Brafman, 2001) R. Brafman, and M. Tennenholtz, “R-max - a general polynomial time algorithm for near-optimal reinforcement learning”, In Journal of Machine Learning Research, Vol. 3, P 213-231, USA 2001

(Briot, 2001) J.P. Briot, Y. Demazeau, "Introduction aux agents: Principes et architecture des systèmes multi-agents", Collection IC2, Hermès, France, 2001

(Brown, 1951) G.W. Brown, “Iterative solution of games by fictitious play”, in Activity Analysis of Production and Allocation, T. C. Koopmans (Editor), P 374-376, USA, 1951

(Calut, 2012) L. Calut, “La définition et la sélection des compétences clés“, résumé du projet Definition and Selection of Competencies de l'OCDE, http://www.deseco.admin.ch/, France, 2012

(Cammarata, 1983) S. Cammarata, D. McArthur, and R. Steeb, “Strategies of cooperation in distributed problem solving”, In Proceedings of the 8th International Joint Conference on Artificial Intelligence, Germany, 1983

(Chator, 2013) O. Chator, J.M. Salotti, and P.A. Favier, “Multi-agent System for Skills Sharing in Sustainable Development Projects”, proceedings of COGNITIVE 2013, the 5th International Conference on Advanced Cognitive Technologies and Applications, IARIA Conference, p. 21-26,Valencia, Spain, 2013

(Chomsky, 1956) N. Chomsky,”Three models for the description of language”, IRE Transactions on Information Theory, no 2, p. 113–124, USA, 1956

(Claus, 1998) C. Claus, C. Boutilier, “The dynamics of reinforcement learning in cooperative multi-agent systems”, In: Proceedings of the 15th National Conference on Artificial Intelligence, P 746–752, USA, 1998

(Cooper, 1990) G.F. Cooper, “The computation complexity of probabilistic inference using Bayesian belief networks, Artificial intelligence”, vol. 42, P 393-405, USA, 1990

(Corkill, 1983) D. D. Corkill and V. R. Lesser, “The use of meta-level control for coordination in a distributed problem solving network”, In Proceedings of the 8th International Joint Conference on Artificial Intelligence, P 748-756, Germany, 1983

(Cornuéjols, 2002) A. Cornuéjols, L. Miclet, and Yves Kodratoff, “Apprentissage Artificiel: Concepts et algorithmes”, Eyrolles, France, 2002

(Cortes, 1995) C. Cortes, and V. Vapnik, “Support-vector networks », Machine Learning, vol. 20, Issue 3, pp. 273–297, USA, 1995

(Coulon, 1990) D. Coulon, J.F. Boisvieux, L. Bourrelly, L. Bruneau, E.Chouraqui, J.-M.David, C.R. Lu, M. Pu, M. Savelli, J. Vrain, ”Le raisonnement par analogie en intelligence artificielle : formalisation, applications”, Actes des 3° journées nationales PRC-GDR Intelligence artificielle, Paris, P 45-88, France, 1990

(Dagum, 1993) P. Dagum, M. Luby, “Approximating probabilistic inference in Bayesian belief network is NP-hard”, Artificial Intelligence”, vol. 60, P 141-153, USA, 1993

(DARPA, 1991) DARPA, R. Bareiss, “Case-Based Reasoning”, Proceedings of a Workshop Held at the Madison Hotel, Morgan Kaufmann 1 edition, Washington, D. C., May 8-10, USA, 1991

(De La Higuera, 2002) C. de la Higuera, “Grammatical Inference: Learning Automata and Grammars”, Cambridge University Press, UK, 2010

(Dean, 1989) T. Dean, K. Kanazawa, “A model for reasoning about persistence and causation”, Computational Inteligence, Vol. 5, Issue 2, pages 142–150, USA, 1989

(Deen, 1999) S. M. Deen, “A Computational Model for a Cooperating Agent System”, Cooperative Information Agents III, proceedings of the Third International Workshop, CIA'99, P 185-205, Sweden, 1999

(Degris, 2006) T. Degris, O; Sigaud, and P.H. Wuillemin, “Learning the structure of factored markov decision processes in reinforcement learning problems”, Proceedings of the 23rd International Conference on Machine Learning, p. 257–264, USA, 2006

(Degris, 2012) T. Degris, P.M. Pilarski, and R.S. Sutton, “Apprentissage par Renforcement sans Modèle et avec Action Continue“, Actes des 7èmes Journées Francophones Planification, Décision, et Apprentissage pour la conduite de systèmes, France, 2012

(Dickerson, 1992) M.T. Dickerson, and R.L. Scot, III Drysdale, and J.R. Sack, “Simple algorithms for enumerating interpoint distances and finding k nearest neighbors”, International Journal of Computational Geometry and Applications, vol. 2, Issue 3, pp. 221–239, USA,

(Diettrich, 2000) T.G. Diettrich, “Hierarchical reinforcement learning with the MAXQ value function decomposition”, Journal of Artificial Intelligence Research, Vol. 13, P 227–303, USA, 2000

(DISCAS, 2006) DISCAS, “Qu’est-ce qu’une compétence ”, http://csrdn.qc.ca/discas/reforme/definitionCompetence.html, Québec, 2006

(Domingos, 1996) P. Domingos, “Unifying instance-based and rule-based induction”, Machine Learning, Vol. 24, Issue 2, P 141–168, 1996

(Domingos, 1997) P. Domingos, and M. Pazzani, “On the optimality of the simple Bayesian classifier under zero-one loss”, Machine Learning, Vol. 29, pp. 103–137, USA, 1997

(Dortier, 2006) J.F. Dortier, "Des fourmis à Internet. Le mythe de l’intelligence collective", Sciences Humaines N°169, Le dossier sur « L’intelligence collective », mars 2006, P.34-39, France, 2006

(Elayeb, 2009) B. Elayeb, “SARIPOD: Système multi-Agent de Recherche Intelligente Possibiliste de documents Web“, thèse de doctorat, Université de Toulouse, France, 2009

(Ermine, 2010) J.L. Ermine, “Knowledge Crash and Knowledge Management", International journal of knowledge and systems science (IJKSS), 1, 4, pp. 79-95, 2010

(Ferber, 1999) J. Ferber, “Multi-Agent Systems. An Introduction to Distributed Artificial Intelligence“, Addison Wesley, London, UK,, 1999.

(Finin, 1994) T. Finin, R. Fritzson Don McKay, and R. McEntire, “KQML as an agent communication language; Third International Conference on Information and Knowledge Management (CIKM’94)”, ACM Press, USA, 1994

(FIPA, 2002) FIPA, “Foundation for Intelligent Physical Agents - Abstract Architecture Specification (Standard Version)”, publication des Technical Committee de la FIPA, Geneva, Switzerland, 2002

(François, 2003) O. François, Ph. Leray, “Etude comparative d’algorithmes d’apprentissage de structure dans les réseaux Bayésiens”, F. D. de Saint-Cyr, Ed., RJCIA2003 – 6ième Rencontres Nationales des Jeunes chercheurs en Intelligence Artificielle, Presse Universitaire de Grenoble, P 167-180, France, 2003

(Fuchs, 2006) B. Fuchs, J. Lieber, A. Mille, A. Napoli, “Une première formalisation de la phase d’élaboration du raisonnement à partir de cas”, Actes du 14ième atelier du raisonnement à partir de cas, Besançon, France, 2006

(Fudenberg, 1998) D. Fudenberg, and K. Levine K., “The theory of Learning in Games”, MIT Press, Cambridge, USA, 1998

(Fung, 1989) R. Fung, K.C. Chang, “Weighting and integrating evidence for stochastic simulation in Bayesian Network”, Proc. of UAI, Vol. 5, P 209-219, Holland, 1989

(Fung, 1994) R. Fung, B. Del Favero, “Backward simulation in Bayesian network”, R. Lopez de Mantaras and D. Poole Proc. of UAI, page 227-234, 1994

(Genesereth, 1987) M.R. Genesereth, and N.J. Nilsson, “Logical Foundations of Artificial Intelligence”, Morgan Kaufmann, USA, 1987

(Gerard, 2002) P. Gerard, “Systèmes de classeurs : étude de l'apprentissage latent“, Thèse de doctorat, spécialité informatique, Université Pierre et Marie Curie, Paris 6, France, 2002

(Gibson, 1977) J. Gibson, “The Theory of Affordances”, R. Shaw & J. Bransford (eds.), Perceiving, Acting and Knowing. Hillsdale, USA, 1977

(Gillet, 1991) P. Gillet, “Construire la formation: outils pour les enseignants et les formateurs“, PUF, Paris, France, 1991.

(Gladun, 2007) A. Gladun, J. Rogushina, “An Ontology-based approach to Student Skills in Multi-Agent E-Learning systems”, International Journal “Information Technologies and Knowledge”, Vol.1, USA, 2007

(Gleizes, 2008) M.P. Gleizes, C. Bernon, F. Migeon, and G. Picard, “Méthodes de développement de systèmes multi-agents” – Ecole des Mines de Sait-Etienne – Génie Logiciel N°86 – Vol. 86, p. 2-7, France, 2008

(Gold, 1967) E.M. Gold, “Language identification in the limit”, Information and Control, Vol. 10, Issue 5, pp. 447–474, USA, 1967

(Grabmeier, 2002) J. Grabmeier, and A. Rudolph, “Techniques of cluster algorithms in data mining”, Data Mining and Knowledge Discovery, Volume 6, Issue 4 , pp 303-360, USA, 2002

(Greenwald, 2003) A. Greenwald, K. Hall, “Correlated-q learning”, Proceedings of the 20th International Conference on Machine Learning, P 242–249, 2003

(Grundstein, 2006) M. Grundstein, “Le knowledge management ou comment gérer les connaissances“, Revue Problèmes Economiques, éditeur La Documentation Française, n° 2894, Paris, France, 2006

(Gutknechtl, 1998) O. Gutknecht, J. Ferber, and E. Lieurain, “Des modèles hétérogènes de simulation par systèmes multi-agents”, actes de la conférence SMAGET'98 - Modèles et Systèmes Multi-Agents pour la Gestion de l'Environnement et des Territoires, Cemagref, Clermont-Ferrand, France, 1998.

(Hallouli, 2004) K. Hallouli, “Reconnaissance de caractères par méthodes markoviennes et réseaux Bayésiens”, Thèse de Doctorat spécialité Signal et Images, Ecole Nationale Supérieur des Télécommunications, Télécom Paris, France, 2004

(Hammond, 1986) K.J. Hammond, “CHEF: a model of case-based planning”. Proceedings of the 6th National Conference on Artificial Intelligence, Morgan Kaufman, P 267-271, USA, 1986

(Hansen, 2000) B.K. Hansen, “Weather Prediction Using Case-Based Reasoning and Fuzzy Set Theory”, Master of Computer Science Thesis, Technical University of Nova Scotia, Halifax, Nova Scotia, Canada, 2000

(Haynes, 1988) T. Haynes, K. Lau, S. Sen, “Learning cases to compliment rules for conflict resolution in Multi-Agent systems”, Sandip Sen, editor, AAAI Symposium on Adaptation, Co-evolution and Learning in Multi-agents Systems, pages 51–56, USA, 1988

(Henrion, 1988) M. Henrion, “Propagating uncertainty in Bayesian network by probabilistic logic sampling”, Proc. of UAI, eds. J Lemmer and L. Kanal, Vol; 2, P 149-163, Holland, 1988

(Holger, 1998) F. Holger, O. Rogalla, and R. Dillmann, “Integrating Skills into Multi-Agent Systems”, In Journal of Intelligent Manufacturing, Vol.9, Nr.2, Netherlands, 1998

(Howard, 1981) R. Howard, and J. Matheson, “Influence diagrams”, Howard, R., and J. Matheson (editors), “Readings on the principles and applications of decision analysis”, Strategic Decisions Group, Vol 2, P 721-762, 1981

(Hu, 1998) J. Hu, and M. Wellman, “Multi-agent reinforcement learning: theoretical framework and an algorithm”, in proceedings of the 15th International Conference on Machine Learning, P 242–250, USA 1998

(Jaakkola, 1994) T. Jaakkola, M.L. Jordan, and S.P. Singh, “Convergence of stochastic iterative dynamic programming algorithms”, In Neural Computation,Vol. 6, P Pages 1185-1201, MIT Press, USA, 1994

(Jackson, 1998) P. Jackson, “Introduction To Expert Systems”, Addison-Wesley Publisher, USA, 1998

(Jacod, 2003) J. Jacod, P. Protter, “L'essentiel en théorie des probabilités“, Cassini, Collection Enseignement des mathématiques , France, 2003

(Jaczynski, 1994) M. Jaczynski, B. Trousse, “Fuzzy logic for the retrieval step of a case-based reasoner”, 2nd European Workshop on Case-Based Reasoning, EWCBR 94, P 313-320, France, 1994

(Jennings, 1998) N.R. Jennings, M. Wooldridge, and K. Sycara, “A roadmap of agent research and development”, in Journal of Autonomous Agents and Multi-Agent Systems, Kluwer Academic Publishers, Boston, USA, 1998

(Jensen, 1990) F. Jensen, S. Lauritzen, K; Olesen, “Bayesian updating in recursive graphical models by local computations”, Computational Statistical Quaterly, Vol. 4, P 269–282, USA, 1990

(Jin, 1990) Y. Jin, and T. Koyama, “Multiagent planning through expectation based negotiation”, 10th AAAI International Workshop on Distributed Artificial Intelligence, Bandera, USA, 1990

(Kast, 2002) R. Kast, “La théorie de la décision“, La Découverte, Collection Repères, France, 2002

(Khʼlifa, 1995) T.B. Khʼlifa, J.P. Micallef, A. Quilis, J.F. Brun, et A. Orsetti, “Biomʼcanique de la force de poussʼe en mʼlʼe de rugby”, Sciences & Sport, Vol. 10, pp. 163-164, 1995, France

(Kohonen, 2000) T. Kohonen, S. Kaski, K. Lagus, J. Salojärvi, J. Honkela, V. Paatero, and A. Saarela, “Self organization of a massive document collection”, Neural Networks, IEEE Transactions on, Volume 11 , Issue: 3, P 574-585, USA, 2000

(Kolodner, 1993) J. L. Kolodner, “Case-Based Reasoning”, Morgan Kaufmann, San Mateo, USA, 1993

(Labarre, 2005) M.O. Labarre, “ Apprentissage Multi-Agent”, Département d'informatique et de recherche opérationnelle, Université de Montréal, Montréal, http://www.emse.fr/~boissier/enseignement/sma05/exposes/marcolivier.pdf, Québec, 2005

(Labori, 1985) H. Labori, “Eloge de la fuite”, Folio Eds., France, 1985

(Lance, 1967) G.N. Lance, W.T. Williams, “A general theory of classificatory sorting strategies: I. Hierarchical systems”, Computer Journal”, Vol. 9, P-373-380, UK, 1967

(Lauritzen, 1988) S.L. Lauritzen, D.J. Spiegelhalter, “Local computation with probabilities on graphical structure and their applications to expert system”, Proc. of the royal statistical society, serie B., Vol. 50, P 154-227, USA, 1988

(Lauritzen, 1989) S.L. Lauritzen, N. Wermuth, “Graphical models for associations between variables, some of which are qualitative and some quantitative”, Annals of Statictics, USA, 1989

(Le Blanc, 2007) B. Le Blanc, and J.L. Ermine, “A shannon’s theory of knowledge “, Creating Collaborative Advantage through Knowledge and Innovation, World Scientific Publishing Company Pte Ltd, pp 51-68, USA, 2007

(Le Boterf, 1994) G. Le Boterf, “De la compétence : essai sur un attracteur étrange”, Les Editions d'Organisation, Paris, 1994

(Le Boterf, 2006) G. Le Boterf, “Ingénierie et évaluation des compétences (5ème édition) ”, Eyrolles, Paris, 2006

(Le Boterf, 2008) G. Le Boterf, “Repenser la compétence. Pour dépasser les idées reçues: 15 propositions”, Eyrolles, Paris, 2008

(Ledru, 1991) M. Ledru & S. Michel, “Capital Compétence dans l'Entreprise. Une approche cognitive“, ESF éditeur, Paris, France, 1991

(Littman, 1994) M. Littman, “Markov games as a framework for multi-agent reinforcement learning”, In proceedings of the 11th International Conference on Machine Learning, P 157–163, USA, 1994

(Littman, 2001) M. Littman, R.S. Sutton, and S. Singh, “Predictive Representation of State“,In Advances in Neural Information Processing Systems (NIPS-14), MIT Press, P 1555–1561, USA 2001

(Littman, 2001) Littman M., “Friend-or-foe Q-learning in general-sum games”, In 8th International Conference on Machine Learning, P 157–163, USA, 2001

(Loriette-Rougegrez, 1994) S. Loriette-Rougegrez, “Prédiction de processus à partir de comportements observés : le systèmes REBECAS“, Thèse de Doctorat : Informatique, Paris, 1994

(Loriette-Rougegrez, 1998) S. Loriette-Rougegrez, “Raisonnement à partir de cas pour des évolutions spatiotemporelles de processus”, revue internationale de géomatique, Vol 8, n° 1-2, France, 1998

(Maders, 2000) H.P. Maders, “Conduire une équipe projet”, p. 109, Paris, France, 2000

(Mallet, 2007) J. Mallet, “Intelligence collective, équipe apprenante et TICE. Les problèmes culturels sous-jacents“, actes du Colloque TICE Med, Mai 2007, Marseille, France, 2007

(Mano, 2005) J.P. Mano, M.P. Gleizes and P. Glize, “Résolution émergente et collective de problèmes par systèmes multi-agents: principes et applications”, traitement du signal, vol. 22, n°4, p. 375-391, France, 2005

(Mc Culloch, 1945) W. Mc Culloch, “A heterarchy of values determined by the topology of nervous nets”, p. 89-93, USA, 1945

(Mc Ginty, 2001) L. Mc Ginty, B; Smyth, “Collaborative case-based reasoning: Applications in personalized route planning”, 4th International Conference on Case-Based Reasoning, ICCBR 2001, Vol. 2080, P 362–376, Canada, 2001

(Mc Sherry, 2006) D. Mc Sherry, “Completeness criteria for retrieval in recommender systems”, Advances in Case-Based Reasoning, 8th European Conference (ECCBR’06), P 9–29, Turkey, 2006

(Mezura, 1999) C. Mezrura, M. Occello, Y. Demazeau, and C. Baeijs, “Récursivité dans les systèmes multi-agents : vers un modèle opérationnel”, In JFIADSMA’99, pages 41–52, Hermès, France, 1999

(Michalski, 1973) R. S. Michalski, “AQVAL/1 — computer implementation of a variable-valued logic system VL1 and examples of its application to pattern recognition,” Proceedings of First International Joint Conference on Pattern Recognition, PP. 3–17, USA, 1973

(Min DD, 2006) Ministère en charge du développement durable, “Cadre de référence pour les projets territoriaux de développement durable et Agendas 21 locaux“, France, 2006

(Mitchell, 1997) M. Mitchell, “Machine Learning”, chap. 13 “Reinforcement Learning”, McGraw Hill, p. 367-390, USA, 1997

(Montaignac, 1999) M. Montaignac, “CE, Sect., 15 février 1961, Lagrange”, revue AJDA n°149722, France, 1999

(Mousseron, 1977) J.M. Mousseron, “Savoir-faire”, Rep. Dr. com. Dalloz, Paris, France, 1977

(Mühlenbein, 1992) H. Mühlenbein, "How Genetic Algorithms Really Work: 1. Mutation And Hill Climbing", Manner, R. and Manderick, B. (eds), Proceedings Of The Second Coriference On Parallel Problem Solving From Nature, Elsevier Science, Vol. 2, P 15-25, Belgium, 1992

(Mulder, 2007) M. Mulder, T. Weigel, and K. Collins, “The concept of competence in the development of vocational education and training in selected EU member states: a critical analysis“, Journal of Vocational Education and Training , vol. 59, 1, pp. 67-88, Netherlands, 2007

(Nash, 1951) J. Nash, “Non cooperative games”, Annals of Mathematics, Vol. 54, Isuue 2, P 286–295, USA, 1951

(Nash, 1953) J. Nash, “Two-person cooperative games”, Econometrica, Vol. 21, Issue 1, P 128–140, USA, 1953

(Nash-1, 1950) J. Nash, “The bargaining problem”, Jr. Econometrica, Vol. 18, Issue 2, P 155–162, USA, 1950

(Nash-2, 1950) J. Nash, “Equilibrium points in n-person games”, Proceedings of the National Academy of Sciences, Vol. 36, Issue 1, P 48–49, USA, 1950

(Nonaka, 2000) I. Nonaka, R. Toyama, and N. Konno, “SECI, Ba and leadership: A unified model of dynamic knowledge creation”, Long Range Planning, 33, pp. 5-34, USA, 2000

(Núñez, 1997) J. Núñez Suárez, G. Winstanley, and R. N. Griffiths, “Distributed planning using a framework for Devolution”, Proceedings of the 6th International Conference on "Interfaces-97: Man-machine interaction and intelligent systems in business", France, 1997

(Odell, 2001) J. Odell, H.V. Parunak, and B. Bauer, “Representing agent interaction protocols in UML ; Workshop on Agent-Oriented Software Engineering (AOSE’01)”, P. Ciancarini, M. Wooldridge (Eds.), LNCS, Springer Verlag, Berlin, pp. 121-140, Germany, 2001

(Ontañón, 2005) S. Ontañón, “Ensemble Case Based Learning for Multi-Agents Systems”, PhD, The University of Barcelona, Higher Technical School of Engineering, Bellaterra, Spain, 2005

(PDMIA, 2008) Groupe PDMIA, “Processus décisionnels de Markov en Intelligence Artificielle”, Lavoisier – Hermes Science Publications, France, 2008

(Pearl, 1988) J. Pearl, “Probabilistic reasoning in intelligent systems : Networks of plausible Inference”, Morgan Kaufman Publishers, Inc., USA, 1988

(Pfeifer, 2007) R. Pfeifer, and J.C. Bongard, "How the body shapes the way we think: a new view of intelligence", The MIT Press, USA, 2007

(Plaza, 2002) E. Plaza, S. Ontañon, “Cooperative multiagent learning”, Lecture Notes in Computer Science, Vol. 2636, P 1-17, USA, 2002

(Plaza, 2005) E. Plaza, L. Mcginty, “Distributed case-based reasoning, The Knowledge Engineering”, Review, Vol. 00:0, 1–4.c 2005, Cambridge University Press, UK, 2005

(Prassad, 1996) M.V. Prassad, V.R. Lesser, and S.E. Lander, “Retrieval and reasoning in distributed case bases”, Journal of Visual Communication and Image Representation, Special Issue on Digital Libraries, Vol. 7, Issue 1, P 74–87, USA, 1996

(Prax, 2000) J.Y. Prax, “Le guide du knowledge management“, Dunod, Paris, France, 2000

(Putterman, 1994) M.L. Putterman, “Markov Decision Proceses. Discrete stochastic dynamic programming”, Wiley Series in Probability and Statistics, book 594, USA, 1994

(Quinlan, 1986) J. Quinlan, “Induction of Decision Trees”, Machine Learning, 1, pp.81-106, USA, 1986

(Quinlan, 1993) J.R. Quinlan, “C4.5: Programs for Machine Learning”, Proceedings of European Conference on Machine Learning, P 3-20, Austria, 1993

(Quinlan, 1994) J.R. Quinlan, “Programs for Machine Learning”, Machine Learning, Vol. 16, Issue 3, pp 235-240, The Netherlands, 1994

(Robinson, 1951) J. Robinson, “An iterative method of solving a game”, Annals of Mathematics, Vol. 54, P 296–301, USA, 1951

(Robinson, 1951) J. Robinson J., “An iterative method of solving a game”, The Annals of Mathematics, Series 2, Vol. 54, No. 2, P 296-301, USA, 1951

(Rocher, 2006) G. Rocher, “The Definitive Guide to Grails“, Apress, New York, USA, 2006

(Rodrigues, 2010) C. Rodrigues, P. Gérard, and C. Rouveirol, ”Apprentissage incrémental de règles d'actions relationnelles”, Actes du 17ème congrés pour la Reconnaissance des Formes et Intelligence Artificielle, France, 2010

(Routier, 2002) JC. Routier, P. Mathieu and Y. Secq, “Dynamic Skills Learning : a Support to Agent Evolution”, 5th Pacific Rim International Workshop on Multi-Agents, PRIMA 2002 Tokyo, Proceedings, August 18–19, P109-122, Japan, 2002

(Roux, 2011) D. Roux, “La notion de compétence“, http://formation.ac-bordeaux.fr/pedagogie/ress_pedago/prod_inspection/socle/notion_competence.ppt, Académie de Bordeaux, France, 2011

(Rummery, 1994) G.A. Rummery, and M. Nirajan, "Online Q-Learning using Connectionist Systems”, Technical Report CUED/F-INFENG/TR 166, Cambridge University, UK, 1994

(Russel, 2003) S. Russel and P. Norvig, “Artificial Intelligence: A Modern Approach”, Prentice Hall, USA, 2003

(Rychen, 2001) D.S. Rychen, L.H. Salganik, “Defining and Selecting Key Competencies”, Hogrefe and Huber Publishers, Göttingen, Allemagne, 2001

(Salotti, 1992) S. Salotti, “Filtrage flou et représentation centrée objet pour raisonner par analogie : le système FLORAN“, Thèse de Doctorat : Informatique, Paris, France, 1992

(Salperwyck, 2009) Ch. Salperwyck, and V. Lemaire, “Classification incrémentale supervisée: un panel introductif”, Actes de EGC'09, France, 2009

(Sawid, 1992) A. Dawid, “Applications of a general propagation algorithm for probabilistic expert systems”, Statistics and Computing, Vol. 2, P 25–36, USA, 1992

(Schank, 1982) R. Schank, “Dynamic Memory: A Theory of Learning in Computers and People”, Cambridge: Cambridge University Press, UK, 1982

(Schank, 1990) R. Schank, “Explanations, Machine Learning and Creativity”, Y. Kodratoff & R. Michalski (Eds.), Machine Learning: An Artificial Intelligence Approach IV, Morgan Kaufmann, P 31-48, USA, 1990

(Schlick, 2009) M. Schlick, “ Théorie générale de la connaissance” , traduction par Christian Bonnet, Éditions Gallimard, collection « Bibliothèque de philosophie », France, 2009

(Schmidhuber, 1996) J. Schmidhuber, J. Zhao, and M. Wiering, “Simple Principle of Metalearning”, Technical report, IDSIA-69-96, Switzerland, 1996

(Shachter, 1989) R.D. Shachter, M.A. Peot, “Simulation approaches to general probabilistic inference on belief networks”, Proc. of UAI, Vol. 5, P 311-318, Holland, 1989

(Shank, 1982) R. Shank, “Dynamic Memory”, Cambridge University Press, UK, 1982

(Shapiro, 2010) L. Shapiro, “Embodied Cognition”, in Oxford Handbook of Philosophy and Cognitive Science, E. Margolis, R. Samuels, and S. Stich (eds.), Oxford University Press, UK, 2010

(Shapley, 1953) L. Shapley, “Stochastic games”, Proceedings of National Academy of Science, Vol. 39, P 1095–1100, USA, 1953

(Shiu, 2001) S.C. Shiu, K. Shiu, D.S. Yeung, C.H. Sun, and X. Z. Wang, "Transferring case knowledge to adaptation knowledge: An approach for case-base maintenance”, Computational Intelligence", Vol. 17, Issue 2, P 295–314., 2001

(Simon, 1969) H. Simon, “The Sciences of the Artificial”,MIT Press, USA, 1969

(Singh, 2000) S. Singh, M. Kearns, Y. Mansour, “Nash convergence of gradient dynamics in general sum games”, In Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence, P 541–548, USA, 2000

(Stinson, 2006) B. Stinson, “PostgreSQL Essential reference”, New Riders, USA, 2002

(Sutton, 1998) R.S. Sutton, and A.G. Barto, “Reinforcement Learning: An Introduction“, MIT Press, USA, 1998

(Sutton, 2008) R.S. Sutton, C. Szepesvari, A. Geramifard, and M.H. Bowling, “Dyna-style planning with linear function approximation and prioritized sweeping”, Proceedings of 24th Conference of Uncertainty in artificial intelligence, P 528-536, Finland, 2008

(Tesauro, 2004) G. Tesauro G., “Extending q-learning to general adaptive multi-agent systems”, S. Thrun, L. Saul, et B. Schölkopf, editors, Advances in Neural Information Processing Systems, Vol. 16, MIT Press, USA, 2004

(TLFI, 2013) Trésor de la Langue Française Informatisé (TLFI), Analyse et Traitement Informatique de la langue Française, http://www.atilf.fr, France, 2013

(Tolchinsky, 2006) P. Tolchinsky, S. Modgil, U. Cortes, and M. Sanchez-Marre, “Cbr and argument schemes for collaborative decision making”, Proceedings of the 6th conference on Computational Models of Argument, Vol. 144, P 71–82, Holland, 2006

(Vapnik, 1998) V.N. Vapnik, “Statistical learning theory”, New York: Wiley, USA, 1998

(Vapnik, 1998) V. Vapnik, “Statistical Learning Theory”, Wiley-Blackwell, UK, 1998

(Varela, 1991) F. Varela, E. Thompson, and E. Rosch, “The Embodied Mind: Cognitive Science and Human Experience”, MIT Press, USA, 1991

(Von Neumann, 1944) O. Von Neumann, and O. Morgenstern, “Theory of Games and Economic Behaviour”, Princeton University Press, USA, 1944

(Von Neumann, 1994) O. Von Neumann, O. Morgenstern, “Theory of Games and Economic Behaviour”, Princeton Classic Editions, USA, 1994

(Watkins, 1989) C.J.C.H. Watkins, “Learning from delayed rewards”, PhD thesis, Cambridge University, UK, 1989

(Weiss, 1999) G. Weiss, "Multiagent Systems, a Modern Approach to Distributed Artificial Intelligence", MIT Press, USA, 1999

(Winston, 1975) P. H. Winston, “Learning Structural Descriptions from Examples”, The Psychology of Computer Vision, Pp. 157-209, New-York, USA, 1975

(Wooldridge, 1998) M. Wooldridge, and N. Jenning, “The Pitfalls of Agent-Oriented Development”, Proceedings of the 2nd Conference on Autonomous Agents, UK, 1998

(Yildizoglu, 2003) M. Yildizoglu, “Introduction à la théorie des jeux“, Dunod, France, 2003

(Young, 1998) H.P. Young, “Individual Strategy and Social Structure: An Evolutionary Theory of Instituions”, Princeton University Press, Princeton, New Jersey, USA, 1998

(Yua, 2003) R. Yua, B. Iunga, H. Panetto, “A multi-agents based E-maintenance system with case-based reasoning decision support”, Faculté des Sciences, CRAN-Université de Nancy I, and Faculty of Hydroelectric Power, Huazhong University of Science and Technology, Wuhan, China, in Engineering Applications of Artificial Intelligence Vol.16, Issue 4, P 321–333, USA, 2003

(Zaki, 2002) M.J. Zaki, C.J. Hsiao, “CHARM : An efficient algorithm for closed itemset mining”, Proceedings of the Second SIAM International Conference on Data Mining, P 12-28, USA, 2002

(Zarifian, 2002) P. Zarifian, “La politique de la compétence et l'appel aux connaissances à partir de la stratégie d'entreprise post-fordiste“, 1er Colloque du groupe de travail Gestion des Compétences et des Connaissances en Génie Industriel, Nantes, France, 2002

Résumé de thèse

Le Conseil Général de la Gironde (CG33) est une collectivité territoriale qui met en œuvre les politiques décidées par des élus. L'une de ses missions fondamentales est l'induction de comportements auprès des citoyens, des acteurs économiques et institutionnels. Citons par exemple la concrétisation, par l'intermédiaire de la réalisation de projets, des politiques publiques autour des thématiques du développement Durable (DD). Un constat issu des retours d'expérience s'impose : les projets mis en œuvre au CG33 sont de plus en plus complexes. Ils requièrent la collaboration de nombreux acteurs publics et privés qui se connaissent mal. Les démarches territoriales de DD nécessitent l'articulation de compétences spécifiques et interdépendantes, qui n'existent pas chez un acteur unique. Ces constats sont généralisables, quel que soit le secteur d'activité professionnelle. La connaissance, l'identification et le partage optimal de compétences est au cœur même de la réussite de projets.

La problématique consiste à répondre aux attentes de tout maître d'œuvre qui doit être capable de définir un projet, d'identifier les compétences qui le composent, les acteurs capables d'exercer ces compétences, puis finalement d'évaluer à postériori la réussite globale du projet. Définir la liste des compétences n'est pas toujours aisé puisqu'elles évoluent au fil du temps. Elles peuvent, par exemple, se transformer suite à l'arrivée d'évolutions techniques. Elles peuvent également disparaitre si elles sont remplacées ou bien finalement non utilisées. Un système optimal se doit donc d'être dynamique lors de la constitution des projets au fil du temps, dans le but de coller au plus prêt de la réalité du contexte dans lequel le système est utilisé. Nous voyons bien ici que la Compétence (et non l'acteur) est l'élément fondamental du système.

Une réponse dite « classique » pourrait consister à réaliser simplement une base de données où les compétences et acteurs seraient des objets statiques. Cette solution ne répondant pas à notre souhait de dynamisme. Nous proposons plutôt un système informatique collaboratif, accessible en ligne, dans lequel nous définissions des « agents compétence » (AC), dynamiques et apprenants, qui évoluent au sein d'une architecture de type multi-agents. Chaque AC y est unique, bien qu'il puisse être incarné sur le terrain par plusieurs acteurs humains. Il possède une « vie propre », des moyens d'actions multiples et multi-localisés (réparti sur plusieurs acteurs physiques). Tout AC dispose de mécanismes de perception de son environnement, de communication avec les autres ACs. Il utilise de ressources (telles que les acteurs humains) et vise à réaliser 4 buts principaux :

1. Constituer sa propre définition (liste des compétences élémentaires)
2. Restituer une liste d'acteurs humains capables de le concrétiser sur le terrain
3. Etablir des relations avec d'autres ACs afin d'optimiser la constitution des projets
4. Se porter spontanément candidat à une participation à de nouveaux projets

Nos ACs ont également un « cycle de vie » décliné en 3 « âges » (enfance, adolescence et maturité). Chacun d'eux correspond à un niveau d'autonomie particulier. Notre SMA est de type « évènementiel ». Les agents y évoluent grâce aux interactions avec les

utilisateurs humains du système. Un « modèle comportemental » dédié et évolutif est proposé dans le but d'optimiser leur dynamisme et de stimuler leur apprentissage.

Mots clés : Systèmes multi-agent – Compétence - Coopération – Gouvernance

Thesis summary

In France, the “Conseil Général de la Gironde” is a local authority that promotes various sustainable development policies and practices. One of its missions is to coordinate and stimulate public and private partnerships in the framework of green building projects.

Whatever the sector of professional activities and the thematic addressed, the concrete projects’ feedbacks show all the complexity to succeed in managing efficiently the collaboration of involved actors. Indeed, each of them has only a partial knowledge of the others’ skills. To answer to the problematic, and in order to improve the management of the projects, we propose an online collaborative tool that allows actors to share skills. According to the “Multi-Agent” theory, "skill agents" have been defined. The key idea is to consider that a skill is an agent of the system, and the actors are only its resources.

Skill agents are dynamic and autonomous, have learning abilities, and have their own life cycle. They pursue four main goals: building their own definition, identifying actors who can concretize them across projects, setting up links with other skill agents, and beeing candidates to new projects. Skill agents interact with human actors to stimulate their cooperation.

This work shows that our model is appropriate for complex collaborative projects, giving results in various areas. For example, a case study based on the selection of players to form a rugby team has been given.

Keywords : Multi-agent system – Skills - Cooperation – Governance