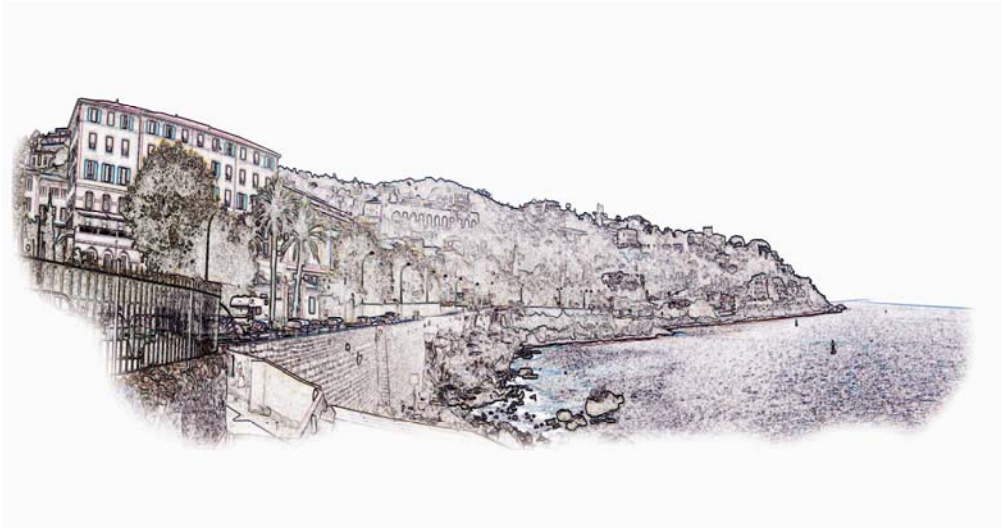
Plate-forme AFIA / Nice, du 30 mai au 3 juin 2005

ATELIER :

Raisonnement à Partir de Cas



Sylvie Després

Bienvenue au 13ème Atelier RÀPC

Le raisonnement à partir de cas (RÀPC) est un paradigme de résolution de problèmes fondé sur la réutilisation d'expériences passées pour résoudre de nouveaux problèmes.

L'atelier RÀPC, organisé chaque année depuis 1993, constitue le lieu de rencontre annuel des chercheurs et des industriels de la communauté française. Ils peuvent y échanger leurs expériences dans le domaine, présenter leurs travaux en cours, proposer de nouveaux projets et débattre sur des thèmes choisis. C'est en particulier un lieu privilégié d'expression pour les doctorants ou les jeunes chercheurs dans le domaine.

Cette année les neuf communications présentées s'inscrivent dans les trois thèmes suivants :

- La conception d'outils d'assistance à la conception de systèmes de RÀPC.
- La mise en œuvre des différentes phases du cycle du RÀPC : élaboration, remémoration, adaptation et réutilisation.
- Le RÀPC et les approches multi-agents.

Avant de souhaiter aux participants, un atelier fertile en idées et en discussions, je tiens à remercier les auteurs pour leur contribution et les membres du comité de programme de l'atelier pour leurs relectures constructives. Je tiens également à remercier le comité d'organisation de la plateforme et tout particulièrement Fabien Gandon.

Sylvie Després

Sommaire

Un assistant pour la conception et le développement des systèmes de RÀPC

Amélie Cordier, Béatrice Fuchs

LIRIS, UMR 5205 CNRS
Université Claude Bernard
43, Boulevard du 11 Novembre 1918
69622 VILLEURBANNE CEDEX
http://liris.cnrs.fr/
{acordier, bfuchs}@liris.univ-lyon1.fr

Résumé : Le raisonnement à partir de cas (RÀPC) est un paradigme de raisonnement complexe ce qui rend délicat le processus développement ou des systèmes qui mettent en œuvre ce type de raisonnement. Dans cet article, nous présentons une ébauche d'architecture pour un environnement d'assistance à la conception et au développement d'applications de RÀPC. Notre approche s'appuie sur les travaux en ingénierie des connaissances avec la prise en compte les deux niveaux: «connaissance» et «symbole». L'outil proposé s'appuie sur les représentations des connaissances par objets dont la richesse des fonctionnalités répond bien aux besoins du RÀPC mis en évidence.

Mots-clés : Raisonnement à Partir de Cas, Représentation des connaissances par objets, Adaptation, Apprentissage.

1 Introduction

Le Raisonnement à Partir de cas (RÀPC) est un paradigme de raisonnement qui s'appuie sur la remémoration de problèmes passés résolus, appelés les cas sources, pour résoudre de nouveaux problèmes, appelés les problèmes cibles (Aamodt & Plaza, 1994). Ce paradigme a été utilisé dans de nombreux systèmes industriels pour résoudre des problèmes dans des domaines d'application variés. Le RÀPC a souvent été présenté comme une solution au goulet d'étranglement de l'étape d'acquisition des connaissances grâce à l'utilisation de connaissances expérimentales, les cas, qui sont plus faciles à recueillir. Bien que certains arguments présentent le RÀPC comme une solution nettement plus facile à mettre en oeuvre que d'autres systèmes à bases de connaissances, force est de constater qu'en pratique, il est impossible de faire l'économie de l'effort d'acquisition de connaissances. Le développement des systèmes de RÀPC se heurte toujours à un problème d'ingénierie des connaissances. En particulier, les connaissances

d'adaptation sont difficiles à modéliser, les fonctionnalités d'apprentissage sont limitées à la mémorisation des cas et l'étape de révision, quant à elle, est trop souvent rudimentaire. Ces systèmes implantent partiellement les fonctionnalités du RÀPC et ont par conséquent un intérêt limité au regard des possibilités théoriques de ce type de raisonnement. Pourtant, des travaux ont été menés afin de modéliser les étapes du raisonnement et aider à leur développement. Mais dans les faits, il est complexe et coûteux de suivre les recommandations issues de ces études, d'autant qu'elles ne sont complètement réifiées dans aucun outil de développement. Même si certains environnements proposent quelques méthodes et outils pour assister la conception d'applications de RÀPC, ceux qui permettent de prendre en compte tous les besoins potentiels des concepteurs d'applications font cruellement défaut. Rares sont les systèmes dotés de possibilités d'extension suffisantes pour personnaliser une application particulière.

Dans cet article, nous nous intéressons à la problématique du développement d'applications de RÀPC ainsi qu'aux pistes de recherche associées. Inspirés par des travaux antérieurs (Fuchs, 1997), nous souhaitons exploiter les avantages des représentations des connaissances par objets pour de telles applications. Nous présentons une idée d'architecture pour la conception d'un environnement d'assistance au développement d'applications de RÀPC. Dans la deuxième partie, nous évoquons les différents travaux menés autour du RÀPC puis nous passons en revue les outils de développement associés que nous considérons en particulier sous l'angle de la représentation des connaissances. Dans la troisième partie, nous proposons l'exploitation de ces résultats au sein d'un outil unique, ouvert et extensible pour l'assistance à la conception d'applications de RÀPC.

2 Raisonnement à partir de cas : modèles et outils

Le raisonnement à partir de cas est un paradigme de résolution de problèmes qui cherche à résoudre un problème cible en s'appuyant sur une base de cas passés résolus. Un cas est constitué d'un problème source et de sa solution source associée. Le RÀPC peut être modélisé par un cycle constitué de cinq étapes - élaboration, remémoration, adaptation, révision et apprentissage - gravitant autour d'une base de connaissances du domaine d'application. Chacune des étapes du cycle mobilise ces connaissances pour supporter la recherche de la solution du problème cible.

Le RÀPC trouve ses origines dans les travaux sur la mémoire dynamique de Schank (Schank, 1982) qui s'intéressaient particulièrement au rôle joué par la remémoration d'expériences passées dans les processus d'apprentissage et de résolution de problèmes. Le RÀPC s'inspire également du raisonnement par analogie dont il est considéré comme un cas particulier. Les travaux successifs sur le RÀPC ont conduit à la réalisation d'outils informatiques de résolution de problèmes s'appuyant entièrement sur ce mode de raisonnement. Si les premières recherches sont issues des travaux en psychologie cognitive, les recherches actuelles sont pluridisciplinaires. Elles ont des ramifications dans les domaines variés de l'intelligence artificielle : représentation des connaissances, classification, mesures de similarité, apprentissage, etc., ce qui en fait un mode de raisonnement complexe. Afin d'appréhender cette complexité, nous allons cerner certains de ces problèmes étape par étape puis plus globalement.

2.1 Les étapes du RÀPC

La première étape du RÀPC est de construire la spécification du problème à résoudre. À partir d'une requête initiale soumise au système, des informations et connaissances sont inférées afin de mieux orienter la recherche pour un objectif et une tâche précis. Élaborer un cas pertinent, c'est-à-dire qui à la fois, reflète le problème courant et soit exprimé en termes cohérents par rapport à la base de connaissances utilisée est une vraie question d'ingénierie des connaissances. Le problème cible élaboré est ensuite utilisé dans l'étape de remémoration pour retrouver un cas «jugé similaire» dans la base de cas. Déterminer la similarité entre deux cas est loin d'être une opération triviale. Elle implique l'utilisation de mesures et de connaissances de similarités fortement liées au domaine d'application. Les travaux qui s'intéressent à la similarité sont nombreux. Des mesures de similarité génériques ont été proposées, et une étude de celles-ci est proposée dans (Rifqi, 1996). Durant l'étape d'adaptation, la solution du cas source est adaptée pour obtenir une solution au problème cible. L'adaptation est un des processus les plus délicats du RÀPC. C'est d'ailleurs pour cette raison qu'il est très souvent négligé ou délégué à l'utilisateur de l'application RÀPC. Dans (Fuchs *et al.*, 2000), les auteurs ont proposé une approche de l'adaptation s'appuyant sur la notion d'influence du problème sur la solution qui, combinée avec les appariements effectués au moment de la remémoration, permet d'adapter la solution du cas cible. Longtemps, les connaissances utilisées pour la remémoration et pour l'adaptation ont été considérées comme distinctes. (Smyth & Keane, 1993) a proposé d'utiliser les connaissances d'adaptation au moment de la remémoration pour privilégier la remémoration des cas minimisant l'effort d'adaptation. Si la remémoration est faite en anticipant correctement sur l'adaptation, cette dernière en sera d'autant plus facilitée et l'on sera sûr d'avoir un cas adaptable. Les connaissances de similarité et d'adaptation apparaissent alors comme étant duales voire même confondues. C'est lors de la phase de révision que la solution proposée peut être corrigée, acceptée ou refusée par l'utilisateur. Cette étape permet d'évaluer l'adaptation. Elle permet également de préparer l'apprentissage puisqu'elle fait émerger de nouvelles connaissances : une nouvelle solution (acceptée ou refusée) mais aussi des connaissances d'adaptation si certaines corrections ont été apportées par l'utilisateur (Aamodt, 1991), ou des connaissances de remémoration si celle si n'est pas jugée satisfaisante (Fox & Leake, 1994). Le résultat de l'évaluation de la solution met en évidence une insuffisance de l'adaptation à produire une solution satisfaisante ou de la remémoration à sélectionner le cas adéquat. L'étape de révision permet également d'évaluer l'utilité du cas nouvellement résolu et d'élaborer une stratégie de rétention ou d'oubli des cas selon leur contribution à la compétence du système (Smyth & Keane, 1995). La phase d'apprentissage soulève également un certain nombre de problématiques de recherche. Les questions qui se posent sont avant tout de savoir quelles sont les connaissances qui doivent être apprises et comment les apprendre. La plupart des recherches portent sur l'apprentissage des cas passés résolus, des méthodes d'indexation et de l'organisation de la base de cas, mais plus rares sont les recherches qui comme (Aamodt, 1991) s'intéressent à l'apprentissage de connaissances implicites telles que les connaissances de similarité ou d'adaptation.

2.2 Les modèles du RÀPC

Plus globalement, certaines études s'intéressent au RÀPC dans son ensemble, par exemple en s'appuyant sur des explications du raisonnement (Massie *et al.*, 2004). Ces explications sont souvent très utiles pour comprendre pourquoi certaines solutions qui paraissent incohérentes ont été proposées et donc pour identifier les failles dans le raisonnement. Elles peuvent également servir de point de départ et être complétées par l'utilisateur pour fournir une solution plus élaborée. On trouve également des travaux visant à proposer un modèle unifié du RÀPC (Fuchs *et al.*, 1999). D'autres travaux se sont intéressés à la représentation des connaissances pour le raisonnement à partir de cas, et en particulier à la représentation des cas ainsi qu'à leur structuration dans une base de cas : (Gomez-Albarran *et al.*, 1999) propose une modélisation du RÀPC en logiques de descriptions, (Napoli, 1992) et (Lieber, 1997) se sont intéressés au lien entre le RÀPC et le raisonnement par classification dans le cadre des RCO, (Aamodt, 1991) a proposé un environnement de représentation des connaissances de type réseau sémantique et des mécanismes de raisonnement associés pour le RÀPC et (Fuchs, 1997), propose de traiter ces aspects comme des problèmes de représentation des connaissances. Un système de RÀPC est un système à base de connaissances et s'intéresser à la conception d'applications de RÀPC nécessite donc de s'intéresser également aux outils de représentation des connaissances.

2.3 Le développement des systèmes de RÀPC

Dans le domaine du développement des systèmes de RÀPC, deux tendances ont été explorées. La première est celle des outils de développement d'applications, comme par exemple Orenge (tec:inno, 2000) ou CBR*TOOLS (Jaczynski & Trousse, 1998), dont l'étude se situe en grande partie au niveau «symbole» et relève donc principalement du génie logiciel. Ces outils implantent une partie des fonctionnalités du RÀPC (Lenz *et al.*, 1998) en proposant en particulier des stratégies de remémoration efficaces, mais ils n'intègrent pas toujours explicitement l'adaptation et l'apprentissage dans le processus de développement, ni un modèle explicite du raisonnement. Le modèle implicite sous-jacent ne permet pas facilement de personnaliser une application nécessitant des fonctionnalités spécifiques. Les mécanismes de ces outils sont la plupart du temps prédéfinis et adaptés à certaines classes d'applications (diagnostic, aide à la décision par exemple).

La seconde catégorie d'environnements de développement est celle des systèmes de représentation de connaissances (SRC) tels que Creekl (Aamodt, 1991) ou Noos (Plaza & Arcos, 1993). Ces systèmes ne sont pas dédiés au RÀPC, mais sont plutôt des systèmes génériques de développement de systèmes à base de connaissances. Ils ne comportent donc pas de modèle pour guider la conception des systèmes de RÀPC et sont trop généraux pour aider efficacement le développement. Dans cette tendance, (Fuchs, 1997) a proposé une modélisation du cycle de raisonnement qui intègre notamment la tâche d'élaboration et explicite les autres étapes et a implanté un outil adéquat pour mettre en œuvre ce modèle.

Le problème de la représentation des connaissances est un problème qui est né avec l'intelligence artificielle et qui se pose toujours. Les premières représentations symbo-

liques étaient les logiques du premier ordre. Bien que très expressives, ces logiques étaient difficiles à manipuler. Rapidement, de nouveaux formalismes plus visuels tels que les réseaux sémantiques et les graphes conceptuels (Chein & Mugnier, 1992) ont émergé. En parallèle, les représentations à base de frames, inspirées des frames de Minsky, ont fait leur apparition. Ces représentations sont particulièrement adaptées à la description des modèles mentaux. Des travaux antérieurs se sont en particulier intéressés à deux de ces formalismes : les *logiques de descriptions* (Napoli, 1997) et les *représentations des connaissances par objets* (RCO) (Euzenat, 1998).

Les logiques de descriptions s'inspirent fortement des logiques du premier ordre mais aussi du concept de *frame*. Elles ont une sémantique forte et présentent l'avantage de produire des représentations dans lesquelles ont peut rapidement déduire de nouvelles connaissances par un processus d'inférence sur la base. En contre-partie, les logiques de descriptions présentent l'inconvénient de ne pas permettre de représenter l'aspect dynamique des éléments modélisés, puisque, comme leur nom l'indique, elles sont essentiellement descriptives.

Les RCO s'inspirent des langages à base de frames et sont à l'origine des langages de programmation par objets. Elles permettent une modélisation structurée des connaissances. Ces représentations s'appuient sur un élément de base, l'objet, qui représente un «concept» du monde réel. Un objet a à la fois des caractéristiques descriptives (propriétés structurelles), souvent appelées attributs et des propriétés comportementales, les méthodes, qui renseignent sur les comportements que peut avoir l'objet. Les objets sont regroupés en classes sur la base de propriétés structurelles et comportementales communes. Les classes sont organisées en hiérarchie les unes par rapport aux autres grâce à la relation d'héritage. Ainsi, contrairement aux logiques de descriptions, les représentations à objets permettent de prendre en compte l'aspect dynamique des éléments modélisés au sein même du système. Elles offrent également des mécanismes d'inférence variés tels que l'héritage, la classification, le filtrage et les facettes procédurales, ce qui permet d'implanter une variété de méthodes de raisonnement ainsi que des fonctionnalités très utiles en pratique (Ducournau *et al.*, 1998). De par leur structure hiérarchique, les représentations à objets permettent de considérer plusieurs niveaux de granularité dans la représentation. L'ensemble de ces propriétés font des RCO des outils permettant une transposition intuitive et aisée des connaissances du monde réel vers des représentations symboliques.

Les RCO et les logiques de descriptions partagent de nombreux principes de représentation et d'inférence. Les classes en RCO peuvent être assimilées aux concepts des logiques de descriptions. L'analogie est la même entre instance et individu. La différence majeure entre les deux formalismes réside dans la notion de rôle. Les rôles dans les logiques de descriptions sont souvent matérialisés par des attributs-liens et ne sont pas hiérarchisés, contrairement aux méthodes des RCO. Certains systèmes de représentations des connaissances par objets, comme AROM (Page *et al.*, 2000) notamment, vont même plus loin en réifiant les associations, ce qui permet de représenter facilement les relations n-aires et les différents rôles qu'elles sous-tendent par exemple.

Certains travaux ont exploité la richesse des représentations des connaissances pour le RÀPC à plusieurs niveaux. Ainsi, (Salotti & Ventos, 1998) propose une modélisation du RÀPC en logiques de descriptions, tandis que (Fuchs, 1997) a développé ROCADE, un

outil de RCO et en a proposé un exemple d'utilisation pour le RÀPC. A un autre niveau, (D'Aquin *et al.*, 2004) s'est intéressé à la notion de points de vues pour la représentation des connaissances d'adaptation, en s'inspirant essentiellement de la façon dont les points de vues sont représentés dans les RCO telles que TROEPS (Marino, 1993).

De nombreuses fonctionnalités des RCO ont motivé notre choix pour ce mode de représentation des connaissances pour les systèmes de RÀPC. En effet, il existe dans les RCO une grande variété de mécanismes de raisonnement utiles pour les besoins du RÀPC. D'autre part, dans un objectif de développement de système à base de connaissances, l'approche centrée objet présente de nombreux avantages tels que la modularité et la réutilisabilité. Finalement la possibilité de disposer de facettes procédurales associées éventuellement à un langage support pour personnaliser une application est indispensable dans cette optique.

3 Un outil support pour la conception d'applications de RÀPC

La conception et le développement de systèmes de RÀPC est, comme pour tout système à base de connaissances, un processus particulièrement complexe. Des outils tels que ReMind (Barletta, 1993) parmi les plus anciens ou plus récemment CBR*Tools (Jaczynski & Trousse, 1998) par exemple ont été développés pour proposer un support aux concepteurs.

L'approche du système CBR*Tools est intéressante et mérite que l'on s'y attarde. L'objectif de CBR*Tools est de faciliter le développement d'applications de RÀPC grâce à un ensemble de composants réutilisables et de méthodes extensibles. Il se présente comme une plate-forme à objets développée en Java et exploitant l'atelier de modélisation Rational Rose. L'approche «repose sur la définition d'une architecture abstraite modélisant les concepts du RÀPC» et «l'architecture intègre des points d'ouverture qui peuvent être configurés par spécialisation ou par intanciation». (Jaczinski, 1998) (p112). Dans cet outil, l'accent est mis sur le développement. Un système de RÀPC est obtenu par réutilisation de classes Java représentant les concepts du RÀPC en utilisant les techniques classiques de la programmation par objets. Il y a donc une bijection entre les concepts et mécanismes du RÀPC et les classes et méthodes de la plate-forme. Les conséquences sont que le niveau connaissance est réduit au minimum et que la connaissance est exprimée essentiellement au niveau «symbole» sous forme de classes Java. CBR*Tools est un outil intéressant pour développer les systèmes RÀPC, ouvert et extensible mais il n'est pas adapté pour la gestion des connaissances du RÀPC. Le logiciel Orenge (tec:inno, 2000) a été développé par la suite dans la même veine que CBR*Tools.

L'approche développée dans le système Rocade, un *framework* développé en 1995 (Fuchs & Mille, 1995), est similaire à CBR*Tools, mais à un niveau d'abstraction différent. Il permet la modélisation au niveau «connaissance» d'un système et des tâches de raisonnement; il constitue donc un environnement de gestion des connaissances à part entière permettant l'explicitation des connaissances. Dans Rocade, le niveau «symbole» est constitué des classes réutilisables du framework comprenant des méthodes d'infé-

rences (héritage, classification, indexation, etc.). Ces méthodes mettent en œuvre le raisonnement à partir de cas et peuvent être spécialisées à deux niveaux : au niveau du framework lui-même et au niveau des facettes des objets et des classes d'objets définis dans le RCO.

Dans la veine du système Rocade, notre approche pour aborder la conception d'un système de RÀPC se rapproche de celle d'un système à base de connaissances. Elle s'apparente à un processus d'ingénierie des connaissances qui vise à expliciter les connaissances et à les modéliser. Cette approche se situe à deux niveaux : le niveau «connaissance» et le niveau «symbole». L'aspect symbolique du système est le «support» sur lequel s'appuie le niveau connaissance. Notre objectif est de proposer un environnement pour le développement d'applications de RÀPC. Cet environnement présente deux facettes distinctes mais néanmoins fortement connectées : un cadre théorique formel d'une part et un ensemble de composants logiciels d'autre part. Le cadre théorique permet de formaliser et de structurer l'ensemble des connaissances dont on dispose sur le RÀPC. Les composants logiciels permettent quant à eux de réifier les concepts énoncés dans le cadre théorique. Ils se présentent sous forme de «briques de base» spécialisables de manière à être réutilisées lors du développement des applications cibles. Il existe un lien fort entre les concepts proposés par le cadre théorique et les composants au niveau symbolique. Grâce à ce lien, un concepteur d'application peut conduire son raisonnement au travers du cadre formel, en restant au niveau connaissance, même si, *in fine*, il manipule des composants logiciels. L'outil idéal fournirait aux concepteurs d'applications de RÀPC un ensemble de composants réutilisables, spécialisables et extensibles pour faciliter et accélérer le développement d'applications spécifiques. L'architecture générale d'un tel système est illustrée par la figure 1.

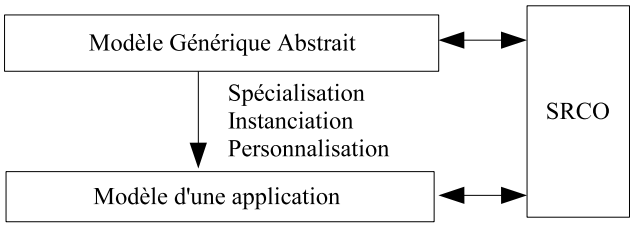


FIG. 1 – *L'architecture générale d'un assistant pour la conception et le développement de systèmes de* RÀPC. *Un modèle générique abstrait est dérivé en un modèle spécifique d'une application par spécialisation. Les deux modèles s'appuient sur une* RCO.

Pour réaliser un tel cadre, nous avons besoin d'un outil de représentation des connaissances. Cet outil servira à la fois à représenter les connaissances sur le raisonnement et les connaissances spécifiques au domaine d'application lors du passage à l'étape de conception. Comme nous l'avons vu, il semble que les représentations des connaissances par objets répondent bien aux besoins exprimés. Dans ce cadre, nous souhaitons présenter une modélisation objet du RÀPC ainsi qu'un modèle de décomposition du raisonnement en sous-tâches de raisonnement, comme cela a été proposé dans (Fuchs, 1997). Selon le principe «diviser pour régner», une telle démarche permet de décompo-

ser chaque tâche en sous-tâches suffisamment simples pour être traitées. La décomposition doit rester générique afin de garder une indépendance franche avec tout domaine d'application. Les sous-tâches élémentaires obtenues pourront être réifiées dans la partie logicielle de l'environnement. A chaque tâche correspond un composant. Les composants ainsi définis pourront alors être réutilisés par les concepteurs d'applications qui auront tout le loisir de les spécialiser pour qu'ils répondent à leurs besoins. Le modèle générique est présenté à la figure 2.

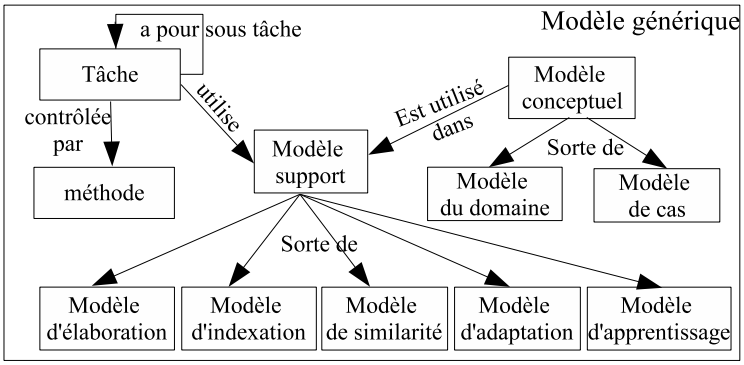


FIG. 2 – *Le modèle générique du* RÀPC.

Le modèle des tâches décrit les différentes tâches de raisonnement et les méthodes associées. Les modèles conceptuels décrivent les connaissances du domaine ainsi que les cas. Le lien entre ces deux types de modèles est effectué par les modèles supports qui mettent en relation les cas et connaissances du domaine avec les tâches de raisonnement précisant leur rôle et leur utilisation dans la résolution de problèmes. Cette approche présente deux intérêts majeurs. Comme nous l'avons déjà dit, elle se situe au niveau connaissance. Toutes les connaissances utiles sur le cycle du RÀPC sont explicitées et sont donc compréhensibles par un spécialiste du domaine. Cet aspect facilite grandement la compréhension du RÀPC par l'ensemble des acteurs du processus de conception des applications. Autre avantage, elle propose un cadre de travail modulaire, évolutif et non limitatif puisqu'elle repose sur le principe de la spécialisation des élements en fonction des besoins. Par exemple, dans la tâche d'élaboration, il serait possible de proposer un outil permettant de guider l'utilisateur sur l'ensemble des connaissances à acquérir pour construire un bon cas. Un tel outil pourrait s'appuyer sur des traces d'utilisation qui joueraient alors un rôle de «facilitateur d'acquisition des connaissances», mettant ainsi l'utilisateur au centre du système. On pourrait également mettre en œuvre un outil centré sur la modélisation de l'adaptation proposé dans (Fuchs *et al.*, 2000). Cet outil proposerait des fonctions d'influence et d'adaptation génériques qui pourraient ensuite être spécialisées en fonction du domaine d'application considéré. Nous pensons que l'étape d'apprentissage est l'étape durant laquelle le plus grand nombre de connaissances est susceptible d'émerger. Or cette étape se résume en règle générale à un simple apprentissage des cas résolus. Si l'on exploitait les connaissances extractibles des interactions entre l'utilisateur et le système lors de la phase de révision, il serait pos-

sible d'apprendre des connaissances d'adaptation qui ne préexistent pas dans la base de connaissance. C'est donc sur ce point que nous allons focaliser notre travail dans un premier temps.

4 Conclusion

Cet article décrit une ébauche d'architecture pour un assistant de conception d'applications de RÀPC. Il présente une justification de l'approche «niveau connaissance» que nous avons suivie. Il montre également l'importance de la représentation des connaissances dans de telles applications et insiste en particulier sur le formalisme sur lequel nous avons choisi de nous appuyer : les représentations des connaissances par objets.

Les perspectives de ce travail sont multiples. Nous projetons d'une part de présenter une formalisation aboutie de cette architecture. D'autre part, nous nous pencherons plus particulièrement sur la question de l'apprentissage des connaissance et en particulier des connaissances d'adaptation, lors des phases de révision et d'apprentissage.

Références

AAMODT A. (1991). *A Knowledge-Intensive, Integrated Approach to Problem Solving and Sustained Learning*. PhD thesis, University of Trondheim, Norwegian Institute of Technology, Department of Computer Science.

AAMODT A. & PLAZA E. (1994). Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AICOM*, **7**, 39–59.

BARLETTA R. (1993). *Remind Developper's Manual*. Rapport interne, Cognitive Systems.

K. BRANTING, K.-D. ALTHOFF & R. BERGMANN, Eds. (1999). *Third International Conference on Case-Based Reasoning - ICCBR'99*, Seeon, Germany. LNAI, Springer, Berlin.

CHEIN M. & MUGNIER M.-L. (1992). Conceptuals graphs: Fundamental notions. *Revue d'Intelligence Artificielle*, **6**(4), 365–406.

D'AQUIN M., LIEBER J. & NAPOLI A. (2004). Représentation de points de vue pour le raisonnement à partir de cas. In *LMO*, p. 245–258.

DUCOURNAU R., EUZENAT J., MASINI G. & NAPOLI A. (1998). *Langages et modèles à objets, état des recherches et perspectives*. INRIA.

EUZENAT J. (1998). *Représentation de connaissance par objets*, In *(Ducournau* et al.*, 1998)*, chapter 10, p. 293–319. INRIA.

FOX S. & LEAKE D. B. (1994). Using introspective reasoning to guide index refinement in case-based reasoning. In *Proceedings of the sixteenth annual Conference of the Cognitive Science Society*.

FUCHS B. (1997). *Représentation des connaissances pour le raisonnement à partir de cas : le système ROCADE*. Thèse d'université, Université Jean Monnet, Saint-Etienne, France.

FUCHS B., LIEBER J., MILLE A. & NAPOLI A. (1999). Towards a Unified Theory of Adaptation in Case-Based Reasoning. In *(Branting* et al.*, 1999)*, p. 104–117.

FUCHS B., LIEBER J., MILLE A. & NAPOLI A. (2000). An Algorithm for Adaptation in Case-Based Reasoning. In *Proceedings of the 14th European Conference on Artificial Intelligence - ECAI 2000*, p. 45–49, Berlin: IOS Press.

FUCHS B. & MILLE A. (1995). Objective c extensions for knowledge representation. In *Object-Oriented Computing in the Natural Sciences, OOCNS-95: Second Multidisciplinary International Workshop of Scientific Computing Environments*, IMAG, Grenoble, France.

GOMEZ-ALBARRAN M., GONZALES-CALERO P., DIAZ-AGUDO B. & FERNANDEZ-CONDE C. (1999). Modelling the Life Cycle Using Description Logics. In *in (Branting* et al., *1999)*, p. 147–161.

JACZINSKI M. (1998). *Modèle et plate forme à objets pour l'indexation des cas par situations comportementales : application à l'assistance à la navigation sur le web.* Thèse d'université, Université de Nice - Sophia Antipolis, Nice, France.

JACZYNSKI M. & TROUSSE B. (1998). An Object-Oriented Framework for the Design and the Implementation of Case-Based Reasoners. In *6Th German Workshop on Case-Based Reasoning*, Berlin, Germany.

LENZ M., BARTSCH-SPÖRL B., BURKHARD H.-D. & WESS S. (1998). *Case-Based Reasoning Technology, from foundations to applications*. LNAI 1400. Berlin: Springer.

LIEBER J. (1997). *Raisonnement à partir de cas et classification hiérarchique. Application à la planification de synthèse en chimie organique*. Thèse d'université, Université Henri Poincaré Nancy 1, Nancy, France.

MARINO O. (1993). *Raisonnement classificatoire dans une representation a objets multi-points de vue*. Thèse d'université, LIFIA / IMAG / INRIA Rhône-Alpes, Université Joseph Fourier - Grenoble I.

MASSIE S., CRAW S. & WIRATUNGA N. (2004). Visualisation of Case-Base Reasoning for Explanation. In *7th European Conference on Case-Based Reasoning*, p. 135–144, Madrid, Spain.

NAPOLI A. (1992). *Représentations à objets et raisonnement par classification en intelligence artificielle*. Thèse d'état, Université de Nancy I, Nancy.

NAPOLI A. (1997). *Une introduction aux logiques de description*. Rapport interne, INRIA.

PAGE M., GENSEL J., CAPPONI C., BRULEY C., GENOUD P. & ZIÉBELIN D. (2000). Représentation de connaissances au moyen de classes et d'associations : le système AROM. In *Langages et Modèles à Objets (LMO'00)*, p. 91–106, Mont Saint-Hilaire (QC, CA).

PLAZA E. & ARCOS J.-L. (1993). *Noos: an integrated framework for problem solving and learning*. Rapport interne, Institut d'investigació en Intelligència Artificial, Barcelona, Spain, Report IIIA-RR-97-02.

RIFQI M. (1996). *Mesures de comparaison, typicalité et classification d'Objets flous : théorie et pratique*. Thèse d'université, Université Pierre et Marie Curie, Paris VI, Paris.

SALOTTI S. & VENTOS V. (1998). Study and Formalization of a Case-Based Reasoning System Using a Description Logic. In *EWCBR98*, p. 286–297.

SCHANK R. (1982). *Dynamic memory; a theory of reminding and learning in computers and people*. Cambridge University Press.

SMYTH B. & KEANE M. T. (1993). Retrieving adaptable cases. the role of adaptation knowledge in case retrieval. In M. M. RICHTER, S. WESS, K.-D. ALTHOFF & F. MAURER, Eds., *First European Workshop on Case-Based Reasoning - EWCBR-93*, p. 209–220, Kaiserslautern, Germany: LNAI, vol. 837, Springer, Berlin.

SMYTH B. & KEANE M. T. (1995). Remembering to forget: A competence-preserving deletion policy for CBR systems. In *14th Joint Conference on Artificial Intelligence, IJCAI-95*, p. 377–382, Montréal, Canada.

TEC:INNO (2000). *The orenge Framework. An Architecture for Intelligent Services in the Electronic business Based on XML and Java*. Rapport interne, tec:inno.

Vers une utilisation du critère pessimiste de Wald pour l'aide à la décision à partir de cas

Mathieu d'Aquin, Jean Lieber, Amedeo Napoli

Équipe Orpailleur, LORIA, (CNRS, INRIA, Universités de Nancy),
BP 239, 54 506 Vandœuvre-lès-Nancy
daquin@loria.fr, lieber@loria.fr, napoli@loria.fr

Résumé : Le projet KASIMIR s'intéresse à la gestion des connaissances décisionnelles en cancérologie. Dans ce projet, la base de connaissances pour le traitement du cancer du sein est utilisé de façon directe pour la majorité des cas et doit être adaptée selon les principes du RÀPC quand un problème de décision cible « non standard » est rencontré. Les problèmes cibles non standards traités dans cet article sont ceux pour lesquels une information utile à la prise de décision est manquante. Dans ce contexte, le critère pessimiste de Wald peut être appliqué. Selon ce critère, les décisions doivent être évaluées sur la base de leurs pires conséquences. Ce critère est appliqué dans la phase de remémoration et s'appuie sur les pires conséquences possibles des décisions susceptibles d'être associées à un problème cible. Cet article est une étude préliminaire en vue de l'utilisation de ce critère pour le système KASIMIR et, plus généralement, pour l'aide à la décision à partir de cas.

Mots-clés : aide à la décision à partir de cas, informations manquantes, critère pessimiste de Wald, cancérologie

1 Introduction

Le raisonnement à partir de cas (RÀPC (Riesbeck & Schank, 1989)) a pour objectif de résoudre un problème *cible* à l'aide d'une base de cas. Une telle base est un ensemble de problèmes déjà résolus associés à leurs solutions. Les étapes du RÀPC considérées dans cet article sont la *remémoration* — sélection dans la base de cas d'un cas *similaire* au problème cible — et l'*adaptation* de la solution du cas remémoré afin de résoudre le problème cible. Suivant les principes de la remémoration guidée par l'adaptabilité (Smyth & Keane, 1996), la sélection effectuée par la remémoration doit s'appuyer sur des connaissances d'adaptation. Idéalement, le cas remémoré doit être celui qui fournira, via l'adaptation, la « meilleure solution au problème cible ». Suivre ce principe présente (au moins) deux avantages qui sont particulièrement intéressants dans les domaines pour lesquels de mauvaises décisions peuvent avoir des impacts désastreux (comme en médecine). Premièrement, la précision de la remémoration est améliorée, si on la compare à une approche « classique » de la remémoration, déconnectée

des connaissances d'adaptation. Deuxièmement, le résultat de la remémoration peut être justifié grâce à certaines connaissances d'adaptation.

Dans cet article, un élément de la remémoration guidée par l'adaptabilité destiné au système KASIMIR est étudié. Le domaine d'application de KASIMIR est l'aide à la décision pour le traitement du cancer du sein. Ce traitement s'appuie sur un *référentiel* (similaire à un guide de bonnes pratiques cliniques ou à un protocole de décision). Le but du RÀPC dans KASIMIR est l'aide à la décision pour les situations non prises en compte par le référentiel.

La section 2 présente des notations concernant le RÀPC. La section 3 décrit le projet KASIMIR en général, en mettant l'accent sur ce qui concerne le RÀPC. La section 4 présente le critère pessimiste de Wald et son utilisation pour une remémoration d'un système d'aide à la décision à partir de cas. Ce critère peut s'appliquer quand des données cruciales concernant le problème cible manquent. Dans une telle situation, les conséquences des différentes décisions doivent être examinées et le critère pessimiste de Wald conduit à décider sur la base des pires conséquences possibles d'une décision. Finalement, après une discussion (section 5), la section 6 conclut l'article.

2 Quelques notations sur le RÀPC

Dans un domaine d'application donné, les notions de *problème* et de *solution* sont supposées être définies. Dans cet article, un problème est une description (éventuellement incomplète) d'une situation pour laquelle une décision est requise et une solution est une décision. Un *cas* est un problème $\texttt{pb}$ associé à une de ses solutions $\texttt{Sol}(\texttt{pb})$; il est dénoté par le couple $(\texttt{pb}, \texttt{Sol}(\texttt{pb}))$. La *base de cas* est l'ensemble des cas disponibles, aussi appelés *cas sources*, dénotés par $(\texttt{srce}, \texttt{Sol}(\texttt{srce}))$. Un *problème source* est un problème dont une solution $\texttt{Sol}(\texttt{srce})$ est connue, autrement dit, tel que $(\texttt{srce}, \texttt{Sol}(\texttt{srce}))$ est un cas source. Le problème à résoudre est dénoté par $\texttt{cible}$ et est appelé *problème cible*.

3 Contexte applicatif : le projet KASIMIR

Le cadre du projet KASIMIR est la gestion des connaissances décisionnelles en cancérologie en Lorraine (Lieber *et al.*, 2002; d'Aquin *et al.*, 2004). Dans cette région, les décisions en cancérologie sont prises à l'aide de *référentiels* (similaires à des guides de bonnes pratiques cliniques). En particulier, le traitement du cancer du sein s'appuie sur le « référentiel sein ». Ce référentiel peut être considéré comme un ensemble de règles : étant donné une règle $\texttt{R} = (\texttt{Prém} \longrightarrow \texttt{Ccl})$, $\texttt{Prém}$ décrit une classe de patients (par des conditions sur l'âge, la taille de la tumeur, etc.) et $\texttt{Ccl}$ décrit une décision thérapeutique associée aux patients de cette classe. Malheureusement, dans 30 à 40% des cas médicaux, une application directe de ces règles n'est pas pertinente, par exemple à cause d'une contre-indication. Un tel cas est alors soumis au *comité de concertation pluridisciplinaire sein* (CCP sein), un comité d'experts des différentes disciplines médicales impliquées dans le traitement du cancer du sein (chimiothérapie, chirurgie, radiothérapie etc.). Le CCP sein *adapte* le référentiel sein pour ce cas médical. Comme

l'a montré (Sauvagnac, 2000), de telles adaptations peuvent entraîner des évolutions du référentiel lui-même.

Le système KASIMIR doit assister le travail du CCP en proposant des adaptations suivant les principes du RÀPC. Les règles $\mathtt{R} = (\mathtt{Prém} \longrightarrow \mathtt{Ccl})$ sont considérées alors comme des cas sources $(\mathtt{srce}, \mathtt{Sol}(\mathtt{srce}))$: $\mathtt{srce} = \mathtt{Prém}$, $\mathtt{Sol}(\mathtt{srce}) = \mathtt{Ccl}$. Par conséquent, ces cas sources sont des *cas fossilisés* (*ossified cases* (Riesbeck & Schank, 1989)). Une discussion sur cette manière inhabituelle d'appliquer le RÀPC est présentée dans (Lieber & Bresson, 2000). Une étude d'acquisition des connaissances d'adaptation auprès d'experts a été menée (Lieber *et al.*, 2001; Lieber *et al.*, 2003). Elle était fondée sur la description des décisions du CCP et sur les explications et justifications de ces décisions par des experts en cancérologie. Plusieurs schémas d'adaptation généraux ont été ainsi mis en évidence. Un d'entre eux, souvent rencontré dans des cas réels de prise de décision traités par le CCP, concerne les cas médicaux pour lesquels certaines informations utiles à la prise de décision manquent. Ce schéma est en fait plus lié à la remémoration (guidée par l'adaptabilité) qu'à l'adaptation. Il est étudié à la section 4.

Actuellement, deux versions de KASIMIR existent : l'une s'appuie sur un formalisme *ad hoc* de représentation des connaissances par objets (Napoli *et al.*, 1994), l'autre, sur OWL (une logique de descriptions (Baader *et al.*, 2003), recommandation du W3C (Bechhofer *et al.*, 2005)). Nous considérerons dans la suite que les problèmes sont représentés par des concepts et sont comparés à l'aide de la relation de subsomption : $\mathtt{pb}_1 \sqsupseteq \mathtt{pb}_2$ signifie que $\mathtt{pb}_1^{\mathcal{I}} \supseteq \mathtt{pb}_2^{\mathcal{I}}$ pour toute interprétation $\mathcal{I}$ satisfaisant les connaissances du domaine représentées. Intuitivement, $\mathtt{pb}^{\mathcal{I}}$ est l'ensemble des instances du problème représenté par $\mathtt{pb}$. L'application directe du référentiel dans KASIMIR s'appuie sur la règle d'inférence suivante (similaire au *modus ponens*) : si $\mathtt{pb} \sqsupseteq \mathtt{pb}'$ alors toute solution de $\mathtt{pb}$ est une solution de $\mathtt{pb}'$. On dira que le problème $\mathtt{pb}$ *couvre* le problème $\mathtt{pb}'$. Si $\mathtt{pb}_1 \sqsupseteq \mathtt{pb}_2$ et $\mathtt{pb}_2 \sqsupseteq \mathtt{pb}_1$, alors $\mathtt{pb}_1$ et $\mathtt{pb}_2$ sont équivalents (ils représentent le même problème, i.e., le même ensemble d'instances de problèmes), ce qu'on dénote par $\mathtt{pb}_1 \equiv \mathtt{pb}_2$. Si $\mathtt{pb}_1$ et $\mathtt{pb}_2$ sont deux concepts représentant des problèmes, alors $\mathtt{pb}_1 \sqcup \mathtt{pb}_2$ est un concept représentant un problème tel que $(\mathtt{pb}_1 \sqcup \mathtt{pb}_2)^{\mathcal{I}} = \mathtt{pb}_1^{\mathcal{I}} \cup \mathtt{pb}_2^{\mathcal{I}}$, pour toute interprétation $\mathcal{I}$.

4 Utilisation du critère pessimiste de Wald pour la remémoration

Cette section présente le critère pessimiste de Wald (abrégé dans la suite en CPW) à l'aide d'un exemple (§4.1) et son application pour la remémoration (§4.2).

4.1 Le critère pessimiste de Wald

Il arrive qu'un problème de décision incomplètement spécifié doive être résolu. Cette section présente le CPW pour une telle prise de décision. Bien que ce critère soit indépendant du RÀPC, nous l'expliquerons en nous appuyant sur des notions et notations du RÀPC.

Considérons le domaine de la comestibilité d'un champignon. Soit $\texttt{cible}$ le problème suivant :

$$\texttt{cible} = \text{« chapeau rouge, volve, lames blanches ou jaunes »}$$

Autrement dit, étant donné un champignon avec de telles caractéristiques, est-ce que je peux le manger ? Afin de résoudre ce problème de décision, il est supposé qu'une base de cas est disponible et qu'elle contient les deux cas $(\texttt{srce}_1, \texttt{Sol}(\texttt{srce}_1))$ et $(\texttt{srce}_2, \texttt{Sol}(\texttt{srce}_2))$, tels que

$$\texttt{srce}_1 = \text{« chapeau rouge, volve, lames jaunes »}$$
$$\texttt{srce}_2 = \text{« chapeau rouge, volve, lames blanches »}$$

La couleur des lames de $\texttt{cible}$ est l'information manquante qui empêche de prendre une décision : si cette couleur était jaune, la décision serait $\texttt{Sol}(\texttt{srce}_1)$, si elle était blanche, la décision serait $\texttt{Sol}(\texttt{srce}_2)$. Un simple examen de la similarité à $\texttt{cible}$ est insuffisant pour différencier ces deux problèmes sources.

À ce stade, le CPW (Wald, 1950; Dubois *et al.*, 2001) peut être appliqué. Ce critère donne une préférence à une décision $\texttt{déc}_1$ par rapport à une décision $\texttt{déc}_2$ si la pire conséquence de $\texttt{déc}_1$ est préférée à la pire conséquence de $\texttt{déc}_2$.

Les décisions associées aux problèmes sources ci-dessus sont :

$$\texttt{Sol}(\texttt{srce}_1) = \text{« Je le mange (il est comestible). »}$$
$$\texttt{Sol}(\texttt{srce}_2) = \text{« Je ne le mange pas (il est vénéneux). »}$$

($\texttt{srce}_1$ représente une oronge vraie et $\texttt{srce}_2$, une amanite tue-mouches).

Il y a deux décisions — $\texttt{déc}_1$ et $\texttt{déc}_2$ — et deux possibilités pour la couleur du champignon $\texttt{cible}$ — (a) et (b). Par conséquent, il y a quatre conséquences possibles pour les décisions — $\texttt{csq}_{1a}$, $\texttt{csq}_{1b}$, $\texttt{csq}_{2a}$ et $\texttt{csq}_{2b}$ — résumées ci-dessous :

	$\texttt{déc}_1 = \texttt{Sol}(\texttt{srce}_1)$ Je mange le champignon	$\texttt{déc}_2 = \texttt{Sol}(\texttt{srce}_2)$ Je ne mange pas le champignon
(a) L'information manquante est « lames jaunes »	Il est bon.	Je perds un champignon comestible.
(b) L'information manquante est « lames blanches »	Je tombe malade.	Je perds un champignon vénéneux.

Supposons que nous ayons l'ordre de préférence suivant entre ces conséquences :

$$\texttt{csq}_{1a} >^{\mathcal{C}} \texttt{csq}_{2b} >^{\mathcal{C}} \texttt{csq}_{2a} >^{\mathcal{C}} \texttt{csq}_{1b}$$

où $\texttt{csq} >^{\mathcal{C}} \texttt{csq}'$ signifie « La conséquence $\texttt{csq}$ est strictement préférée à la conséquence $\texttt{csq}'$ ». La pire conséquence d'une décision est $\texttt{csq}_{1b}$ et donc, selon le CPW, la décision susceptible d'entraîner cette conséquence doit être évitée. L'autre décision — « Je ne mange pas de champignon » — est choisie. Par conséquent, selon ce critère, le cas $(\texttt{srce}_2, \texttt{Sol}(\texttt{srce}_2))$ doit être préféré au cas $(\texttt{srce}_1, \texttt{Sol}(\texttt{srce}_1))$: la symétrie qui empêchait de faire un choix entre $\texttt{srce}_1$ et $\texttt{srce}_2$ est cassée grâce à la connaissance

des décisions associées, des conséquences de ces décisions et de l'ordre de préférence entre ces conséquences.

Remarque 1 : Supposons que l'on dispose de l'information suivante : « Il est plus probable que les lames de cible soient jaunes plutôt que blanches. » Avec cette information additionnelle, il semble naturel de dire que $\texttt{srce}_1$ est plus similaire à cible que ne l'est $\texttt{srce}_2$, mais, en fait, cette information n'a pas d'impact sur la décision prise selon le CPW. En effet, l'ordre de préférence n'est pas modifié par cette information. Si on considère que la bonne décision est $\texttt{déc}_2$, dans ce cas, cela signifie que la notion de similarité assez intuitive qui s'appuie sur les probabilités n'est pas pertinente dans ce cas. Par ailleurs, il est important de noter que cette similarité intuitive ne tient compte que du problème source $\texttt{srce} \in \{\texttt{srce}_1, \texttt{srce}_2\}$ et du problème cible, alors que le CPW tient compte également de la solution $\texttt{Sol}(\texttt{srce}) \in \{\texttt{déc}_1, \texttt{déc}_2\}$ de srce et des conséquences de ces décisions.

Remarque 2 : Avec l'exemple ci-dessus, le raisonnement n'était pas fondé seulement sur les problèmes sources et leurs solutions, mais également sur leurs contextes d'application, en l'occurrence, un contexte gastronomique. En effet, si le contexte change, la préférence $>^{\mathcal{C}}$ peut changer, ce qui peut aboutir à une autre décision. Supposons, dans un nouveau contexte, que je sois en train de mourir de faim et que je doive absolument manger. Dans ce cas, j'obtiens le tableau suivant des conséquences $\texttt{csq}'_{1a}$, $\texttt{csq}'_{1b}$, $\texttt{csq}'_{2a}$ et $\texttt{csq}'_{2b}$:

	$\texttt{déc}_1 = \texttt{Sol}(\texttt{srce}_1)$ Je mange le champignon	$\texttt{déc}_2 = \texttt{Sol}(\texttt{srce}_2)$ Je ne mange pas le champignon
(a) L'information manquante est « lames jaunes »	Je survis.	Je meurs de faim.
(b) L'information manquante est « lames blanches »	Je tombe malade, mais je survis.	Je meurs de faim.

Avec l'ordre de préférence

$$\texttt{csq}'_{1a} >^{\mathcal{C}} \texttt{csq}'_{1b} >^{\mathcal{C}} \texttt{csq}'_{2a} = \texttt{csq}'_{2b}$$

le CPW conduira à éviter la conséquence « Je meurs de faim. » et donc la décision $\texttt{déc}_2$. Par conséquent, la remémoration s'appuyant sur le CPW préférera $(\texttt{srce}_1, \texttt{Sol}(\texttt{srce}_1))$ à $(\texttt{srce}_2, \texttt{Sol}(\texttt{srce}_2))$.

Le CPW peut être modélisé comme suit. On suppose que, dans le domaine d'application considéré, l'ensemble $\mathcal{C}$ de toutes les conséquences de décisions est fini et qu'un ordre total $\geqslant^{\mathcal{C}}$ sur $\mathcal{C}$ est défini et interprété par « est préféré à »[1]. Étant donné le problème de décision pb et une décision déc, soit $\mathcal{C}(\texttt{pb}, \texttt{déc}) \subseteq \mathcal{C}$ l'ensemble des conséquences possibles de la décision déc, dans le contexte du problème pb. $\mathcal{C}(\texttt{pb}, \texttt{déc})$ peut être interprété comme une disjonction de conséquences. Par exemple, si $\mathcal{C}(\texttt{pb}, \texttt{déc}) =$

1. Ce qui suit peut être étendu à des hypothèses plus générales.

$\{\mathtt{csq}, \mathtt{csq}', \mathtt{csq}''\}$, cela signifie que la décision $\mathtt{déc}$ dans le contexte du problème $\mathtt{pb}$ aura pour conséquences $\mathtt{csq}$, $\mathtt{csq}'$ *ou* $\mathtt{csq}''$. Soit $\mathtt{PC}(\mathtt{pb}, \mathtt{déc}) = \min \mathcal{C}(\mathtt{pb}, \mathtt{déc})$, le minimum étant calculé selon l'ordre $\geqslant^{\mathcal{C}}$ (ce minimum existe parce que $\mathcal{C}$ est fini et que $\geqslant^{\mathcal{C}}$ est total) : $\mathtt{PC}(\mathtt{pb}, \mathtt{déc})$ est la pire conséquence de la décision $\mathtt{déc}$ dans le cadre du problème de décision $\mathtt{pb}$. Le CPW indique qu'une décision $\mathtt{déc}_1$ doit être préférée à une décision $\mathtt{déc}_2$ pour un problème $\mathtt{pb}$ si $\mathtt{PC}(\mathtt{pb}, \mathtt{déc}_1) \geqslant^{\mathcal{C}} \mathtt{PC}(\mathtt{pb}, \mathtt{déc}_2)$.

Remarque 3 : Wald a montré que cette approche peut être vue comme une application de la stratégie minimax en théorie des jeux, en considérant qu'un joueur est la personne en charge de la prise de décision et que l'autre est la nature, considérée de façon pessimiste : la pire conséquence de la meilleure décision suivant le CPW est $\max_{\mathtt{déc}} \min \mathcal{C}(\mathtt{pb}, \mathtt{déc})$. C'est pourquoi Wald qualifie ce critère de *minimax strategy*. L'expression *Wald pessimistic criterion* est utilisée dans (Dubois *et al.*, 2001), article dans lequel un raffinement de ce critère dans le cadre de la théorie des possibilités est proposé : dans cette étude, l'incertitude sur les informations du problème cible est représenté à l'aide d'une distribution de possibilités. Dans (Lieber *et al.*, 2001), nous utilisions le terme « principe de précaution ».

4.2 Application pour la remémoration

4.2.1 Hypothèses

La première hypothèse concerne les préférences entre conséquences de décisions. Pour KASIMIR, les conséquences d'une décision de traitement sont les différents états du patient après application du traitement. Cela concerne son espérance de vie, sa qualité de vie, etc. On suppose que les experts sont capables de comparer ces conséquences : quelles que soient les conséquences de décision $\mathtt{csq}$ et $\mathtt{csq}'$, ils peuvent indiquer si $\mathtt{csq} \geqslant^{\mathcal{C}} \mathtt{csq}'$ ou $\mathtt{csq}' >^{\mathcal{C}} \mathtt{csq}$. Cette première hypothèse sera discutée à la section 5.

La deuxième hypothèse concerne la base de cas et, plus précisément, la décision associée à tout problème source : on suppose que cette décision est la meilleure au sens du CPW. On peut l'exprimer ainsi : pour tout cas source $(\mathtt{srce}, \mathtt{Sol}(\mathtt{srce}))$, la décision $\mathtt{Sol}(\mathtt{srce})$ associée à $\mathtt{srce}$ est choisie en appliquant le CPW. Cela signifie que, pour toute décision $\mathtt{déc}$:

$$\mathtt{PC}(\mathtt{srce}, \mathtt{Sol}(\mathtt{srce})) \geqslant^{\mathcal{C}} \mathtt{PC}(\mathtt{srce}, \mathtt{déc}) \tag{1}$$

Enfin, les hypothèses techniques suivantes concernant la fonction $\mathcal{C}(\cdot, \cdot)$ sont faites, pour tous problèmes $\mathtt{pb}_1$ et $\mathtt{pb}_2$ et pour toute décision $\mathtt{déc}$:

$$\mathtt{pb}_1 \equiv \mathtt{pb}_2 \quad \Rightarrow \quad \mathcal{C}(\mathtt{pb}_1, \mathtt{déc}) = \mathcal{C}(\mathtt{pb}_2, \mathtt{déc}) \tag{2}$$

$$\mathcal{C}(\mathtt{pb}_1 \sqcup \mathtt{pb}_2, \mathtt{déc}) = \mathcal{C}(\mathtt{pb}_1, \mathtt{déc}) \cup \mathcal{C}(\mathtt{pb}_2, \mathtt{déc}) \tag{3}$$

(2) est une application du principe de non pertinence de la syntaxe et (3) exprime la nature disjonctive de la fonction de conséquences. Les propriétés ci-dessous peuvent être déduites de (2) et (3) :

$$\mathtt{pb}_2 \sqsupseteq \mathtt{pb}_1 \quad \Rightarrow \quad \mathtt{PC}(\mathtt{pb}_1, \mathtt{déc}) \geqslant^{\mathcal{C}} \mathtt{PC}(\mathtt{pb}_2, \mathtt{déc}) \tag{4}$$

$$\mathtt{PC}(\mathtt{pb}_1 \sqcup \mathtt{pb}_2, \mathtt{déc}) = \min[\mathtt{PC}(\mathtt{pb}_1, \mathtt{déc}), \mathtt{PC}(\mathtt{pb}_2, \mathtt{déc})] \tag{5}$$

Preuve de (4) : Si $\texttt{pb}_2 \sqsupseteq \texttt{pb}_1$, alors $\texttt{pb}_2 \equiv \texttt{pb}_1 \sqcup \texttt{pb}_2$. (2) entraîne alors $\mathcal{C}(\texttt{pb}_1 \sqcup \texttt{pb}_2, \texttt{déc}) = \mathcal{C}(\texttt{pb}_2, \texttt{déc})$. Donc, d'après (3) : $\mathcal{C}(\texttt{pb}_1, \texttt{déc}) \cup \mathcal{C}(\texttt{pb}_2, \texttt{déc}) = \mathcal{C}(\texttt{pb}_2, \texttt{déc})$. D'où : $\mathcal{C}(\texttt{pb}_1, \texttt{déc}) \subseteq \mathcal{C}(\texttt{pb}_2, \texttt{déc})$. En prenant le minimum selon l'ordre $\geqslant^{\mathcal{C}}$ des deux membres de cette inclusion, on obtient (4).
Preuve de (5) : il suffit d'appliquer (3) et le fait que $\min A \cup B = \min(\min A, \min B)$.

4.2.2 Utilisation du CPW pour la remémoration

Principe

Le CPW est utilisé pour indiquer une préférence entre deux cas sources, $(\texttt{srce}_1, \texttt{Sol}(\texttt{srce}_1))$ et $(\texttt{srce}_2, \texttt{Sol}(\texttt{srce}_2))$, dans le cas où manquent des informations sur `cible` permettant de le résoudre par application directe d'un de ces deux cas : $\texttt{srce}_1 \not\sqsupseteq \texttt{cible}$ et $\texttt{srce}_2 \not\sqsupseteq \texttt{cible}$. Intuitivement, on peut penser à une descente dans un arbre de décisions pour laquelle, à un nœud particulier, on ne peut répondre à la question qui y est attachée, à cause d'informations manquantes concernant le problème cible. En revanche, on sait que l'un des deux cas sources contient l'information manquante. Cette situation peut être modélisée comme suit :

$$\texttt{srce}_1 \sqcup \texttt{srce}_2 \sqsupseteq \texttt{cible} \tag{6}$$

Autrement dit, aucun des deux problèmes sources ne couvre `cible`, mais leur réunion si. Par conséquent, pour toute décision `déc`, la propriété (4) entraîne :

$$\texttt{PC}(\texttt{cible}, \texttt{déc}) \geqslant^{\mathcal{C}} \texttt{PC}(\texttt{srce}_1 \sqcup \texttt{srce}_2, \texttt{déc})$$

Si aucune autre information n'est disponible, on choisira la décision qui maximise $\texttt{PC}(\texttt{srce}_1 \sqcup \texttt{srce}_2, \texttt{déc})$. Choisir entre les deux cas sources revient à choisir la décision $\texttt{déc} \in \{\texttt{Sol}(\texttt{srce}_1), \texttt{Sol}(\texttt{srce}_2)\}$ qui maximise ce terme :

$$\begin{aligned}
&\max_{\texttt{déc} \in \{\texttt{Sol}(\texttt{srce}_1), \texttt{Sol}(\texttt{srce}_2)\}} \texttt{PC}(\texttt{srce}_1 \sqcup \texttt{srce}_2, \texttt{déc}) \\
&= \max_j \min[\texttt{PC}(\texttt{srce}_1, \texttt{Sol}(\texttt{srce}_j)), \texttt{PC}(\texttt{srce}_2, \texttt{Sol}(\texttt{srce}_j))] \qquad \text{cf. (5)} \\
&= \max_j \min_i \texttt{PC}(\texttt{srce}_i, \texttt{Sol}(\texttt{srce}_j)) = \max_j \min_{i \neq j} \texttt{PC}(\texttt{srce}_i, \texttt{Sol}(\texttt{srce}_j))
\end{aligned}$$

La dernière égalité est une conséquence de (1) : parmi les quatre termes $\texttt{PC}(\texttt{srce}_i, \texttt{Sol}(\texttt{srce}_j))$, le minimum est nécessairement atteint parmi ceux d'entre eux pour lesquels $i \neq j$. Il suffit donc de comparer $\texttt{PC}(\texttt{srce}_1, \texttt{Sol}(\texttt{srce}_2))$ et $\texttt{PC}(\texttt{srce}_2, \texttt{Sol}(\texttt{srce}_1))$.

Connaissances nécessaires

Pour appliquer le CPW à la remémoration, il suffit donc, pour tous les couples de cas sources $((\texttt{srce}_i, \texttt{Sol}(\texttt{srce}_i)), (\texttt{srce}_j, \texttt{Sol}(\texttt{srce}_j)))$, avec $i \neq j$, de connaître le résultat t_{ij} du test $\texttt{PC}(\texttt{srce}_i, \texttt{Sol}(\texttt{srce}_j)) \geqslant^{\mathcal{C}} \texttt{PC}(\texttt{srce}_j, \texttt{Sol}(\texttt{srce}_i))$. Comme $\geqslant^{\mathcal{C}}$ est total, t_{ji} est la négation logique de t_{ij} (sauf dans le cas particulier de l'égalité des pires conséquences, auquel cas, le choix effectué par le CPW est arbitraire). Cela fait

néanmoins $\frac{n(n-1)}{2}$ booléens t_{ij} qu'il faut connaître, pour n, le nombre de cas de la base, ce qui est fastidieux, voire pratiquement impossible, pour l'expert (supposé pouvoir faire ces comparaisons : cf. première hypothèse de la section 4.2.1).

À moins de disposer de connaissances générales qui permettraient d'inférer t_{ij}, l'approche que nous proposons (et que nous envisageons pour KASIMIR) consiste à ne consulter l'expert qu'au moment où la question t_{ij} se produit pour la première fois et de mémoriser sa réponse (le système s'enrichirait alors au fur et à mesure de son utilisation). En effet, nous avons l'intuition que le nombre de situations pratiques pour lesquelles on a besoin de connaître t_{ij} est bien plus faible que $\frac{n(n-1)}{2}$. Ainsi, la mobilisation du temps de travail de l'expert sera largement diminuée [2].

Adaptation par copie

La remémoration de KASIMIR est guidée par l'adaptabilité. Dans cette situation, l'adaptation est une simple copie : $\texttt{Sol}(\texttt{cible}) = \texttt{Sol}(\texttt{srce}_i)$, où $(\texttt{srce}_i, \texttt{Sol}(\texttt{srce}_i))$ est le cas remémoré ($i \in \{1, 2\}$).

Généralisation

Le CPW peut servir à choisir parmi plus que deux cas sources de façon similaire : on cherche à maximiser le terme $\max_j \min_{i \neq j} \texttt{PC}(\texttt{srce}_i, \texttt{Sol}(\texttt{srce}_j))$.

L'application à un ensemble de cas sources se justifie quand `cible` est couvert par la disjonction de tous les problèmes sources concernés mais pas par la disjonction d'une partie stricte d'entre eux. Par exemple, avec trois cas sources, $(\texttt{srce}_i, \texttt{Sol}(\texttt{srce}_i))$, $i \in \{1, 2, 3\}$, on doit avoir $\texttt{srce}_1 \sqcup \texttt{srce}_2 \sqcup \texttt{srce}_3 \sqsupseteq \texttt{cible}$, mais $\texttt{srce}_i \sqcup \texttt{srce}_j \not\sqsupseteq \texttt{cible}$, quels que soient $i, j \in \{1, 2, 3\}$.

5 Discussion

Connaissance de $\geqslant^{\mathcal{C}}$ par les experts

La première hypothèse sur laquelle nous nous sommes appuyés (cf. section 4.2.1) est que les experts connaissent l'ordre $\geqslant^{\mathcal{C}}$: ils sont capables, quelles que soient $\texttt{csq}, \texttt{csq}' \in \mathcal{C}$ de dire laquelle de ces deux conséquences est préférable à l'autre. Dans certain cas, cela doit être vrai. On imagine en effet que pour un médecin, le décès d'un patient est pire que tout autre conséquence potentielle d'une décision thérapeutique. En revanche, cela devient beaucoup plus discutable dans d'autres situations. Vaut-il mieux,

2. Une tentative de justification de cette intuition s'appuie sur le fait que `cible` représente un problème réel pour lequel manque une information cruciale : on ne sait si ce problème relève en réalité de $\texttt{C}_i = \texttt{cible} \sqcap \texttt{srce}_i$ ou de $\texttt{C}_j = \texttt{cible} \sqcap \texttt{srce}_j$ (où $\sqcap$ est l'opérateur de conjonction de concepts). Pour qu'une telle situation se produise, il nous semble que $\texttt{C}_i$ et $\texttt{C}_j$ doivent être « voisins » (p. ex., si une confusion entre une oronge vraie et une amanite tue-mouches est vraisemblable, confondre cette dernière avec une girolle l'est beaucoup moins). Si l'on modélise cela dans une interprétation $\mathcal{I}$, par une distance d, cela revient à dire que $d(\texttt{C}_i^{\mathcal{I}}, \texttt{C}_j^{\mathcal{I}})$ est petit (avec $d(A, B) = \min\{d(x, y) \mid x \in A, y \in B\}$). *A fortiori*, $d(\texttt{srce}_i^{\mathcal{I}}, \texttt{srce}_j^{\mathcal{I}})$ doit être petit (car $d(\texttt{srce}_i^{\mathcal{I}}, \texttt{srce}_j^{\mathcal{I}}) \leqslant d(\texttt{C}_i^{\mathcal{I}}, \texttt{C}_j^{\mathcal{I}})$) et donc $\texttt{srce}_i$ et $\texttt{srce}_j$ doivent être proches, au sens de cette distance.

par exemple, perdre temporairement tous ses cheveux ou définitivement la moitié d'entre eux [3] ? Cela dépendra de l'avis du patient, plus que de celui de l'expert.

Notre première proposition pour l'application du CPW pour la remémoration devra donc être affinée pour tenir compte du fait que les experts peuvent n'avoir qu'une connaissance incomplète de l'ordre total $\geqslant^{\mathcal{C}}$.

Intégration au sein de la remémoration guidée par l'adaptabilité

L'approche présentée ci-dessus doit être à terme intégrée avec un module de remémoration guidée par l'adaptabilité utilisant le principe des reformulations et des chemins de similarité (Melis *et al.*, 1998). Sans expliquer ce principe en détail, supposons qu'on ait un problème cible qui ne soit couvert par aucune disjonction de problèmes sources, mais qu'on parvienne à mettre en évidence une modification $\texttt{cible} \mapsto \varphi(\texttt{cible})$ du problème à résoudre telle que :

(i) Les connaissances d'adaptation disponibles permettent de résoudre $\texttt{cible}$ à partir d'une solution de $\varphi(\texttt{cible})$.

(ii) Le CPW peut être utilisé pour résoudre $\varphi(\texttt{cible})$, i.e., il existe deux problèmes sources $\texttt{srce}_1$ et $\texttt{srce}_2$ tels que $\texttt{srce}_1 \not\sqsupseteq \varphi(\texttt{cible})$, $\texttt{srce}_2 \not\sqsupseteq \varphi(\texttt{cible})$ et $\texttt{srce}_1 \sqcup \texttt{srce}_2 \sqsupseteq \varphi(\texttt{cible})$.

Dans ce cas, on peut résoudre $\varphi(\texttt{cible})$ en choisissant entre $\texttt{Sol}(\texttt{srce}_1)$ et $\texttt{Sol}(\texttt{srce}_2)$ en utilisant le CPW. Puis, on résout $\texttt{cible}$ en utilisant la propriété (i) ci-dessus.

Extension de la portée du CPW *pour la remémoration*

Pour le problème $\texttt{cible}$ de la section 4.1, si l'on ne sait rien sur la couleur des lames (pas même qu'elles sont blanches ou jaunes), on voudrait quand même appliquer le CPW. On peut envisager d'appliquer de la même façon les mêmes cas sources (lames blanches et lames jaunes), même si ça ne garantit pas que la pire conséquence soit évitée (à supposer que ne pas manger le champignon $\texttt{cible}$, si celui-ci a des lames vertes, conduise à une conséquence catastrophique). Cela dit, en terme de remémoration, puisqu'on n'a pas $\texttt{srce}_1 \sqcup \texttt{srce}_2 \sqsupseteq \texttt{cible}$, comment sélectionner les cas sources en question ? Une piste pourrait être de redéfinir la notion de couverture. Plus haut, nous avons dit qu'un problème $\texttt{pb}$ couvrait un problème $\texttt{pb}'$ si $\texttt{pb} \sqsupseteq \texttt{pb}'$. On peut étendre cette relation en une relation qui tienne compte des connaissances d'adaptation : $\texttt{pb}$ couvre $\texttt{pb}'$ si les connaissances d'adaptation permettent de résoudre $\texttt{pb}'$ à partir d'une solution $\texttt{Sol}(\texttt{pb})$ de $\texttt{pb}$, ce que nous dénoterons par $\texttt{pb} \simeq \texttt{pb}'$. On peut alors considérer que le CPW modifié s'applique quand $\texttt{srce}_1 \not\simeq \texttt{cible}$, $\texttt{srce}_2 \not\simeq \texttt{cible}$ et $\texttt{srce}_1 \sqcup \texttt{srce}_2 \simeq \texttt{cible}$.

Utilisation du CPW *avec des cas sources concrets*

Quand les cas sources sont des cas concrets (et non des cas fossilisés), deux pistes sont envisagées.

3. Cet exemple ne s'appuie pas sur un exemple médical réel.

Soit on peut associer à un cas source $(\texttt{srce}, \texttt{Sol}(\texttt{srce}))$ un problème $\texttt{gén}$ plus général que $\texttt{srce}$ — $\texttt{gén} \sqsupseteq \texttt{srce}$ — et tel que la solution $\texttt{Sol}(\texttt{srce})$ résout de façon satisfaisante $\texttt{gén} : \texttt{Sol}(\texttt{gén}) = \texttt{Sol}(\texttt{srce})$. Une telle généralisation pourrait être faite par l'expert, en enlevant de $\texttt{srce}$ les caractéristiques spécifiques ne jouant pas de rôle dans la prise de décision $\texttt{srce} \mapsto \texttt{Sol}(\texttt{srce})$. On obtient ainsi un cas général $(\texttt{gén}, \texttt{Sol}(\texttt{gén}))$[4].

Ceci n'est pas toujours possible en pratique. Le CPW peut-il s'appliquer néanmoins ? Une piste dans ce sens pourrait être d'utiliser la redéfinition de la notion de couverture présentée ci-dessus (avec $\simeq$ à la place de $\sqsupseteq$).

Utilisation d'autres critères de décision pour la remémoration

Considérons, dans le cadre applicatif du projet KASIMIR, un problème de décision chirurgicale. Soit $\texttt{srce}_1$ et $\texttt{srce}_2$, deux problèmes sources ne différant que par le fait que $\texttt{srce}_1$ concerne des patients présentant des troubles de la coagulation du sang alors que $\texttt{srce}_2$ concerne des patients ne présentant pas de tels troubles. On suppose que $\texttt{Sol}(\texttt{srce}_1)$ est une décision chirurgicale conduisant à moins de risque hémorragique que $\texttt{Sol}(\texttt{srce}_2)$. Soit $\texttt{cible}$, le problème caractérisé par un patient partageant les caractéristiques communes de $\texttt{srce}_1$ et $\texttt{srce}_2$ mais dont on ignore s'il présente des problèmes de coagulation : $\texttt{srce}_1 \not\sqsupseteq \texttt{cible}$, $\texttt{srce}_2 \not\sqsupseteq \texttt{cible}$ et $\texttt{srce}_1 \sqcup \texttt{srce}_2 \sqsupseteq \texttt{cible}$. Or, la pire conséquence d'une chirurgie est toujours le décès du patient, par conséquent $\texttt{PC}(\texttt{srce}_1, \texttt{Sol}(\texttt{srce}_2)) = \texttt{PC}(\texttt{srce}_2, \texttt{Sol}(\texttt{srce}_1))$ et le CPW ne permet pas de décider. Plus généralement, si l'on applique le CPW à la lettre, on ne doit jamais faire de chirurgie ! En revanche, si l'on tient compte dans la décision du fait que cette pire conséquence est bien plus probable dans le cas d'un patient présentant des problèmes de coagulation, on préférera, dans le cas du problème $\texttt{cible}$, appliquer la chirurgie $\texttt{Sol}(\texttt{srce}_1)$ qui prend davantage en compte le risque hémorragique. Par conséquent, le CPW est parfois insuffisant et doit laisser place à des améliorations.

Dans (Dubois *et al.*, 2001), un raffinement du CPW dans le cadre de la théorie des possibilités est proposé : dans cette étude, l'incertitude sur les informations du problème cible est représentée à l'aide d'une distribution de possibilités. Cet article présente par ailleurs d'autres critères de décision, par exemple, des critères optimistes qui peuvent s'appliquer à l'aide à la décision dans les domaines où les conséquences d'un mauvais choix ne sont pas désastreuses.

De façon générale, l'aide à la décision à partir de cas devrait pouvoir tirer profit des travaux sur l'aide à la décision dans l'incertain. De tels travaux permettent de tenir compte, pour une décision, de ses conséquences négatives et de son utilité espérée (en médecine, ce sont les effets indésirables et le bénéfice thérapeutique d'un traitement). Cependant, une raison pour choisir une approche RÀPC pour l'aide à la décision est que, d'une part, trop peu de données pertinentes sont disponibles pour une approche statistique et que, d'autre part, les connaissances générales du domaine sont incomplètes. Dans ces conditions, des critères s'appuyant sur beaucoup de données ou sur des

4. On retrouve le principe d'une telle généralisation dans (Veloso, 1994), avec la notion de *the foot-printed features of the initial state* E_0 d'un plan : l'état initial E_0 est généralisé en ne considérant que les informations de E_0 qui sont nécessaires à l'exécution du plan. Ce principe a été repris dans (Lieber, 1997) pour associer à un cas $(\texttt{srce}, \texttt{Sol}(\texttt{srce}))$ un index $\texttt{idx}(\texttt{srce})$ ($\texttt{gén} = \texttt{idx}(\texttt{srce})$).

connaissances complètes ne sont pas satisfaisants. En revanche, les critères demandant (relativement) peu de données et de connaissances, tel que le CPW, semblent adaptés à l'aide à la décision à partir de cas.

6 Conclusion

Cet article étudie l'application du critère pessimiste de Wald pour la remémoration d'un système d'aide à la décision à partir de cas : étant donné un problème de décision cible pour lequel des données cruciales sont manquantes, le CPW donnera une préférence pour le cas source qui évitera la pire conséquence possible. L'application du CPW s'appuie sur la connaissance de la préférence entre conséquences de la décision associée à un cas source dans le contexte d'un autre cas source, connaissance que le système pourra acquérir au fur et à mesure de son utilisation auprès d'experts.

Ce travail en est à son tout début ; l'application stricte du CPW peut être effectuée quand les cas sources sont suffisamment généraux pour que le problème cible puisse être couvert par la disjonction de deux d'entre eux. Une discussion a été présentée qui envisage comment certaines de ces hypothèses pourraient être relâchées. Cette discussion a aussi mis en évidence le fait que le CPW n'est pas toujours approprié à l'aide à la décision à partir de cas et que d'autres critères liés à l'aide à la décision dans l'incertain peuvent être envisagés dans ce cadre. Les hypothèses sur lesquelles nous nous sommes appuyés sont sans doute un peu forte pour beaucoup de situations pratiques, mais le but de cet article était de proposer un premier modèle.

Remerciements

Les auteurs remercient Henri Prade qui leur a donné des pointeurs bibliographiques sur le critère pessimiste de Wald. Ils remercient également les trois rapporteurs de cet article pour leurs remarques intéressantes et encourageantes. Par ailleurs, ils les prient de les pardonner pour n'avoir pas pu prendre en compte toutes leurs remarques.

Références

F. BAADER, D. CALVANESE, D. MCGUINNESS, D. NARDI & P. PATEL-SCHNEIDER, Eds. (2003). *The Description Logic Handbook.* cambridge, UK: Cambridge University Press.

BECHHOFER S., VAN HARMELEN F., HENDLER J., HORROCKS I., MCGUINNESS D. L., PATEL-SCHNEIDER P. F. & STEIN L. A. (2005). OWL Web Ontology Language Reference, www.w3.org/TR/owl-ref. dernière consultation : janvier.

D'AQUIN M., BRACHAIS S., LIEBER J. & NAPOLI A. (2004). Decision Support and Knowledge Management in Oncology using Hierarchical Classification. In K. KAISER, S. MIKSCH & S. W. TU, Eds., *Proceedings of the Symposium on Computerized Guidelines and Protocols (CGP-2004)*, volume 101 of *Studies in Health Technology and Informatics*, p. 16–30: IOS Press.

DUBOIS D., PRADE H. & SABBADIN R. (2001). Decision-theoretic foundations of qualitative possibility theory. *European Journal of Operational Research*, **128**, 459–478.

LIEBER J. (1997). *Raisonnement à partir de cas et classification hiérarchique. Application à la planification de synthèse en chimie organique.* Thèse d'université, Université Henri Poincaré Nancy 1.

LIEBER J., BEY P., BOISSON F., BRESSON B., FALZON P., LESUR A., NAPOLI A., RIOS M. & SAUVAGNAC C. (2001). Acquisition et modélisation de connaissances d'adaptation, une étude pour le traitement du cancer du sein. In *Actes des journées ingénierie des connaissances (IC-2001)*, p. 409–426, Grenoble.

LIEBER J. & BRESSON B. (2000). Case-Based Reasoning for Breast Cancer Treatment Decision Helping. In E. BLANZIERI & L. PORTINALE, Eds., *Advances in Case-Based Reasoning — Proceedings of the fifth European Workshop on Case-Based Reasoning (EWCBR-2k)*, LNAI 1898, p. 173–185. Springer.

LIEBER J., D'AQUIN M., BEY P., BRESSON B., CROISSANT O., FALZON P., LESUR A., LÉVÊQUE J., MOLLO V., NAPOLI A., RIOS M. & SAUVAGNAC C. (2002). The Kasimir Project: Knowledge Management in Cancerology. In *Proceedings of the 4th International Workshop on Enterprise Networking and Computing in Health Care Industry (HealthComm 2002)*, p. 125–127.

LIEBER J., D'AQUIN M., BEY P., NAPOLI A., RIOS M. & SAUVAGNAC C. (2003). Acquisition of Adaptation Knowledge for Breast Cancer Treatment Decision Support. In M. DOJAT, E. KERAVNOU & P. BARAHONA, Eds., *9th Conference on Artificial Intelligence in Medicine in Europe2003 - AIME 2003, Protaras, Chypre*, Lecture Notes in Artificial Intelligence 2780, p. 304–313.

MELIS E., LIEBER J. & NAPOLI A. (1998). Reformulation in Case-Based Reasoning. In B. SMYTH & P. CUNNINGHAM, Eds., *Fourth European Workshop on Case-Based Reasoning, EWCBR-98*, Lecture Notes in Artificial Intelligence 1488, p. 172–183: Springer.

NAPOLI A., LAURENÇO C. & DUCOURNAU R. (1994). An object-based representation system for organic synthesis planning. *International Journal of Human-Computer Studies*, **41**(1/2), 5–32.

RIESBECK C. K. & SCHANK R. C. (1989). *Inside Case-Based Reasoning.* Hillsdale, New Jersey: Lawrence Erlbaum Associates, Inc.

SAUVAGNAC C. (2000). *La construction de connaissances par l'utilisation et la conception de procédures. Contribution au cadre théorique des activités métafonctionnelles.* Thèse d'Université, Conservatoire National des Arts et Métiers.

SMYTH B. & KEANE M. T. (1996). Using adaptation knowledge to retrieve and adapt design cases. *Knowledge-Based Systems*, **9**(2), 127–135.

VELOSO M. M. (1994). *Planning and Learning by Analogical Reasoning.* LNAI 886. Springer Verlag, Berlin.

WALD A. (1950). *Statistical Decision Functions.* Wiley, New York.

Utilisation de ressources sémantiques pour l'élaboration et la remémoration de cas

Valentina CEAUSU, Sylvie DESPRES

Université René Descartes
CRIP5 – Equipe IAA – Groupe SBC
UFR Mathématiques et Informatique
45 rue des Saints-Pères
75006 PARIS
valentina.ceausu@math-info.univ-paris5.fr
sd@math-info.univ-paris5.fr

Résumé : Dans un système de RàPC exploitant des documents semi structurés, des ressources sémantiques peuvent être intégrées pour faciliter la construction des cas cibles et le processus de remémoration. On s'intéresse dans ce papier à l'utilisation de ressources sémantiques du domaine telles que les ontologies et les terminologies. Dans le système en cours de réalisation, les cas représentent des agents qui évoluent dans un environnement. Une terminologie des accidents de la route permet d'élaborer le cas cible. L'étape de remémoration est orientée par l'alignement de cette ressource terminologique avec l'ontologie de l'accidentologie.

Mots-clés : raisonnement à partir de cas, remémoration, ontologie.

1 Introduction

Le raisonnement à partir de cas permet de résoudre de nouveaux problèmes en réutilisant des expériences passées. Ce papier présente la manière dont des ressources sémantiques telles que les ontologies peuvent être utilisées pour aider l'élaboration des cas d'un système de RàPC et guider la phase de remémoration. La description des cas source est fondée sur un méta modèle représentant des prototypes d'accidents et une ontologie de l'accidentologie. Une terminologie des accidents de la route est utilisée dans l'élaboration des cas cibles. L'étape de remémoration repose sur la mise en correspondance des deux ressources.

2 Présentation générale du système à construire

La finalité du système est l'analyse d'un ensemble d'accidents de la route survenus sur un même secteur routier pour construire des propositions d'aménagement afférentes à ce secteur. Ces propositions définissent des mesures de prévention visant à une réduction du nombre des accidents sur le terrain d'étude. Le paradigme du raisonnement à partir de cas a été choisi pour réaliser cette étude.

2.1 Les ressources du système

La conception du futur système fait intervenir différents types de ressources : les procès verbaux d'accident (PV), et les scénarios types d'accidents. Les PV sont des données brutes tandis que les scénarios types sont des connaissances expertes explicitées par les chercheurs en accidentologie. Des représentations calculables par la machine de ces ressources ont été établies au cours des travaux menés conjointement avec le département MA de l'INRETS.

Un PV (Fig. 1) est un document qui comprend des textes rédigés en langue naturelle décrivant l'accident (synthèse des faits, nature des faits, déclarations des impliqués, etc.) et des rubriques correspondant à des informations codées concernant les lieux, les véhicules et les personnes impliquées.

Synthèse des faits

D'après les déclarations, le véhicule FORD n° XXX conduit par M XXX boulevard de la République venant de la Place Marcel Sembat et **se dirige vers** le Pont d'Issy. Arrivé à la hauteur du passage piéton implanté au 74 du dit boulevard, il **aperçoit un piéton** Me XXX sur ce dit passage, arrêté au milieu de la chaussée. Il **s'arrête** puis **redémarre** et **entre en collision** au niveau de son avant gauche avec Me XXX **sa traversée en courant**. Me XXX **être engagée sur le passage piétons** et **ne pas avoir vu arriver** le véhicule FORD. Le piéton **traverse** des n° pairs vers les n° impairs et de gauche à droite par rapport au sens de progression du véhicule. Suite au choc, le piéton **chute sur** la chaussée. Me XXX blessée, est transportée par les S.P. à l'hôpital. A.PARE. Non admise.

Personnes concernées

Identification conventionnelle	Concernée à titre de	Age	Entendue	Incapacité prévue	Prise de sang	Résultat
A	Conducteur	53	X			
Y01A	*Piéton*	*64*	*X*	*INC*		

Fig. 1 - Extrait d'un PV rendu anonyme par P.A.C.T.O. L .

Les passages en langue naturelle décrivent le déroulement de l'accident selon plusieurs points de vues : force de l'ordre (synthèse des faits), personnes impliquées (déclarations) et témoins (témoignages). L'analyse des accidents de la route est effectuée

à partir de la forme électronique du procès verbal qui a préalablement été rendue anonyme par le logiciel PACTOL (Centre d'Etudes Techniques de l'Equipement (CETE) de Rouen).

Des scénarios types d'accidents ont été élaborés par des chercheurs en sécurité routière. Il s'agit d'outils conçus pour l'analyse des accidents qui constituent des connaissances expertes sur le déroulement et les conséquences des accidents routiers. Un scénario d'accident (Fig. 2) est défini comme un prototype de déroulement correspondant à un groupe d'accidents présentant des similitudes d'ensemble du point de vue de l'enchaînement des faits et des relations de causalité, pour les différentes phases conduisant à la collision.

Situation de conduite

Un véhicule circule en agglomération dans une intersection généralement large (7 cas), **en provenance d'une voie principale** (10 cas dont 5 grandes artères), **et s'apprête à tourner à gauche ou à droite**. **Un piéton, souvent âgé** (plus de 70 ans dans 4 cas**), commence (ou s'apprête) à traverser la branche de destination du véhicule, en général sur un passage piéton** (8 cas ; dans deux autres cas : à proximité immédiate d'un passage).

Situation d'accident

Le conducteur réalise sa manœuvre de changement de direction alors que le piéton poursuit (7 cas) **ou engage** (3 cas) **sa traversée**. **Le plus souvent, le conducteur n'a pas vu le piéton** (6 cas), ou très tardivement (3 cas, 1 cas indéterminé).

Situation d'urgence

Dans la plupart des cas (8 cas), **le conducteur ne réalise pas de manœuvre d'urgence**, où celle-ci ne peut être mise en œuvre avant le choc (autres cas : 1 cas avec freinage, 1 cas indéterminé).

Situation de choc

Le véhicule heurte le piéton.

Propositions d'aménagement associées : **Réduction des vitesses**

Fig. 2 - Scénario type 8

Un scénario est décrit selon les modèles établis pour les accidents par les chercheurs.
Aux « pathologies routières » que constituent les scénarios types peuvent être associées des connaissances sur les moyens de prévention correspondant. Ces connaissances s'appuient sur de revues de la littérature scientifique internationale en sécurité routière. Une première étude a été menée au département *MA* de l'*INRETS* afin d'établir des scénarios types d'accidents impliquant des piétons et les éléments pour leur prévention. Les propositions d'aménagement relatives à ces scénarios sont fournies sous forme de discussion. Les scénarios complétés des propositions d'aménagement qui leur sont associés constitueront à terme la base de cas.

2.2 L'architecture du système

Le système est composé de deux modules (Fig. 3). Le premier module reçoit en entrée un ensemble de PV d'accidents survenus sur le même secteur routier. Sa finalité est

l'affectation d'un scénario d'accident à chaque PV et la construction d'un profil représentatif de l'insécurité routière sur ce terrain. Un profil est composé d'une série de scénarios, chacun affecté d'un poids correspondant aux nombres de PV du terrain d'étude qui lui sont affectés, cette série rendant compte de la majeure partie des cas survenus sur ce secteur.

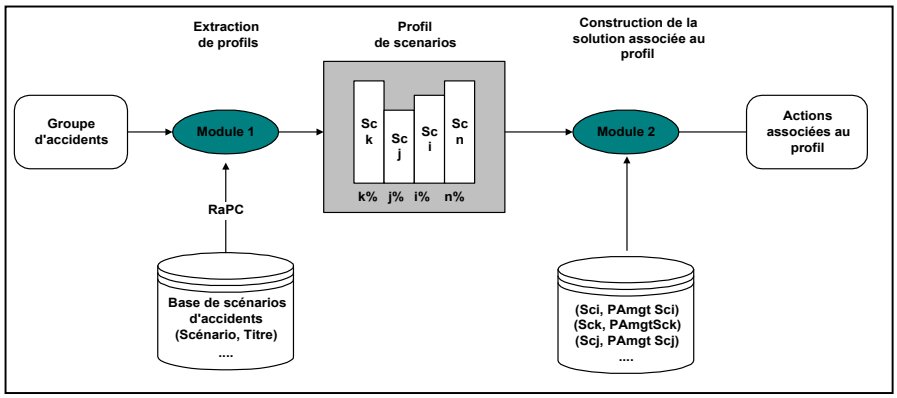


Fig. 3 - Architecture du système

A partir des profils ainsi établis, le second module combine, adapte et pondère les propositions d'aménagement correspondants aux différents scénarios pour pouvoir constituer une stratégie de prévention globale qui soit pertinente compte tenu du « profil » de l'insécurité sur ce secteur. L'élaboration du cas cible et la remémoration sont réalisées par le premier module. La construction des propositions d'aménagements associées au profil est réalisée par le second module.

3 Représentation et élaboration des cas

Un cas cible du système est élaboré à partir d'un PV, en exploitant à la fois les informations codées et les passages en langue naturelle. Une base de cas initiale est construite à partir des scénarios types d'accident. Les modèles élaborés par les chercheurs en accidentologie ont été utilisés pour construire un méta modèle de représentation des accidents et des scénarios d'accidents exploitable par le système. Les ressources ontologique et terminologique respectivement construites à partir des scénarios et de PV contribuent à l'élaboration des cas source et cible.

3.1 Modèles issus de l'accidentologie

Un modèle de l'accident fondé sur le système HVE (Humain, Véhicule, Environnement) et les modèles fonctionnel et séquentiel ont été établis par les chercheurs en accidentologie [Fleury, 1998]. Le système HVE comprend un usager (H), un outil de déplacement (V) et un environnement routier (E) support de ce déplacement. L'environnement routier est formé de l'environnement, de l'infrastructure et du trafic.
Le modèle fonctionnel est un modèle de l'opérateur qui est centré sur la description des mécanismes et qui a pour finalité la compréhension des dysfonctionnements. Il est fondé sur les fonctions psychologiques et les activités psychomotrices mise en jeu dans la conduite automobile. Ce fonctionnement est celui de l'homme vu comme un système de traitement de l'information.
Le modèle séquentiel vise à rendre compte de la complexité et de la dynamique spatio-temporelle particulière des accidents de la circulation et à produire des analyses aptes à faire émerger des perspectives de prévention à des niveaux multiples La structure du modèle est objective, séquentielle et fondée sur des critères de manœuvre, d'espace et de temps. Le modèle séquentiel distingue quatre situations principales articulées autour de ces deux ruptures : la situation de conduite ; la situation d'accident ; la situation d'urgence ; la situation de choc. L'accident est conçu comme le symptôme d'un dysfonctionnement du système HVE, c'est-à-dire une défaillance au niveau des adaptations des composants du système. La compréhension du déroulement d'un accident, représenté par les modèles séquentiel et fonctionnel, repose essentiellement sur l'étude des interactions entre les composants du système HVE.
Un méta modèle (Fig. 4) unique de représentation des accidents et des scénarios a été établi [Després, 2002]. Il s'appuie sur les modèles fonctionnel et séquentiel du domaine et s'inscrit dans l'approche systémique adoptée par les chercheurs de l'INRETS. Il décrit l'état des systèmes HVE dans les différentes phases du modèle séquentiel.
La description de ces différents états est effectuée en utilisant les relations qui lient les trois éléments du système. Deux types de connaissances sont à considérer pour caractériser le modèle du système HVE : les composants du système et les relations, dites intra_HVE, établies entre ces composants. Les composants du système HVE sont des données factuelles statiques sur l'usager impliqué (H), le véhicule (V) et l'environnement (E). Les relations intra_HVE explicitent les interactions entre les trois composants du système : des interactions entre l'homme et l'environnement qui caractérisent la dynamique de la prise d'information ; interactions entre l'homme et le véhicule qui font référence à la dynamique du guidage de l'outil de déplacement ; des interactions entre le véhicule et l'environnement qui correspondent à ce qu'on appelle la dynamique du véhicule. Le modèle fonctionnel intervient dans la description des relations de perception. Dans une même phase du modèle séquentiel, les relations inter_HVE permettent de décrire les interactions entre deux systèmes HVE. L'enchaînement des faits qui conduisent à l'accident est figuré par des relations de transition (inter_phase). Le chemin critique qui correspond à l'explication du déroulement de l'accident est composé de telles

relations. Le méta modèle permet donc de modéliser l'évolution concomitante de l'ensemble des systèmes HVE impliqués dans le scénario d'accident.

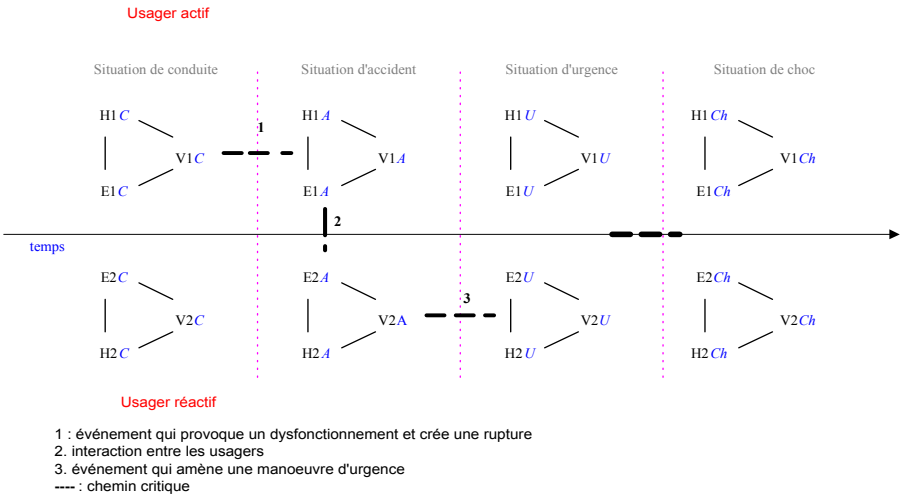


Fig. 4 - Meta modèle de représentation des cas

3.2 Les ressources sémantiques

Représenter un cas consiste à renseigner le méta modèle avec les connaissances contenues dans les scénarios s'il s'agit d'un cas source ou avec les données contenues dans le PV s'il s'agit d'un cas cible. L'intérêt du méta modèle est de permettre la comparaison d'objets qui ne sont donc pas de même nature. Cette situation a conduit à l'élaboration de deux ressources sémantiques construites respectivement à partir des scénarios types et des PV. Les deux ressources correspondent à l'ontologie de l'accidentologie et à la ressource terminologique construite à partir des PV. Une ressource terminologique est une structuration des termes spécifiques à un domaine particulier qui permet de créer une modélisation des connaissances du domaine. Il est nécessaire de travailler avec ces deux ressources car elles représentent des connaissances issues de communautés distinctes dont les objectifs sont respectivement l'analyse pour la prévention de l'accident et la détermination de la responsabilité dans l'accident.
L'ontologie de l'accidentologie [Després, 2002] élaborée à partir de connaissances expertes, de textes du domaine, de scénarios types d'accidents fournit un modèle des concepts du domaine et des relations qu'ils entretiennent. Chaque concept est désigné par un terme du domaine et est caractérisé par des attributs. Plusieurs types de relations sont

établies entre les concepts : des relations hiérarchiques (IS-A) et des relations étiquetées par des verbes qui décrivent les interactions entre concepts.
La ressource terminologique propre aux PV a été établie à partir d'un corpus constitué d'environ 250 procès-verbaux d'accidents de la route survenus dans la région de Lille. Les concepts et les relations du domaine sont structurés de manière identique à l'ontologie de l'accidentologie. Toutefois, pour pouvoir renseigner un cas cible une correspondance doit être effectuée entre les deux ressources. Cette comparaison s'apparente à de l'alignement entre deux ressources ontologiques. L'alignement consiste à rechercher des relations entre des concepts appartenant à des ontologies différentes [Klein, 2001].

3.3 Représentation des cas et élaboration de la base de cas initiale

Un cas est décrit par deux catégories d'éléments : les variables globales et les agents. Les variables globales précisent le nombre d'agents impliqués, l'environnement dans lequel l'accident est survenu (infrastructure large, voie principale) et le contexte de l'accident (en/hors intersection, en/hors agglomération). Un agent (Fig. 5) représente un impliqué et son véhicule. Cette représentation permet de pallier les difficultés liées à la métonymie entre H et V. Un agent est défini par ses composants H et V et par son évolution dans les différentes situations du méta modèle. Chaque composant est désigné par un terme qui relève des concepts H et V et possède des attributs qui décrivent le concept. L'évolution d'un agent est décrite par un ensemble de relations décrivant des interactions entre H et V (intra, inter et trans).

Agent1 H-V	Composant	Terme	Evolution	Liste des attributs
	H	Piéton	SC : s'apprête à traverser SA : poursuit sa traversée	âge, situation de famille, taux d'alcoolémie
	V	x		
Agent2 H-V	H	Conducteur	*SC : circuler sur,* *SA : réalise manœuvre;* *SC : heurter*	âge, situation de famille, taux d'alcoolémie
	V			type, puissance, kilométrage

Fig. 5 - Représentation d'un agent

La base de cas initiale du système est composée de cas sources construits à partir des scénarios d'accidents (Fig. 6). Les cas sources ont la forme (Problème, Solution). Le scénario d'accident constitue la partie « Problème » et la solution d'aménagement qui lui est associée constitue la partie « Solution ».

```
- <Agent>
    <Humain>Conducteur</Humain>
    <Véhicule>VL</Véhicule>
    <Coefficient>5</Coefficient>
  - <Evolution>
    - <Prop>
        <Verbe>circuler_en</Verbe>
        <Coefficient>1</Coefficient>
      </Prop>
    - <Prop>
        <Verbe>ne_pas_détecter</Verbe>
        <Coefficient>2</Coefficient>
      </Prop>
    - <Prop>
        <Verbe>survenir</Verbe>
        <Coefficient>5</Coefficient>
      </Prop>
    </Evolution>
  </Agent>
```

Fig. 6 - Extrait d'un cas source

Un éditeur de scénario fondé sur l'ontologie de l'accidentologie a été créé pour construire les cas sources. L'éditeur permet à l'utilisateur de décrire un scénario en utilisant les concepts et les relations de l'ontologie assurant ainsi l'homogénéité des descriptions. Il permet également de pondérer les composants de chaque agent et les relations qui caractérisent le scénario type décrit.

3.4 Elaboration du cas cible

Un cas cible est élaboré automatiquement à partir d'un PV. Dans un PV, les variables globales et le contexte de l'accident figurent dans des rubriques codifiées. Les procédures automatiques pour retrouver les valeurs des variables globales et le contexte de l'accident reposent sur cette structuration.

La description des agents impliqués dans l'accident nécessite l'identification de plusieurs types: - les termes désignant chaque composant ; - les valeurs des attributs de chaque composant ; - l'évolution de l'agent qui est exprimée par un ensemble de constructions verbales. Les termes désignant les composants de l'agent et les valeurs de leurs attributs sont identifiés automatiquement. Les parties du PV en langue naturelle (la synthèse des faits et les déclarations) sont exploitées pour identifier l'évolution des agents. Elle est exprimée par les verbes figurant dans la description de l'accident.

Des techniques de fouilles de textes [Ceausu & Després, 2005] ont été employées afin d'extraire ces informations. Elles sont appliquées aux textes (la synthèse des faits et les déclarations) qui ont été préalablement annotés par un analyseur syntaxique (Cordial et

TreeTagger) et permettent d'extraire des connaissances en utilisant des patrons linguistiques (cf. Fig.7).

(Verbe, Préposition – **venir, de ; diriger, vers** *)*
(Verbe, Préposition, Adjectif **- circuler, sur, gauche***)*
(Nom, Verbe, Préposition – **véhicule, circuler, sur***)*

Fig. 7 - Patrons et des regroupements associés

Un patron est un regroupement de catégories lexicales.
A l'issue de ce premier traitement, les agents intervenant dans l'accident sont décrits et un ensemble *R* de syntagmes verbaux leur est associé. A ce stade, chaque agent interroge la ressource terminologique pour identifier les relations dont un des arguments est un composant de l'agent.

$$R_{agent}(tHumain, tVéhicule) = R_{ressource}(tHumain) \cup R_{ressource}(tVéhicule) \quad (1)$$

L'intersection de l'ensemble $R_{agent}(tHumain, tVéhicule)$ avec l'ensemble des syntagmes verbaux obtenus identifie l'évolution de l'agent dans le contexte donné.

$$Evolution_{agent} = R \cap R_{agent}(tHumain, tVéhiclue) \quad (2)$$

Le recours à la ressource terminologique construite à partir des PV permet aussi de désigner ces relations avec le vocabulaire du domaine et de s'appuyer sur la classification des relations pour déterminer la phase de l'accident décrite.
L'évolution de l'agent est ainsi identifiée sous la forme d'un ensemble de relations. Les composants des agents du cas cible sont désignés par des termes du domaine spécifiques aux PV, qui se retrouvent dans la ressource terminologique. L'évolution des agents est décrite par des constructions verbales appartenant à la même ressource.

4 Remémoration

L'étape de remémoration permet d'identifier les cas sources les plus similaires au cas cible. L'exploitation des différents niveaux de raisonnement utilisés par les chercheurs pour analyser un accident [Després, 2002] sert de base à la réalisation de cette étape et devrait permettre de réduire la complexité des comparaisons à effectuer.
Pour un cas cible donné, la remémoration est réalisée en deux étapes : la première permet de retrouver un ensemble de cas source similaires au cas cible et la seconde affine cette sélection par un processus de vote. Cette étape de remémoration est en cours de formalisation. Les résultats concernant l'indexation sont opérationnels. Le processus de vote n'est pas encore implanté car les mesures de similarité restent à consolider.

La première étape repose sur l'indexation de la base de cas. Les cas sources sont stockés dans une base de cas indexée dynamiquement selon les valeurs des variables globales intervenant dans un cas et les relations qui caractérisent l'évolution des agents. L'indexation permet de sérier le type d'accidents correspondant au cas cible.
Les valeurs des variables globales du cas cible sont prises en compte pour identifier un ensemble de cas source similaires. L'indexation de la base de cas permet d'optimiser cette première sélection. Une collection de cas source ayant le même contexte que le cas cible et faisant intervenir le même nombre d'agents constitue le résultat de cette première étape.
La seconde étape permet d'affiner cette première sélection via un processus de vote. Le vote est réalisé par chaque agent du cas cible pour exprimer le degré de ressemblance entre l'agent votant et les agents d'un cas source. Le vote se traduit par une note accordée par chaque agent du cas cible à un cas source. Les notes accordées par les agents du cas cible aux cas sources permettront d'estimer la similarité entre les cas.
Le processus de vote permet de choisir, dans l'ensemble des cas sources obtenues dans la première étape, les cas les plus similaires au cas cible. Le vote est réalisé par chaque agent du cas cible afin d'estimer son degré de similarité avec les agents des cas sources. La note accordée par l'agent à un cas source exprime ce degré similarité. Le calcul de cette note prend en compte les composants de l'agent et son évolution.
Une première mesure de similarité est à l'étude (1). Elle exprime le fait qu'en l'absence de ressemblance entre les composants la similarité est nulle. En cas de ressemblance des composants, la similitude entre les relations qui caractérise l'agent est étudiée.

$$\mathrm{Sim}\left(a_i,a_j\right)=\mathrm{SimComposant}\left(a_i,a_j\right)\otimes\mathrm{SimEvolution}\left(a_i,a_j\right) \quad (1)$$

où $\left(a_i\right)$ représente un agent du cas cible et $\left(a_j\right)$ représente un agent du cas source.
La similarité relative aux composants constituant les agents exprime le degré de ressemblance entre les agents relativement à leurs composants. Elle est pour l'instant calculée comme une somme pondérée des ressemblances sur les termes définissant les composants H et V des agents cible et source :

$$\mathrm{SimComposant}\left(a_i,a_j\right)= c_h\mathrm{Sim}\left(\mathrm{humain}_i,\mathrm{humain}_j\right)+c_v\mathrm{Sim}\left(\mathrm{véhicule}_i,\mathrm{véhicule}_j\right) \quad (2)$$

où c_h et c_v caractérisent la représentativité de chaque composant au sein d'un cas source.
L'alignement des ressources sémantiques (ontologie et terminologie issue des PV) permet d'estimer les valeurs de $\mathrm{Sim}\left(\mathrm{humain}_i,\mathrm{humain}_j\right)$ et $\mathrm{Sim}\left(\mathrm{véhicule}_i,\mathrm{véhicule}_j\right)$.
La similarité relative à l'évolution des agents caractérise le degré de ressemblance entre les agents relativement aux relations qui définissent leur évolution.
Un ensemble de relations est associé à chaque agent du cas cible et du cas source. Chaque relation est typée en fonction de la situation où elle intervient dans le modèle séquentiel définit par les chercheurs. Les descriptions des cas source attribuent un coefficient d'importance à chaque relation. Le calcul de la similarité prend en compte la proximité entre les relations et les pondérations qui leur sont associées.

$$SimEvolution(a_i, a_j) = \frac{\sum_r c_r * Sim(relSource_r, relCible_r)}{\sum_r c_r} \quad (3)$$

Les coefficients c_r expriment l'importance de la relation $relCible_r$ pour le cas cible considéré. L'alignement des deux ressources permet l'estimation des valeurs $Sim(relSource_r, relCible_r)$.
Le résultat du vote est exprimé par une note accordée par chaque agent du cas cible à un cas source. Chaque agent du cas cible évalue la similarité avec les agents du cas cible en utilisant (1) à (3) . On obtient un vecteur de valeurs de similarité. La note pour un cas source donné correspond au maximum des valeurs du vecteur de similarité ainsi identifié. Les notes accordées par chaque agent du cas cible à un cas source permettent d'évaluer la similarité entre le cas cible et le cas source sous la forme d'une moyenne :

$$Sim(cible, source) = \frac{\sum_i^{nA} note_i}{nA} \quad (4)$$

Dans (4) « nA » représente le nombre d'agents impliqués dans l'accident et « $note_i$ » la note accordée par l'agent i à un cas source. L'estimation de la similarité en utilisant le vote est implémentée dans l'algorithme:

Pour chaque agent du cas cible :
- évaluer sa similarité avec chaque des agents du cas source selon les relations (1) à (3) ;
- attribuer une note au cas source ;
Calculer la similarité entre le cas cible et le cas source selon (4) ;

La méthode de remémoration proposée a l'avantage de prendre en compte les deux catégories d'éléments décrivant un cas, les variables globales et les agents. Néanmoins, le vote des agents reste déterminant dans la sélection des cas source similaires. Les agents accordent une note fondée sur les ressemblances : ressemblances d'évolution et aussi au niveau des composants.

5 Conclusion et perspectives

Ce travail est en cours de réalisation. A l'heure actuelle, l'alignement des deux ressources est à l'étude et une solution sera proposée prochainement. La prochaine étape concerne l'implémentation effective de la solution dans notre système et les améliorations possibles. Les améliorations peuvent porter sur l'élaboration du cas cible et les mesures de

similarité. Dans l'élaboration du cas cible, les techniques de fouilles de textes pourront être enrichies afin d'avoir une meilleure description du cas.
Dans l'étape de remémoration, les mesures de similarité proposées sont fondées sur les caractéristiques communes des représentations. L'introduction de mesures qui prennent compte les différences entre les représentations est à l'étude.

Références

BERGMANN R. (1998) "On the use of Taxonomies for Representing Case Features" and Local Similarity Measures in Proceedings of the 6th German Workshop on Case-Based Reasoning (GWCBR).

CEAUSU V., DESPRES S. (2005) Fouille de textes pour orienter la construction d'une ressource terminologique, EGC'2005, Paris .

CHERFI H., NAPOLI A., TOUSSAINT Y. (2003) Vers une méthodologie de fouille de textes s'appuyant sur l'extraction de motifs fréquents et de règles d'association, CAP.

DESPRES S. (2002) "Contribution à la conception de méthodes et d'outils pour la gestion des connaissances", Habilitation à Diriger des Recherches en Informatique, Université René Descartes.

FLEURY D. (1998) – Sécurité et urbanisme. La prise en compte de la sécurité routière dans l'aménagement urbain. Presses de l'Ecole Nationale des Ponts et chaussées, 299p.

GUPTA K.M., AHA D.W., SANDHU N. (2002) Exploiting taxonomic and causal relations in conversational case retrieval. *Proceedings of the Sixth European Conference on Case-Based Reasoning* (pp. 133-147). Aberdeen, Scotland: Springer.

HAHN U., SCHNATTINGER K. (1998) Towards text knowledge engineering. Proc. of AAAI'98, pages 129-144.

KLEIN M. (2001) Combining and relating ontologies: an analysis of problems solutions. In A.Gomez-Perez, M. Gruninger, H. Stuckenschmidt, M. Uschold, Eds., *Workshop on ontologies and Information sharing, IJCAI '01*, p. 309-327, Seattle, USA.

KODRATOFF Y. (1999) Knowledge Discovery in Texts: A definition, and Applications. In LNAI. Proc.of the 11th Int'lSymp.ISM'99, volume 1609, p. 16-29, Warsaw:Springer.

SEGUELA P. (1999) " Adaptation semi-automatique d'une base de marqueurs de relations sémantiques sur des corpus spécialisés ", in *Actes de TIA'99 (Terminologie et Intelligence Artificielle),* Nantes, *Terminologies Nouvelles* n°19, pp 52-60.

SMYTH B., McCLAVE P. (2001): Similarity vs. diversity. In Proceedings of the 4th International Conference on CBR, pages 347–361, Vancouver, BC, Canada.

WEBER R., AHA D.W., SANDHU N. & MUNOZ-AVILA H. (2001). A textual case-based reasoning framework for knowledge management applications. In Proceedings of the Ninth German Workshop on Case-Based Reasoning. Baden-Baden, Germany.

WIRATUNGA N., CRAW S., MASSIE S. (2003) Index Driven Selective Sampling for CBR, In proceedings of the 5th International Conference on Case-Based Reasoning, Springer-Verlag.

WIRATUNGA N., KOYCHEV I., MASSIE S. (2004) Feature Selection and Generalisation for Retrieval of Textual Cases. In Proceeding of the 7-th European Conference on Case-Based Reasoning.Berlin, Springer-Verlag.

Un système de raisonnement à partir de cas dans une plateforme de e-maintenance

Ivana Rasovska, Brigitte Chebel-Morello, et Noureddine Zerhouni

Laboratoire d'Automatique de Besançon,
{rasovska,bmorello,zerhouni}@ens2m.fr

Résumé : L'article présente une modélisation des connaissances pour un système d'aide au diagnostic et à la réparation. Ce système emploie la méthodologie du raisonnement à partir de cas et s'intègre dans une plate-forme de e-maintenance. Nous nous intéressons particulièrement à l'étape d'élaboration et d'acquisition de cas ainsi qu'au choix des mesures de similarité et d'algorithme de recherche des cas similaires. La base de cas contient une vingtaine de cas et les premiers tests de recherche des cas similaires sont discutés dans la conclusion.

Mots-clés : diagnostic technique, maintenance industrielle, système de raisonnement à partir de cas.

1 Introduction

1.1 Contexte de travail : Une plate-forme de e-maintenance

Le cadre de notre travail s'inscrit dans un projet Européen de e-maintenance Proteus, qui a pour objectif le développement d'une plate-forme distribuée, coopérative fournissant un ensemble d'aide relatif aux différentes activités de maintenance. La plat-forme illustrée par la fig 1 intègre de nombreux systèmes et bases de connaissances. Ces différents systèmes sont : une GMAO (Gestion de la Maintenance Assistée par Ordinateur) ; un SCADA (Supervisory Control and Data Acquisition); un système de gestion des données techniques et de e-documentation et enfin un système d'aide à la décision en diagnostic et réparation des équipements industriels dont nous sommes en charge. Une première étude sur ces systèmes nous a orienté vers les systèmes de RàPC. En effet, dans le domaine de diagnostic technique de nombreux systèmes de RàPC ont été expérimentés et certains d'entre eux sont commercialisés et utilisés avec succès, des applications concernant des entreprises comme PSA, British Airways etc…. Nous avons réalisés une étude comparative de sept systèmes de diagnostic, à savoir CREEK, PAD'IM, CaseLine, CheckMate, MAIC, Cassiopée et LADI. Sur sept systèmes cinq ont été développés à l'aide de logiciels commerciaux comme ReMind, CasePoint, Kaidara, CBR Express, ReCall etc…. L'achat d'un système de développement s'avérait-il opportun pour résoudre notre problème ?

Bon nombre de ces systèmes ne développent pas toutes les phases du RàPC, contrairement à PAD'IM et CREEK. Par exemple, les RàPC conversationnel ne présentent pas la phase d'adaptation. Dans le cas où cette phase est proposée comme dans KAIDARA, les utilisateurs sont amenés à acheter plusieurs produits de la société distribuant l'outil de développement. En effet il faut agencer un ensemble de modules pour réaliser les différentes phases du raisonnement à partir de cas : module de création d'une base de cas de sa mise à jour et de sa maintenance, module destiné à la recherche des cas similaires et enfin modules d'adaptation de cas qui permet de créer des règles simples. Cette modularité peut poser problème quant à l'interactivité des phases de recherche de cas et d'adaptation de ce cas. Cet inconvénient associé au coût de ce système de développement nous a conduit à développer notre propre système de RàPC à partir d'un éditeur d'ontologie gratuit Protégé.

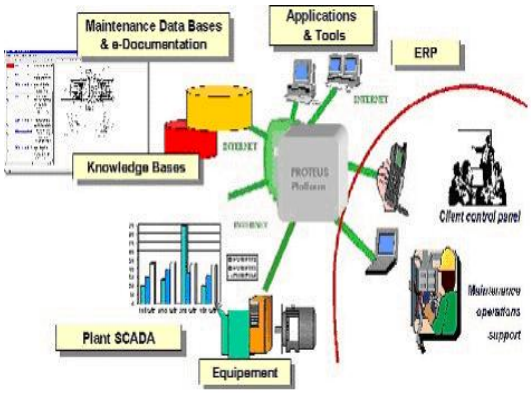


Fig. 1- Plate-forme Proteus (www.proteus-iteaproject.com)

1.2 Système d'aide au diagnostic et de réparation

Le but de notre système à développer est d'établir un diagnostic en déterminant le composant défaillant et de proposer une action de réparation adéquate avec des informations supplémentaires telles que les pièces de rechange, les outils nécessaires, la compétence requise de l'opérateur et la documentation technique appropriée. Il est centré sur l'équipement à maintenir, par un premier niveau de surveillance lié à l'équipement (la supervision des données mesurées à l'aide de capteur), un deuxième niveau élaborant le diagnostic de pannes et leur réparation qui font objet de cette communication. Ces 2 niveaux serviront de support à deux autres niveaux concernant la gestion des ressources aussi bien humaines que techniques, et la gestion des politiques de maintenance (Rasovska et al., 2004a).

L'intégration de notre système dans la plate-forme générique permet une acquisition des données relativement facile et efficace. Le système récupère le maximum d'informations et de données automatiquement en renvoyant les requêtes à

d'autres systèmes et bases de connaissances. L'ontologie[1] commune pour l'ensemble des services web proposé dans le cadre de la plate-forme permet d'employer les connaissances générales du domaine et participe ainsi à la représentation des connaissances dans notre système.

La modélisation des connaissances du domaine, suit la démarche employée par les experts du domaine de maintenance, en appliquant des méthodes d'analyse relatives à la sûreté de fonctionnement, à savoir : l'AMDEC (Analyse des Modes de Défaillance, de leur Effets et de leurs Criticités), l'analyse fonctionnelle qui a pour but la modélisation hiérarchique d'équipements et les arbres de défaillances qui identifient les défaillances possibles d'un équipement donné, voir la fig. 2.

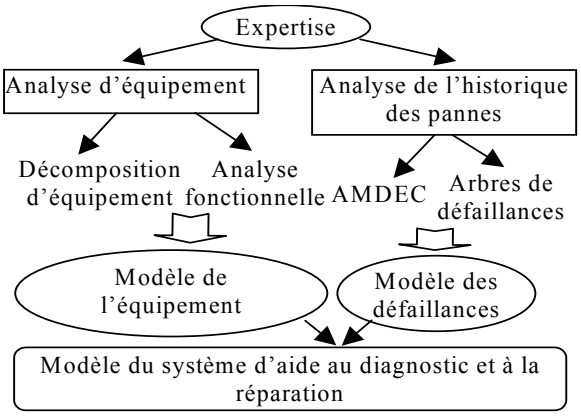


Fig. 2 - Expertise de l'équipement

Cette démarche de modélisation des connaissances du domaine, a été appliqué sur 2 types d'équipements et vous sera développé au paragraphe 2 sur une application particulière de transpalettage. Ce qui permettra de créer une structure appropriée de cas associé aux connaissances générales utilisées pour les phases de remémoration et d'adaptation.

Nous présentons au paragraphe suivant, l'étude menée sur un démonstrateur du projet Proteus : un système de transfert de palettes SORMEL. Après une brève description de l'application, les connaissances du domaine sont présentées, permettant ainsi de caractériser un composant. L'objectif de ce transfert de palette est d'assurer le bon cheminement d'une palette sur les différents postes de transit. Pour assurer son suivi, une décomposition géographique du système est proposée. De plus une typologie des composants sera exploitée dans la phase adaptation afin de proposer des actions de réparation à réaliser, associé à des ressources humaines capables d'assurer

[1] faire une ontologie, c'est décider des individus qui existent, des concepts et des propriétés qui les caractérisent et des relations qui les relient (Charlet et al., 1996). Les relations les plus classiques sont les relations d'héritage, les relations de composition. Les autres relations associant les termes n'ont pas de sémantique implicite évidente.

la réparation. Cette modélisation sera à la base de la représentation d'un cas. L'étape de remémoration des cas similaires se fera grâce aux réseaux de recherche des cas similaires (Case Retrieval Nets).

2 Application

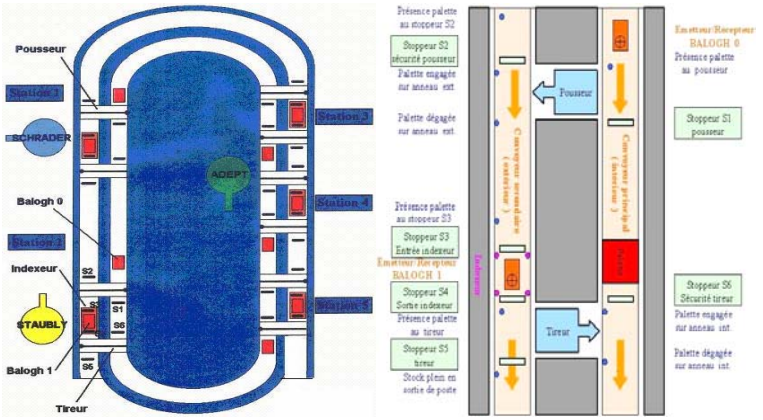


Fig. 3 - Composition du système de transfert SORMEL et composition d'une station

Le modèle du système de transfert de palettes SORMEL est présenté à la fig.3. C'est un îlot flexible d'assemblage organisé en double anneau et constitué de 5 stations de travail (partie gauche de la fig.3). Le déplacement des palettes est assuré par frottement sur des courroies, elles-mêmes entraînées par des moteurs électriques. Les palettes sont munies d'une étiquette magnétique qui leur sert de "mémoire embarquée". Ces mémoires peuvent être lues dans chaque station grâce à des plots magnétiques de lecture/écriture (BALOGH) et permettent la mémorisation d'une gamme d'assemblage de produits. Ces étiquettes permettent donc de déterminer le cheminement des palettes à travers le système. Les palettes sont véhiculées sur l'anneau intérieur permettant ainsi le transit entre les différentes stations. Lorsque l'une des palettes doit subir une opération de la part d'un robot (information lue sur l'étiquette de la palette), cette dernière est déviée sur l'anneau extérieur où se trouve le poste de travail concerné. Chaque station de travail est équipée d'actionneurs pneumatiques (pousseurs, tireurs, indexeurs) et électriques (stoppeurs) ainsi que d'un certain nombre de capteurs inductifs (capteurs de proximité D) et de capteurs magnétiques de lecture et écriture (Balogh 0 et 1), voir la partie droite de la fig.3.

2.1 Modélisation des connaissances

A partir des modèles d'équipement et de défaillances nous avons développé une ontologie des descripteurs du cas montrée sur la fig.4. L'ontologie a contribué à la modélisation et à la représentation du cas dans la base de cas. Elle nous a permis

d'identifier les descripteurs nécessaires et incontournables dans la représentation d'un cas.

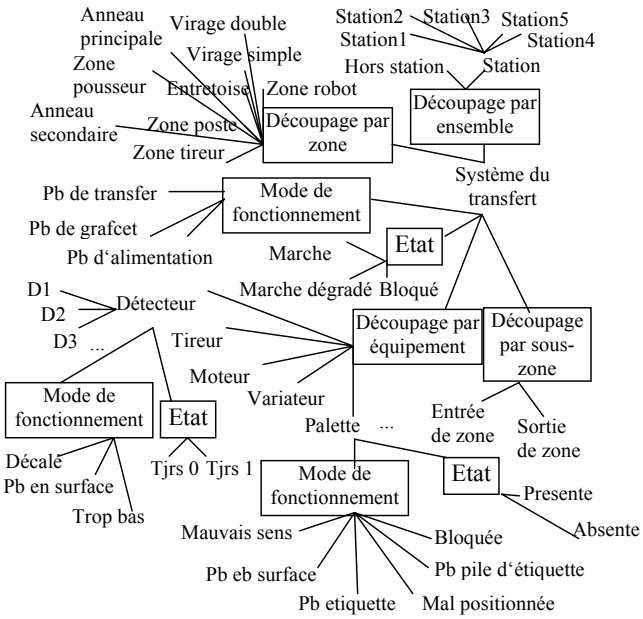


Fig. 4– Ontologie du domaine d'expertise du transfert SORMEL

2.2 Caractérisation d'un composant

Dans le cas relatif au transfert de palettes, les données issues de composants de différentes natures, à savoir capteurs, contrôleurs et unités de commande sont traitées. A un composant est associé un état et un mode de fonctionnement (table 1 - exemple concernant les détecteurs).

Table 1. Exemple des modes de fonctionnement de détecteur

Composant équipement	Symbol	Etat palette (contexte)	Etat de composant	Mode de défaillance de l'équipement action
Détecteur	D1,D2 D3,D4 D5,D6 D7,D8 D9	Présente	Tjrs 0	Pb détecteur trop bas lever détecteur
				Pb détecteur décalé pousser détecteur
				Pb détecteur défectueux changer détecteur
		Absente	Tjrs 1	Pb détecteur – pb en surface (élément métallique, poussière de fer) nettoyer détecteur

Nous identifions d'abord le contexte dans lequel le composant donné se trouve (dans le cas de détecteur de la présence de la palette - la présence ou l'absence de la

palette présente le contexte). Ensuite nous associons au détecteur son état (il peut prendre la valeur 1 s'il détecte toujours la palette ou la valeur 0 s'il ne la détecte pas). Le mode de fonctionnement du détecteur résulte de l'ensemble de contexte donné et de l'état de détecteur (si la palette est présente et le détecteur ne la détecte pas (égal à 0), le mode de fonctionnement est dégradé (le détecteur peut être mal positionné, décalé ou défectueux). Nous déterminons aussi l'action à faire pour corriger le problème.

2.3 Différentes décompositions

Typologie des composants : A partir de l'analyse fonctionnelle d'équipements nous avons construit une typologie des composants du système qui correspond à un modèle fonctionnel. Nous avons déterminé les ensembles d'équipements dont le fonctionnement est similaire ou assuré par les mêmes fonctions (électrique, pneumatique, mécanique, magnétique, etc…), voir la fig. 5. Un exemple est la classe des détecteurs fonctionnant en mode magnétique. Deux équipements assurent la même fonction de détection de la présence de palette, à savoir le détecteur et le plot magnétique Balogh qui assure la fonction supplémentaire de lecture et écriture sur l'étiquette de la palette. Ensuite ces classes ont des instances correspondant à des équipements définis tels que détecteur D1, D2, D3 etc… et Balogh0 etc.

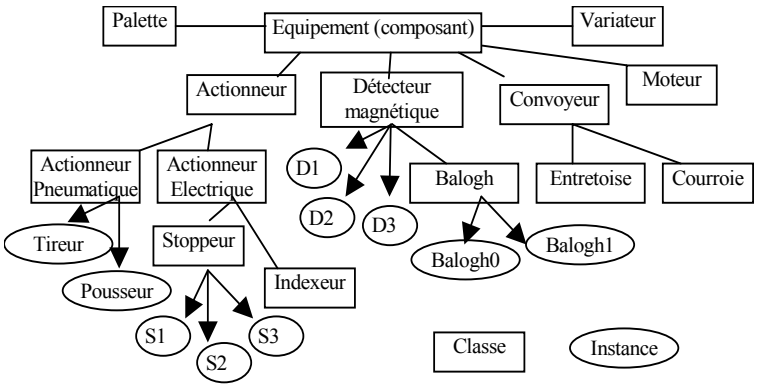


Fig. 5- Ontologie des composants du système

Découpage spatial : Le découpage spatial correspond à la décomposition géographique du système de transfert et est justifié pour le suivi de la palette qui est transférée d'un poste vers un autre et qui parcourt ainsi tous ces zones qu'on retrouve fig. 6. La plate-forme de démonstration est décomposée en ensembles, zones et sous zones auxquelles nous avons ensuite associé des composants.

Les ensembles des composants associés à chaque zone ou sous zone constituent un contexte dans lequel le composant(s) défaillant(s) est (sont) identifié(s) et déterminé(s) en comparant les valeurs et les états des différents composants dans ce contexte donné. L'ontologie des descripteurs d'un cas décrite à la figure 4 a été

implémentée à l'aide d'un éditeur d'ontologie Protégé. Protégé est un outil intégré utilisé par les développeurs de systèmes et par les experts du domaine, afin de d'élaborer des systèmes basés sur la connaissance (*http://protege.stanford.edu*).

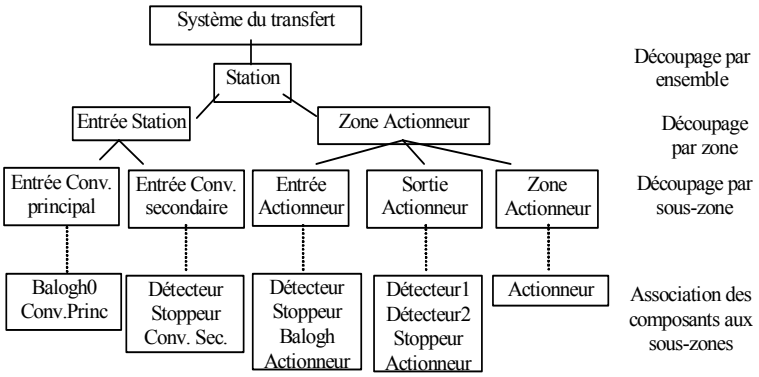


Fig. 6- Aperçu de découpage spatial du modèle (géographique)

3 Elaboration et acquisition d'un cas

La phase d'élaboration d'un cas se révèle être un aspect d'ingénierie de la connaissance important. L'élaboration d'un nouveau cas consiste à faciliter la description du problème pour permettre la recherche d'un cas dont la solution sera la plus facilement adaptable.

Il existe de nombreux travaux sur la représentation des cas, mais la représentation la plus communément reprise est la représentation structurée en liste de descripteurs qui peuvent être des objets complexes. La représentation orientée objet permet de manipuler les connaissances à caractère complexe à l'aide de différentes formes de mise en relation des classes représentant les concept du monde (Ruet, 2002). Cette représentation permet d'élaborer aisément une structure hiérarchique de descripteurs que l'on pourra exploité dans la phase d'adaptation.

La plupart des applications industrielles du RàPC choisissent de décrire les cas comme des formulaires remplis. Les informations sont saisies quand un problème se pose et sont complétées et corrigées au fur et à mesure que le problème est résolu. Dans notre cas, un formulaire est présenté sous forme de deux écrans de dialogue entre le système d'aide au diagnostic et les acteurs de maintenance via le portail web de la plate-forme. Notre formulaire est composé de questions fermées issues d'une structure arborescente (montrée à la fig.7), permettant de cerner la zone à problème, de localiser les sous zones concernées et d'identifier la classe des composants défaillants.

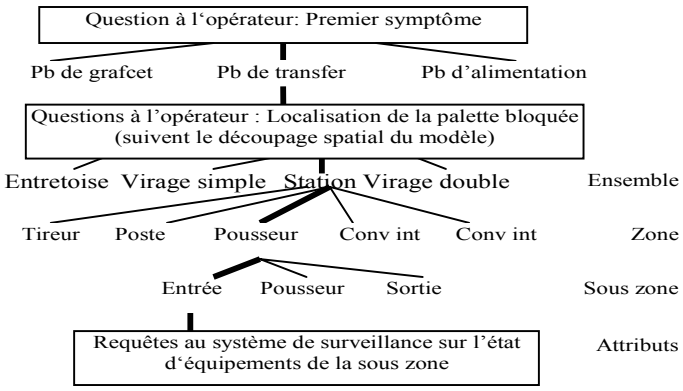


Fig. 7 - Le formulaire des questions

La démarche utilisée dans notre étude consiste à compléter ou filtrer la description d'un problème en se fondant sur la connaissance du domaine pour inférer tout ce qu'il est possible à partir d'une description éventuellement incomplète, et pour pondérer les descripteurs en fonction des dépendances identifiées entre les descripteurs problème cible et les descripteurs de la solution recherchée.

3.1 Représentation du cas

Un exemple de cas est présenté au tableau 2. Le cas contient deux parties : la description et la solution. La description du cas est un ensemble de caractéristiques : le premier symptôme, la localisation de la défaillance et un ensemble d'attributs – des équipements et les valeurs décrivant leurs états. La solution de cas consiste à faire une liste récapitulative des équipements identifiés dans le contexte avec leurs modes de fonctionnement ce qui nous mène à l'identification de la classe de composant en panne et l'action de réparation, associée dans le cas échéant à la proposition de documentations appropriées, les outils nécessaires etc... Pendant la phase d'élaboration d'un nouveau cas nous précisons donc :

- le contexte en localisant la panne qui se détermine par ensembles, zones et sous zones ;
- les composants de ce contexte et leurs états (des équipements et leurs valeurs).

A partir de la localisation des zones, sous zones etc… nous identifions les composants (détecteurs, actionneurs, Balogh etc.). Pour ce faire nous appliquons des règles de détection de défaillances, mises au point à partir d'une étude concernant les interactions entre les défaillances intervenant dans une même zone. Ces règles de détection plus fines, s'appliquant localement permettent de déterminer la classe des composants défaillants

SI ((palette est présente) ET (détecteur ne signale pas))

ALORS [détecteur défaillant OU Palette mise dans un mauvais sens]

Notre base de cas compte une vingtaine de cas

Tableau 2. Exemple d'un cas

Cas (type 1)	**Attribut : valeur**
Description	
Symptôme	Pb de transfert [pb d'alimentation, pb de grafcet, pb de transfert]
Identification du contexte : localisation	Localisation_ensemble : station [station, entretoise, virage-simple, virage_double] Localisation_zone: pousseur [tireur, pousseur, indexeur(poste), conv-int, conv-ext] Localisation_sous-zone : entrée [entrée, pousseur,sortie]
Attribut du contexte : état	Détecteur D1 : 1 [0 (palette absente), 1 (palette présente)] Balogh 0 : 1 [0 (palette n'entre pas dans la station), 1 (palette entre)] Stoppeur S1 : 0 [0 (stoppeur en haut), 1 (stoppeur en bas)] Pousseur : ne revient pas à sa position [pousse, ne revient pas à sa position, ne pousse pas]
Solution	
Mode de fonctionnement d'équipement	Symptôme pas de transfert indique : palette bloquée Localisation de la palette bloquée : station.pousseur.entrée D1 : bon fonctionnement Balogh 0 : palette entre dans la station correspond au fonctionnemnt prévu S1 : bon fonctionnement Pousseur dégradé
Action, vérification	Vérin du pousseur hors service

4 Phases de remémoration et d'adaptation

La phase de remémoration des cas similaires dépend essentiellement de la représentation des cas, de la structure de base de cas, des mesures de similarité et de l'exactitude de la réponse attendue. Dans une étude comparative des systèmes existants en diagnostic technique et réparation, nous avons identifié plusieurs mesures de similarité basé la plupart du temps sur la représentation des cas orientée objets. Les similarités étudiées sont de type statique (Keane et al., 2001), c'est à dire que les mesures prennent en compte la structure des classes et les relations entre ces classes qui sont comparées en fonction des classes communes des attributs et non dans l'aspect des relations dynamiques. L'aspect orienté processus est négligé dans ces modèles. Pourtant le contexte du processus peut aussi contribuer à la similarité conceptuelle des objets.

4.1 Mesures de similarité

Les mesures de similarité dans notre système sont adaptées à la représentation orientée objet. La similarité $Sim(O_1, O_2)$ représente la similarité globale entre deux descriptions de cas O_1 et O_2 :

$$\mathbf{Sim}(\mathbf{O}_1, \mathbf{O}_2) = \sum_{i=1}^{p} \omega_i \mathbf{sim}_i(\mathbf{o}_1, \mathbf{o}_2),$$

où ω_i est le poids de l'attribut *i, p et le nombre d'attributs et* sim_i est la similarité locale calculée pour la classe commune de deux représentations de l'attribut *i*. Pour deux objets o_1,o_2 la similarité se calcule en remontant au premier concept commun de ces objets et en comparant les slots/attributs commun à ce niveau. Par exemple, le cas dans la base de cas concerne le pousseur (table 2). La description d'un nouveau cas

correspond au tireur (table 3) et dans la modélisation des connaissances on voit que les deux concepts font partie de la classe actionneur. La classe généralisée actionneur va conduire à la solution pour le nouveau cas.

Table 3. Exemple de la description d'un nouveau cas

Nouveau cas	**Attribut : valeur**
Description Symptôme	 Pb de trasnfert
Identification du contexte : localisation	Localisation_ensemble : station Localisation_zone: tireur Localisation_sous-zone : entrée
Attribut du contexte : état	Détecteur D6 : 1 Balogh 1 : 1 Stoppeur S5 : 0 Tireur : ne revient pas à sa position

4.2 Algorithme de recherche des cas similaires

Dans la phase de remémoration des cas similaires nous avons adopté la théorie de réseau de recherche de cas (Case Retrieval Nets) présenté dans (Lenz, 1999). Les connaissances de base sont les entités d'information représentées comme les termes dans la structure ontologique auxquels nous donnons des valeurs admissibles. Un cas est un ensemble de ces entités d'information qui sont connectés par des arcs de similarité et les cas sont accessibles à partir de ces entités à travers des arcs de pertinence (partie gauche de la fig. 8).

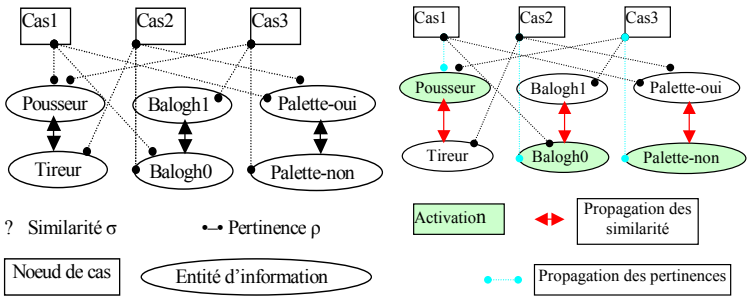


Fig. 8 – Exemple d'activation de réseau de recherche des cas similaires

La recherche des cas similaires se fait par propagation des activations ce qui nous montre la partie droite de la fig. 8. Le processus de recherche des cas similaires par propagation des activations (Lenz, 1998) se fait en trois étapes : l'activation initiale qui vérifie la présence des caractéristiques; la propagation des similarités qui compare les valeurs de caractéristiques identifiées et enfin la propagation des pertinences qui affecte une valeur caractérisant la pertinence des descripteurs étudiés.

4.3 Remémoration et d'adaptation

La décomposition du système guidée par la localisation présentée à la figure 6 sert dans un premier temps à la localisation de la défaillance et dans un deuxième temps à l'identification des composants pouvant être défaillant dans le cadre de la zone identifiée. Ensuite les états des composants sont comparés avec des règles et les modes de fonctionnement de ces composants sont déterminés ainsi que le composant défaillant. Dans la phase d'adaptation, nous allons exploiter la hiérarchie concernant les composants afin de généraliser les règles de décision. Le système de transfert est constitué de 5 stations identiques. Cela nous permet de construire des classes génériques pour adapter les solutions a chaque station relative.

L'adaptation se fera sur le diagnostic de cas s'appuyant sur la hiérarchie des objets dans le réseau de recherche des cas. Dans les travaux de Lenz sur les réseaux de recherche de cas similaires, l'adaptabilité des cas n'est pas prise en compte dans la phase de remémoration. Par contre dans notre étude, nous exploitons cette hiérarchie de descripteurs pour substituer un descripteur donné par un descripteur de la même famille et exploiter ces actions associées. Cela revient à se baser sur les actions de réparations (la solution) de la classe générique qui sont adaptées en fonction des différences entre la classe générique et les cas particuliers.

5 Conclusion

Nous avons intégré dans une plateforme de e- maintenance, un système d'aide à la décision, basé sur le raisonnement à partir de cas. Afin de ne pas limiter, notre système aussi bien sur le plan de la modélisation que sur l'interactivité des phases de remémoration et d'adaptation, nous n'avons pas employé de système de développement de RàPC, mais avons développé notre propre système à l'aide d'un éditeur d'ontologie nommé Protégé, bien adapté aux besoins d'acquisition des connaissances utilisé par notre base de cas. La modélisation retenue pour notre application est composée de 2 modèles, un fonctionnel qui sera représentée par une ontologie du domaine relatif aux composants et un modèle de décomposition spatial qui permettra de localiser la palette qui transite sur le système de transfert étudié. Un formulaire a été construit pour élaborer les cas sources. Dans la phase de remémoration, un réseau de recherche des cas similaires a été développé afin de permettre une recherche efficace orientée vers l'adaptabilité des solutions des cas sources dans le contexte de cas cible étudié.

Ce travail n'est pas terminé et nécessite des tests supplémentaires, nécessitant une génération aléatoirement de défaillances associées à une palette. Les tous premiers tests ont permis de retrouver les cas déjà prévus dans à la conception. De plus, nous envisageant gérer dans un deuxième temps un flux de palettes sur le système de transfert, ce qui compliquera aisément le problème.

Références

AAMODT, A. (1989). Towards robust expert systems that learn from experience – an architectural framework. In 3rd European Knowledge acquisition for Knowledge Based Systems Workshop, Paris 1989, pp.311-326.

AAMODT, A. ET PLAZA, E. (1994). Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches. AI Communications, 1994, 7(i): pp 39-59.

BERGMANN, R. & STAHL, A. (1998). Similarity Measures for Object-Oriented Case Representations. In Proceedings of the 4th European Workshop on Case-Based Reasoning (Eds. B. Smyth and P. Cunningham), Springer-Verlay, pp.25-36.

BISSON, G. (2001). Une approche symbolique/numérique de la notion de similarité. Dans Diday & Kodratoff (Eds.), Actes des 4ièmes journées sur induction symbolique/numérique, Orsay France, 1994, pp. 93-96.

BÖRNER, K. (1998). CBR for Design. Dans Lenz, M., Bartsch-Spörl, B., Burkhard, H.D., Wess, S. (Eds.), Case-Based Reasoning Technology: From Foundations to Applications, Lecture Notes in Artificial Intelligence, Springer Verlag.

CHARLET, J. BACHIMONT, B., BOUAUD, J., ZWEIGENBAUM, P. (1996). Ontologie et réutilisabilité: expérience et discussion. Acquisition et ingénierie des connaissances : tendances actuelles.

FUCHS, B. (1997). Représentation des connaissances pour le raisonnement à partir de cas : Le système ROCADE. Thèse de doctorat, Laboratoire Image Signal Acoustique de CPE Lyon.

KEANE, M.T., SMYTH, B. ET O'SULLIVAN, F. (2001). Dynamic similarity: A processing perspective on similarity. In Ramscar & Hahn (Eds.). *Similarity & Categorisation,* Oxford University Press, pp.179-192.

KOLODNER. &. J JONA, M.Y. (1991). Case-Based Reasnonig : An Overview. Tech report #15. Institute for the Learning Sciences, Northwestern University.

LENZ, M. (1999). Case Retrieval Nets as a Model for Building Flexible Information Systems. Thèse à l'Université de Humboldt.

LENZ, M., AURIOL, E. ET MANAGO, M. (1998). Diagnosis and Decision Support. Dans Lenz M., Bartsch-Spörl, B., Burkhard, H.D., Wess, S. (Eds.), Case-Based Reasoning Technology: From Foundations to Applications, Lecture Notes in Artificial Intelligence, Springer Verlag.

MARK, W., SIMOUDIS, E., HINKLE, D., (1996). Case-Based Reasoning: Expectations and Results, in Leake, D.B. (ed) *Case-Based Reasoning: Experiences, Lessons and Future Directions*, pp.269-294, MIT Press.

MILLE, A. (1995). Raisonnement basé sur l'expérience pour coopérer à la prise de décision, un nouveau paradigme en supervision industrielle. Thèse de doctorat, Université de Saint Etienne.

RASOVSKA, I., CHEBEL-MORELLO, B., ZERHOUNI, N. (2004 a). A conceptual model of maintenance process in unified modelling language. INCOM, sur CDROM.

RASOVSKA, I., CHEBEL-MORELLO, B., ZERHOUNI, N. (2004 b). Modélisation de connaissances de maintenance : aide au diagnostic et à la réparation.5° conférence MOSIM'04, Nantes France, 1994, pp. 511-518.

RUET, M. (2002). Capitalisation et réutilisation d'expériences dans un contexte multiacteur. Thèse au Laboratoire Génie de Production de l'Ecole Nationale d'Ingénieurs de Tarbes.

WATSON, I. ET MARIR, F. (1994). Case-Based Reasoning : A Review. The Knowledge Engineering Review.

WATSON, I. (2001). Knowledge Management and Case-Based Resaoning : a Perfect Match ? Proceedings of the 14th Int. FLAIRS Conference, Key West Florida, AAAI Press, pp 118-123.

WILKE, W., BERGMANN, R. (1996). Adaptation with the INRECA System, Proceedings of the 12th European Conference on Artificial Intelligence ECAI 96, Workshop:Adaptation in CBR, Budapest, Hungary.

Construction de solutions de cas sources en vue de leurs réutilisations combinées

Rim BENTEBIBEL , Sylvie DESPRES

Université René Descartes
CRIP5 – Equipe IAA – Groupe SBC
UFR Mathématiques et Informatique
45 rue des Saints-Pères
75006 PARIS
rim.bentebibel@math-info.univ-paris5.fr
sd@math-info.univ-paris5.fr

Résumé : Cet article s'appuie sur le paradigme du raisonnement à partir de cas qui exploite d'anciens épisodes mémorisés de résolution de problèmes appelés cas source dans le but de résoudre de nouveau cas. Dans ce papier, nous étudions la construction de solutions pour des cas sources dont la partie problème est un scénario type d'accidents. Les connaissances expertes du domaine utiles à l'élaboration de ces solutions ne sont pas encore totalement formalisées. La construction des propositions du profil (cas cible) constituera une combinaison des solutions proposées dans les scénarios types constituant le profil (cas source).

Mots-clés : RàPC, construction de solutions de cas sources, adaptation et/ou réutilisation, accidentologie.

1 Introduction

Le travail présenté se situe dans le cadre du projet GO3 « Nouvelles connaissances pour la sécurité » du PREDIT (Programme national de Recherche Et D'Innovation dans les Transports terrestres). Le projet a pour objectif de développer un système de raisonnement à partir de cas qui à partir d'un groupe d'accidents survenus dans un secteur géographique particulier, permettra d'y associer un profil de scénarios types et qui fournira un ensemble d'objectifs et de principes d'action à entreprendre pour aménager le secteur étudié. Un profil est composé d'une série de scénarios types. Tant en France qu'à l'étranger, il n'existe aucun système d'aide à l'aménagement urbain qui permette d'exploiter la notion de profil de scénarios types d'accidents. L'intérêt d'un tel outil sera d'apporter dans un premier temps une aide matérielle aux chercheurs de l'INRETS (Institut National de Recherche sur les Transports et leur Sécurité) en leur permettant de travailler sur un plus grand nombre d'accidents. Par la suite, il devrait constituer une aide aux techniciens des collectivités locales chargés de l'aménagement des voiries.

Nous présentons la manière dont nous projetons de mettre en œuvre le raisonnement à partir de cas dans ce contexte puis nous détaillons les concepts qui interviennent dans la conception du futur système. Après avoir décrit la représentation des propositions d'aménagements établie en accord avec les chercheurs, nous introduisons quelques réflexions sur la façon dont nous envisageons l'adaptation et la réutilisation des propositions d'aménagement propre à chaque scénario pour la construction de proposition associée aux profils de scénarios.

2 Présentation du système en cours de développement

Le paradigme du raisonnement à partir de cas été choisi pour répondre au problème de l'aménagement du réseau urbain en prenant en compte la sécurité. Dans ce paragraphe, nous présentons l'architecture et les ressources du système.

2.1 L'architecture du système

Le système est composé de deux modules (cf. fig1) qui font l'objet des thèses de V.Ceausu et R. Bentebibel.

- Le premier module reçoit en entrée un ensemble de procès-verbaux d'accidents survenus sur le même secteur routier. Sa finalité est l'affectation d'un scénario d'accident à chaque PV et la construction d'un profil représentatif de l'insécurité routière sur ce terrain. Un profil est composé d'une série de scénarios types, chacun affecté d'un poids correspondant aux nombres de PV du terrain d'étude qui lui sont affectés, cette série rendant compte de la majeure partie des cas survenus sur ce secteur.
- A partir des profils ainsi établis, le second module combine, adapte et pondère les moyens de prévention correspondants aux différents scénarios types pour pouvoir constituer une stratégie de prévention globale qui soit pertinente compte tenu du « profil » de l'insécurité sur ce secteur.

L'élaboration du cas cible et l'étape de remémoration sont réalisées par le premier module et la construction de la proposition d'aménagements associée au profil est réalisée par le deuxième module. L'objet de ce papier porte sur le deuxième module.

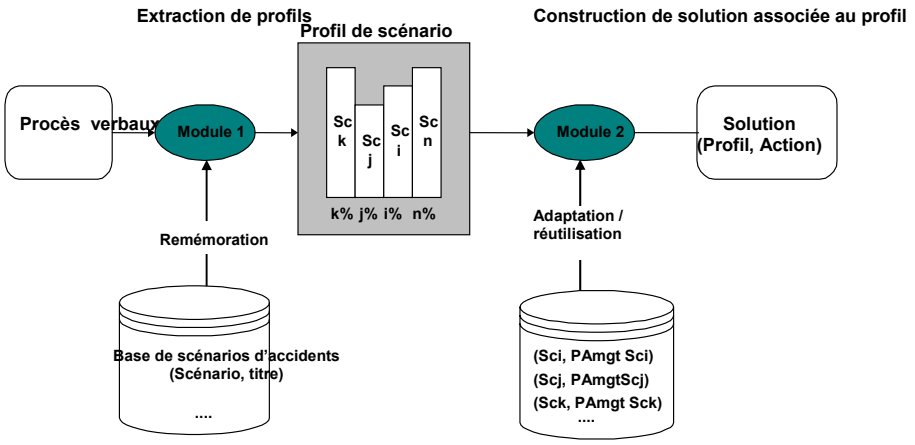


Fig.1 - Architecture du système.

2.2 Les ressources du système

La conception du futur système fait intervenir différents types de ressources : les procès verbaux (PV), les scénarios types d'accidents et les profils d'accidents. Les PV sont des données brutes tandis que les scénarios types et les profils sont des connaissances expertes explicitées par les chercheurs en accidentologie. Des représentations calculables par la machine de ces ressources ont été établies au cours des travaux menés conjointement avec le département Mécanismes d'accidents (MA) de l'INRETS [rapport INRETS n°256].

2.2.1 Les procès-verbaux

Un PV est un document semi-structuré. Il comprend des textes rédigés en langue naturelle qui décrivent l'accident (synthèse des faits, nature des faits, déclarations des impliqués, etc.) et des rubriques correspondant à des informations codées concernant les lieux, les véhicules et les personnes impliquées. Les passages en langue naturelle décrivent le déroulement de l'accident selon plusieurs points de vues : force de l'ordre (synthèse des faits), personnes impliquées (déclarations) et témoins (témoignages). L'analyse des accidents de la route est effectuée à partir de la forme électronique du PV qui a préalablement été rendue anonyme par le logiciel PACTOL (Centre d'Etudes Techniques de l'Equipement (CETE) de Rouen).

2.2.2 Les scénarios types d'accidents

Des scénarios types d'accidents ont été élaborés par des chercheurs en sécurité routière. Il s'agit d'outils conçus pour l'analyse des accidents qui constituent des connaissances expertes sur le déroulement et les conséquences des accidents routiers.

Un scénario d'accident est défini comme un prototype de déroulement correspondant à un groupe d'accidents présentant des similitudes d'ensemble du point de vue de l'enchaînement des faits et des relations de causalité, pour les différentes phases conduisant à la collision. Il est décrit selon les modèles établis par les chercheurs (cf. figure 2) du département MA.

Scénario type 8 : Conducteur **tournant** puis heurtant en sortie de carrefour un piéton traversant souvent non détecté

Situation de conduite

Un véhicule circule en agglomération dans une intersection généralement large (7 cas), **en provenance d'une voie principale** (10 cas dont 5 grandes artères), **et s'apprête à tourner à gauche ou à droite**. **Un piéton, souvent âgé** (plus de 70 ans dans 4 cas**), commence (ou s'apprête) à traverser la branche de destination du véhicule, en général sur un passage piéton** (8 cas ; dans deux autres cas : à proximité immédiate d'un passage).

Situation d'accident

Le conducteur réalise sa manœuvre de changement de direction alors que le piéton poursuit (7 cas) **ou engage** (3 cas) **sa traversée**. **Le plus souvent, le conducteur n'a pas vu le piéton** (6 cas), ou très tardivement (3 cas, 1 cas indéterminé).

Situation d'urgence

Dans la plupart des cas (8 cas), **le conducteur ne réalise pas de manœuvre d'urgence**, où celle-ci ne peut être mise en œuvre avant le choc (autres cas : 1 cas avec freinage, 1 cas indéterminé).

Situation de choc

Le véhicule heurte le piéton.

Fig 2. - Exemple de scénario d'accidents.

2.2.3 Les propositions d'aménagements associées aux scénarios types

Aux « pathologies routières » que constituent les scénarios types peuvent être associées des connaissances sur les moyens de prévention correspondant. Ces connaissances s'appuient sur de larges revues de la littérature scientifique internationale en sécurité routière. Une première étude a été menée au département MA de l'INRETS afin d'établir des scénarios types d'accidents impliquant des piétons et les éléments pour leur prévention [rapport INRETS n°256]. Les propositions d'aménagement relatives à ces scénarios sont fournies sous forme de discussion (environ 2 à 3 pages par scénario) (cf. figure 3).

Discussion
On peut estimer que ce scénario type représente environ 5,8% des accidents corporels de piéton recensés en France. Il peut être rapproché d'un type d'accident identifié […]
Des possibilités de prévention paraissent envisageables notamment dans les domaines suivants :
- la réduction des vitesses, en particulier en évitant une conception trop fluide de la géométrie des mouvements tournants en intersection […]
- l'exploitation des carrefours à feux […]
- L'apprentissage de la conduite […]
- L'éducation des piétons. Comme dans le scénario type 9 mais à un moindre degré
- […]

Fig.3 - Extrait des propositions d'aménagement.

2.2.4 Les profils de scénarios types

Lorsque les cas d'accidents du terrain d'étude ont été affectés aux différents scénarios types, il est possible de décrire le « profil » de l'insécurité routière sur ce terrain.

Un profil est composé d'un ensemble de scénarios types, chacun affecté d'un poids correspondant aux nombres de PV du terrain d'étude qui lui sont affectés, cette série rendant compte de la majeure partie des cas survenus sur ce secteur (cf. figure 4). Actuellement, nous ne disposons pas encore de profil en entrée du module 2.

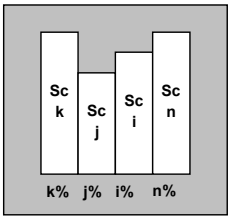


Fig. 4 - Un profil de scénarios types.

3 Modèle des ressources du cas source

Dans un système de RàPC, un cas source est un couple : (problème, solution). Dans ce contexte, le problème est constitué de la description du scénario ainsi que de la question posée par les disfonctionnements mis en évidence dans le scénario. La solution exprime l'objectif recherché (exemple : la réduction de la vitesse) et les moyens d'aménagements qui lui sont associés (exemple : éviter les conception trop fluides de la géométrie des mouvements tournants en intersection).

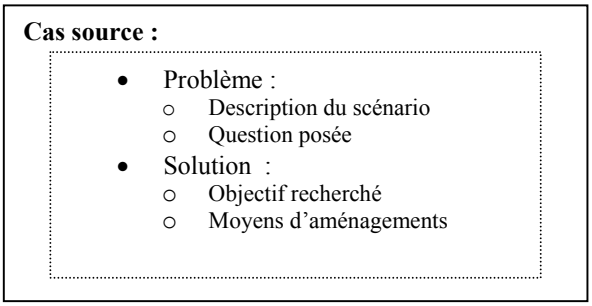


Fig. 5 - Un profil de scénarios types.

3.1 Vers un modèle de représentation des cas sources

La partie modélisation / formalisation de cette partie est dictée par la finalité de notre travail et est validée par les chercheurs. En effet, la structuration (cf. figure 5) telle qu'elle est élaborée, nous permettra la réutilisation et/ou l'adaptation lors de la prochaine phase du RàPC. La structuration des propositions d'aménagement associé à chaque scénario type est faite à partir de travaux présentés dans le rapport INRETS n°256.

Une première représentation de ces différentes propositions d'aménagement, exploitable par le système et traduisant les connaissances expertes, a été construite. Cette phase de représentation des connaissances est accomplie en collaboration avec les chercheurs du département MA. Elle présente une difficulté majeure puisque ces recherches sont menées de façon concomitante. En effet, suite à notre premier travail d'élaboration des propositions, des observations et des modifications ont été proposées par les chercheurs du domaine. Il s'agit de questions de fond car il y a peu de problèmes liés à des erreurs d'interprétation. Cette situation tient essentiellement à la formulation que nous avons adoptée et que nous avons voulu 'trans-scénarios' ce qui selon les chercheurs tend à gommer des différences dans les formulations, qui n'étaient pas, dans le travail initial, liées au hasard mais souvent à la spécificité des scénarios.

Ces propositions d'aménagements sont structurées selon deux rubriques : l'objectif de l'aménagement et les principes d'aménagement.

Un cas source est constitué d'un scénario et de suggestions concernant les actions à entreprendre. Une proposition d'aménagement est structurée selon l'objectif de l'aménagement et les principes d'aménagement (Objectif, principe d'intervention / aménagement, sous aménagement, Discussion et remarques, références).

La structuration adoptée est donc du type :

Description du scénario
- Question posée
  - Objectif recherché

moyens ⇨ Aménagement et principe d'aménagement (ex : réduction de la largeur de chaussée) , ou Objectif de l'aménagement (ex : organisation du stationnement, conception de l'éclairage public, exploitation des carrefours à feux)
  - ➢ Aménagements en détail.

La figure 6 illustre une représentation d'un cas source exprimée en XML.

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
- <Scénarios>
  - <scénario numero="1">
    - <Problème>
        <description scénario="Piéton traversant (souvent adulte, adolescent), initialement masqué souvent par véhicules
          stationné ou arrêté." />
        <Problème posé="grande largeur de l'infrastructure ==> vitesse élevée et prise d'information du piéton difficile" />
      </Problème>
    - <solutions>
      - <proposition nom="réduction de vitesse">
          <aménagement nom="Renforcement durable du système de contrôle et de sanction des excès de vitesse"
            Ref="CERTU, 1994" />
          <aménagement nom="Réduction de la largeur de la chaussée (pour une meilleur prise d'information du piéton)"
            Ref="Elvik,2001" />
        </proposition>
      - <proposition nom="Réaménagement de la voirie">
        - <aménagement nom="Aménagement facilitant les traversées de piétons et soulignant les lieux de traversée">
            <aménagement_détail nom="mise en place de refuge piéton" />
            <aménagement_détail nom="terre plein central" />
          </aménagement>
          <aménagement nom="Réduction de la largeur de la chaussée" Ref="CERTU, 1994" />
          <aménagement nom="amélioration de la position et de la conception des arrêts de bus ou cars" Ref="cf. infra" />
        </proposition>
      </solutions>
    </scénario>
  </Scénarios>
```

Fig 6 - Exemple du cas source.

Scénario type 8 **5,8% des accidents corporels**

Conducteur **tournant** puis heurtant en sortie de carrefour un

- infrastructure large ou conditions d'approche rapides en amont de l'intersection :
  => favorise une vitesse relativement élevée du véhicule dans la manœuvre tournante
  => insuffisance en matière de recherche d'informations et de prévisions
- Réduire la vitesse
  - ⇨ Eviter les conceptions trop fluides de la géométrie des mouvements tournants en intersection
  - ⇨ Eviter les carrefours trop vastes
- Prendre des mesures de réduction de vitesse en amont et en l'approche de tels sites
  - L'exploitation des carrefours à feux (gestion de feux à trois phases)

Fig 7 - Exemple de proposition d'aménagement de scénario.

La méthode utilisée a consisté en quatre étapes : - structuration des propositions d'aménagements décrites dans les scénarios types d'accidents ; identification des propositions similaires associées aux différents scénarios types ; construction d'une table contenant ces propositions et les scénarios types concernés correspondants ; la validation par les chercheurs. L'étape de validation est en cours.

3.2 Un éditeur de proposition d'aménagement

L'automatisation du processus de construction de propositions d'aménagements a nécessité l'élaboration d'un éditeur.

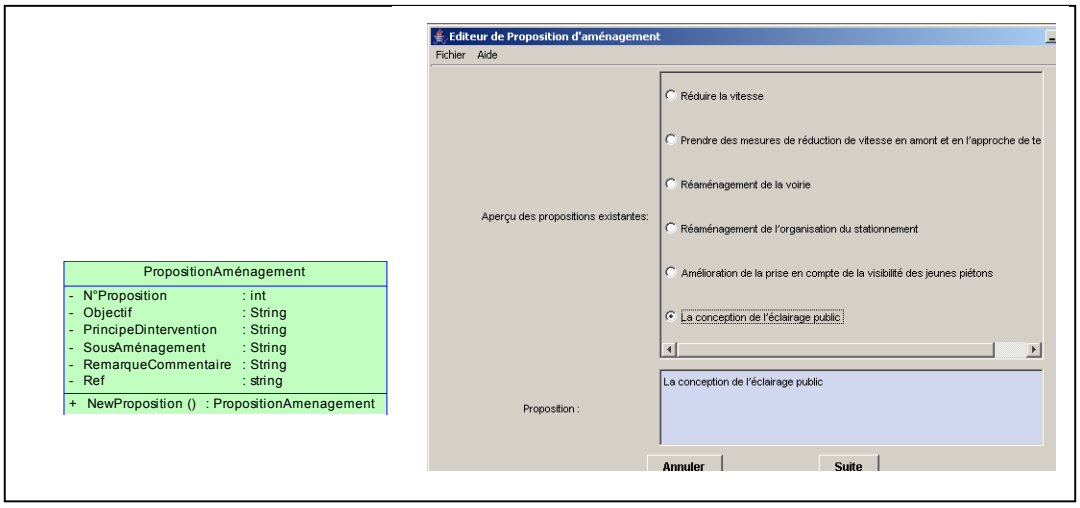


Fig8 – La classe et l'éditeur de propositions d'aménagement des scénarios

L'éditeur est en cours de réalisation, il permettra :

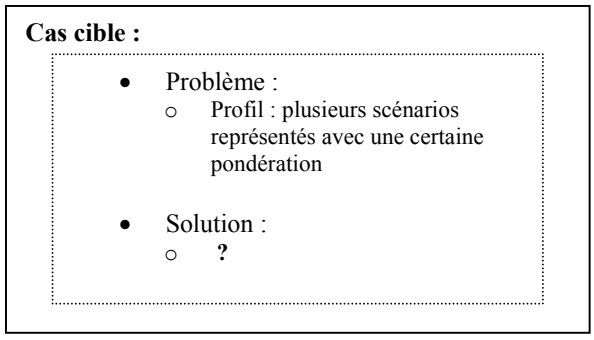
- d'alimenter la base de propositions d'aménagements avec les résultats d'aménagements des scénarios types « piéton » dans un premier temps, puis, éventuellement, de l'ensemble des scénarios types élaborés par les spécialistes du domaine.
- décrire les différentes entrées d'une proposition d'aménagement : l'objectif de l'aménagement, le ou les principes d'interventions et les sous-aménagements. Il est également possible d'y mentionner des remarques ainsi que des références bibliographiques.

La classe PropositionAménagement décrite dans la figure 8, fait apparaître les entrées d'une proposition d'aménagement. Elle sera instanciée grâce à l'éditeur de proposition d'aménagement.

4 La construction de la solution du cas cible

Un cas cible a deux parties : une partie problème et une partie solution. Le profil de scénarios constitue la partie problème ; la partie solution est à construire à partir des propositions d'aménagement associées à chaque scénario.

La construction de la solution du cas cible consiste à adapter et réutiliser les actions proposées dans les différents scénarios types intervenant dans le profil. La solution du cas cible sera déterminée à partir de celles qui correspondent aux scénarios types du profil en tenant compte de leurs pondérations. La détermination des actions nécessitera par conséquent de travailler sur les phases d'adaptation et/ou réutilisation du raisonnement à partir de cas [Fuchs et *al*., 1999].



4.1 Construction de la solution du cas cible

Pour la construction de la solution associée au cas cible, nous disposons :

- du profil contenant les références des scénarios types de ce profil et de leur pondération ;
- des propositions associées aux scénarios types.

Les scénarios types dont la pondération est inférieure à un certain seuil calculé en fonction du pourcentage pondéré d'apparition de ce scénario dans les accidents corporels, ne seront probablement pas traités. Ce seuil sera déterminé en collaboration avec les chercheurs de l'INRETS.

La détermination de l'ordre d'importance des propositions d'un profil est faite par ordre décroissant de la fonction FcO(Proposition). Une première version de cette fonction est décrite en fonction des différentes propositions des scénarios types sélectionnés du profil par la somme des pondérations des scénarios types concernés.

FcO(Proposition k) = Σ [Pondération i * Proposition k (Scénario i)] .

(Proposition k (Scénario i)=1 si le profil contient le scénario i , Proposition k (Scénario i)=0 sinon).

Exemple :

Soit le profil Profα obtenu à l'issu de la phase d'extraction de profil suivant :

Profα= {(Scénario 2, Pondération = 48%) ; (Scénario 11, 37%) ; (Scénario 16, 15%) }

Table 1 - Scénarios types/Propositions du profil Profα

Scénario 2	Scénario11	Scénario16	Proposition φ
1	0	0	Proposition a
1	0	1	Proposition b
1	1	0	Proposition c
0	1	1	Proposition d
0	1	0	Proposition e
0	0	1	Proposition f

FcO(Proposition a) = Pondération(Sc2) = 48 Pts.

FcO(Proposition b) = Pondération(Sc2) + Pondération(Sc16) = 48+15 = 63 Pts.

FcO(Proposition c) = Pondération(Sc2) + Pondération(Sc11) = 48+37 = 85 Pts.

…etc.

L'affichage des propositions relatif à un profil se fera alors par ordre décroissant des FcO(Proposition φ) .

Remarque : Pour éviter de mettre en valeur l'importance d'une proposition qui obtient par exemple 30 Pts, et une proposition qui a 2Pts, un affichage par classes est envisageable (exemple : *classe1 : entre 100 et 50Pts ; classe 2 : entre 49 et 20Pts; Classe 3 : entre 19 et 5Pts ; classe 4 : entre 3 et 0Pts*)

5 Conclusion

La modularité de ce projet permet la réalisation des deux étapes indépendantes l'une de l'autre. La première étape (thèse de V.Ceausu) au cours de laquelle des PV routiers sont apparentés à des

scénarios type et des profils de scénarios sont construits. La seconde étape (thèse de R. Bentebibel), présentée dans cet article et qui débute, met en place la réalisation du module de construction de propositions associées au profil. Le travail actuel est principalement centré sur l'automatisation structurée des propositions d'aménagement des scénarios décrits dans les discussions du rapport INRETS n°256 [T. Brenac & *al.*, 2003] ainsi que l'élaboration d'un éditeur de proposition d'aménagement relatif aux scénarios types. La prochaine étape concerne la construction de propositions d'aménagements relatives au profil donné en entrée par le premier module d'extraction des profils.

Références

AAMODT A.- A Knowledge-Intensive, Integrated Approach to Problem Solving and Sustained Learning. Doctoral dissertation, Trondheim University, TrondHeim, Norway, May 1991.

AAMODT A., Plaza. E.- Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches. AI Communications, 7(1):39--58, mars 1994.

BRENAC T., NACHTERGAËLE C., REIGNIER H. – Scénarios types d'accidents impliquant des piétons. RTS (Décembre 2003).

DESPRES S., CEAUSU V. – Raisonnement à partir de cas pour contribuer à améliorer l'aménagement du réseau urbain en prenant en compte la sécurité. (2004)

DESPRES S. - "Contribution à la conception de méthodes et d'outils pour la gestion des connaissances", Habilitation à Diriger des Recherches en Informatique, Université René Descartes, décembre 2002.

FERRANDEZ, F., FLEURY, D., MALATERRE, G. (1986) – « L'étude Détaillée d'Accidents (EDA) : une nouvelle orientation de la recherche en sécurité routière. » Recherche Transports Sécurité, n° 9-10, pp.17-20.

FLEURY, D. (1998) – Sécurité et urbanisme. La prise en compte de la sécurité routière dans l'aménagement urbain. Presses de l'Ecole Nationale des Ponts et chaussées, 299p.

FUCHS, B., LIEBER, J., MILLE, A., NAPOLI, A., *Towards a unified theory of adaptation in CBR*, Proceedings of the third International Conference on Case-based Reasoning, ICCBR-99, Lecture notes in Artificial Intelligence, Springer Verlag, seeon, Germany, 25-27 august 1999.

INRETS - L'étude détaillée d'accidents orientée vers la sécurité primaire. Méthodologie de recueil et de préanalyse. Sous la direction de Francis Ferrandez, avec la collaboration de Thierry Brenac, Yves Girard, Daniel

LECHNER, JOURDAN, J.L., MICHEL, J., NACHTERGAËLE, C., -Presses de l'Ecole Nationale des Ponts et chaussées, 1995.

LAMONTAGNE, L. & LAPALME, G. (2002). Raisonnement à base de cas textuels. Etat de l'art et perspectives. RSTI – RIA. Volume 15 – n°3/2002, pp. 339 à 366.

REYNIES, A. (2002) – Classification et discrimination en analyse de données symboliques. Application à l'exploitation de scénarios d'accidents routiers. Thèse de l'Université Paris IX-Dauphine, 2002.

TOBO A., EL AICHI M., GIBOIN A., ITEY P., MATTA N., CORBY O., DIENG R**.** (1999) – RESEDA – un serveur de connaissances pour guider l'analyse des accidents de la route. Rapport final du contrat DRAST, n.97mt29, INRIA, Sophia Antipolis.

Exécution adaptative par observation et analyse de comportement dans un contexte de jeu éducatif

Karim Sehaba, Pascal Estraillier

L3i-Université de La Rochelle
Pôle Sciences et Technologie
17042 La Rochelle Cedex 1 – FRANCE
{ksehaba, pestraillier}@univ-lr.fr

Résumé : Les logiciels ludo-éducatifs classiques ont généralement une construction statique, permettant d'adapter les jeux qu'ils proposent seulement par rapport au profil de l'utilisateur, ce qui limite la qualité des résultats qu'ils sont susceptibles de fournir. Notre architecture utilise le raisonnement à partir de cas pour faire évoluer dynamiquement l'exécution des jeux en tenant compte naturellement du profil du joueur, mais aussi des consignes du concepteur du jeu et surtout du comportement du joueur. En particulier, nous observons les actions du joueur au cours de la session afin de "comprendre" son comportement et d'y répondre, de manière personnalisée et en temps réel. Nous avons appliqué nos résultats à un environnement logiciel conçu pour des enfants autistes par une équipe de pédopsychiatres.

Mots-clés : Architecture des logiciels ludo-éducatifs, Raisonnement à partir de cas, Adaptation, Comportement.

1 Introduction

Un des enjeux majeurs des médias interactifs, et en particulier *des jeux éducatifs*, consiste à maîtriser en temps-réel la chaîne de traitements complexes mettant en jeu une série d'extraction d'informations, d'optimisation de traitements, de comparaison avec des expériences passées en vue de déterminer la meilleure des réactions à provoquer.

Dans ce contexte, notre recherche consiste à définir un système capable de *"comprendre"* le comportement d'un *joueur* et d'y répondre, de manière personnalisée et en temps réel, par des activités adaptées en tenant compte des *consignes* du concepteur du jeu. L'objectif à terme est de faire interagir efficacement différents niveaux d'analyse de scènes et/ou d'événements pour les intégrer dans un système d'observation et d'analyse de comportement en vue d'aide à la décision.

Le contexte applicatif s'articule autour du projet *Autisme*, en partenariat avec le service de pédopsychiatrie de l'hôpital de La Rochelle. Il s'agit de proposer aux enfants souffrant d'autisme, un processus d'apprentissage efficace de consignes par manipulations interactives d'outils informatiques.

Dans un premier temps, nous avons établi une cartographie des comportements d'enfants présentant des troubles autistiques. L'observation de ces comportements,

lorsqu'ils sont inscrits dans un registre de significations, réside dans la mise en évidence d'indicateurs précoces de risque de passages à l'acte ou même d'actions parasites comme la rupture, la colère, l'évitement,... Cette analyse permet, dans une seconde étape, d'améliorer l'adéquation entre les actions du système et les comportements des enfants dans le but de les maintenir attentifs et donc réceptifs vis-à-vis de l'outil informatique.

Il s'agit donc, à partir de différentes sessions de jeu, de faire évoluer une base de cas (bien évidemment qualifiés par l'équipe médicale). Lorsqu'une activité est en cours, il sera nécessaire d'identifier, à partir du comportement de l'enfant, le cas le plus approprié pour choisir, en fonction des directives du concepteur du jeu, l'activité correspondant à la meilleure évolution possible de la session de jeu.

Un paradigme adapté …

Le raisonnement à partir de cas est un paradigme de résolution de problèmes s'appuyant sur la réutilisation d'expériences passées, stockées dans une *base de cas*, pour résoudre de nouveaux problèmes appelés *cas cibles*. Le principe fondamental de ce paradigme consiste à chercher dans la base de cas des cas similaires au cas cible et de les adapter à la nouvelle situation du cas cible. Toute nouvelle expérience peut être mémorisée dans la base de cas la rendant immédiatement disponible pour les problèmes futurs. Le raisonnement à partir de cas (RàPC) est utilisé dans divers domaines d'applications telles que la supervision industrielle (Fuchs, 1997), l'aide à la navigation dans l'hypermédia (Trousse et *al.*, 1999), l'accidentologie et le droit médical (Despres, 2002),... On s'intéresse ici à l'utilisation du RàPC dans le domaine des systèmes d'apprentissage et des jeux éducatifs. Dans ce cadre, le RàPC a surmonté les limitations des approches basées sur une base de règles comme le souligne (Watson & Marir, 1994).

De fait, de nombreux systèmes d'apprentissage (Elorriaga & Fern, 2000) (Funk & Conlan, 2002) utilisent le RàPC dans leur processus de raisonnement pour générer des activités éducatives. Ces travaux mettent en avant l'aspect d'adaptabilité des activités proposées par le système par rapport au profil de l'utilisateur, mais sans tenir compte des aspects liés à la *dynamicité* du système, et à *l'observation du comportement* de l'utilisateur. Pourtant, dans de nombreuses applications liées aux logiciels éducatifs et thérapeutiques pour enfants, ces deux aspects sont importants pour plusieurs raisons :

- Les utilisateurs évoluent et les activités proposées par le système peuvent devenir incohérentes au cours de la session d'apprentissage ce qui motive le choix d'un système dynamique.
- L'analyse comportementale est un facteur clef pour les thérapeutes. Son intérêt réside dans la mise en évidence d'indicateurs précoces qui peuvent alerter sur le risque de décompensations, de passages à l'acte ou même d'actions parasites (comme des stéréotypes).
- Cette analyse permet d'améliorer l'adéquation entre les actions du système et les comportements des utilisateurs dans le but de les maintenir attentifs et donc réceptifs vis-à-vis de l'outil informatique.

Nous proposons une architecture d'un système permettant de :

- Construire une séquence d'activités répondant aux buts éducatifs que l'utilisateur veut/doit atteindre en tenant compte de son profil.
- Observer en permanence les actions de l'utilisateur au cours de la session et leurs associer une interprétation de son comportement. L'observation est faite sur les actions effectuées sur les périphériques : souris, écran tactile, clavier… et sur les gestes, mouvements corporels, orientations des yeux…
- À partir de son observation, le système détecte les cas où les activités proposées ne répondent plus à l'évolution actuelle du comportement de l'utilisateur et met à jour la séquence d'activités en pareil cas.

La structure de cet article est la suivante : la deuxième partie définit les termes utilisés dans notre approche. La partie suivante donne une description générale de l'architecture du système. Nous détaillons ensuite le processus de décision. Il s'agit de présenter le modèle de représentation des cas, le processus de raisonnement ainsi que le mécanisme de gestion d'exceptions. Enfin, la dernière partie présente quelques résultats de l'application.

2 Définitions – Hypothèses

Avant de présenter notre approche, il nous semble nécessaire de présenter certains termes. Ainsi,

- Un ***Jeu*** possède des paramètres de configuration et des objectifs à atteindre. Il est caractérisé par :
  - un ***Décor*** statique,
  - une collection d'***Objets*** (pictogramme, boule, image, musique,...) statiques ou munis de comportements qui définissent les formes d'interactions associées.
  - des ***Actions*** possibles du joueur, en particulier la manipulation des objets au moyen de l'IHM.
  - des ***règles*** de fonctionnement (les règles du jeu) relatives aux objets et aux actions possibles.
- Une ***Activité*** est une instance d'un jeu (avec une configuration particulière et des objectifs qualifiés et quantifiés).
- Un ***Protocole*** est une séquence d'activités ordonnées, déterminée en vue de permettre aux joueurs de réaliser des objectifs complexes.
- Un ***Cas*** est un protocole associé à un certain profil du joueur, lui permettant d'atteindre des objectifs parfaitement caractérisés (qualitativement, quantitativement).
- Une ***Consigne*** (ou ***directive***) caractérise un état du système (et en particulier son évolution liée au comportement du joueur) et lui associe des traitements qui s'adaptent au cours de l'activité.

Du point de vue de l'utilisateur, nous distinguons le ***Joueur*** (au sens classique du terme) de ***l'Expert***. Le Joueur est défini par un profil le caractérisant. L'Expert est

caractérisé par son savoir-faire dans un domaine donné. C'est lui qui définit le Jeu, détermine les Activités associées et définit les différents Cas en adaptant un Protocole au profil du Joueur. Son rôle consiste également à définir les Consignes associées au Joueur.

3 Présentation générale de l'architecture

L'architecture générale que nous proposons, présentée dans la figure 1, est construite à partir de trois sous systèmes : un système d'observation et d'analyse de comportement, un système de décision, et un système d'action.

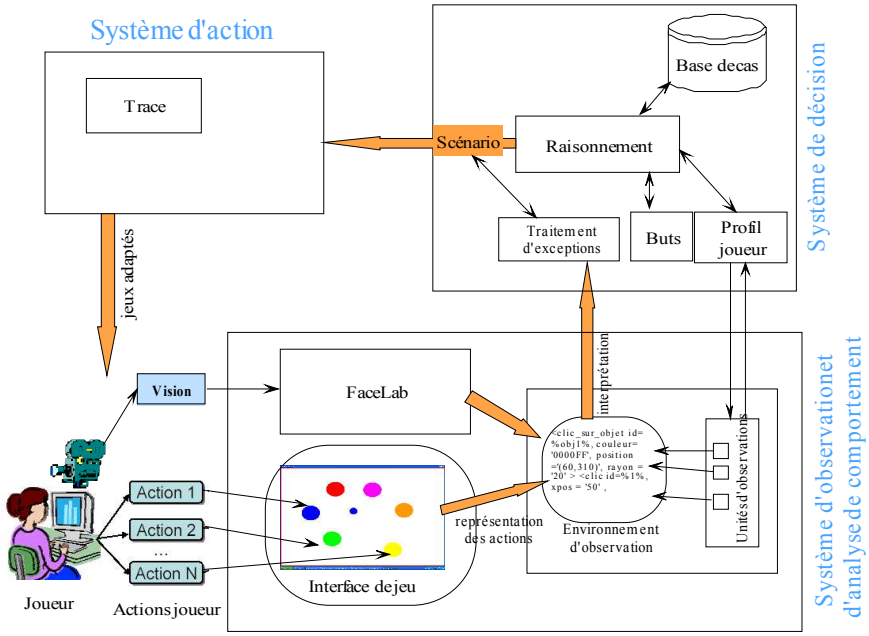


Fig. 1 – L'architecture générale.

Système d'observation et d'analyse de comportement : En s'inspirant des études issues de l'Interaction en Langue Naturelle (Sabah, 1990) (Allen et al., 2001) (Sansonnet, 1999), plus particulièrement de *la théorie des affordances* (Gibson, 1977) (Gibson, 1979) *et la théorie de la sémantique procédurale* (Wittgenstein, 1953) (Johnson-laird, 1977) (Woods, 1981), ce système est chargé de récupérer les actions du joueur et de leurs associer des mots d'état caractérisant son comportement. L'observation est faite selon deux approches, *Action logicielle* et *Vision*. Dans la première approche, il s'agit de récupérer les actions du joueur sur les périphériques : souris, écran tactile, clavier… Dans l'approche vision, assurée par le système logiciel-

matériel **FaceLab**, il s'agit de mesurer les caractéristiques concernant la représentation 3D du visage et de l'orientation du regard.

Système de décision : ce système utilise le raisonnement à partir de cas (Kolodner, 1993), pour générer des protocoles adaptés au profil du joueur. Les protocoles engendrés par ce système peuvent et doivent être modifiés lorsqu'ils se révèlent obsolètes au cours de la session d'apprentissage. Ce qui garantit une exécution adaptative.

Système d'action : ce système est chargé d'exécuter les activités du protocole fourni par le système de décision. Une autre tâche importante de ce système est la sauvegarde de la trace d'exécution. La trace d'exécution doit permettre de retrouver exactement ce que le système avait prévu de faire faire au joueur (les activités qui lui proposées et ce qu'il avait réalisé). Ces informations ont plusieurs intérêts. Dans le contexte de l'autisme, la trace peut permettre de noter l'évolution de l'enfant; elle sera aussi à l'origine de certaines règles : " si l'on a déjà présenté un tel jeu la semaine dernière, alors proposer un autre cette semaine" ou encore, " si l'enfant n'a pas eu d'activité de tel type depuis un mois, alors on introduit cette activité dans la session ",…

4 Système de décision

Le système de décision doit fournir au système d'action, un protocole adapté au profil du joueur et aux buts éducatifs à atteindre. Pour assumer cette tâche, ce système contient cinq modules : **Buts**, **Profil du joueur**, **Base de cas**, **Raisonnement** et **Traitement d'exceptions**.

Dans chaque session, certains **buts** éducatifs sont en activité dans le système de décision et certaines informations caractérisant le joueur, sont tenues dans le module **Profil du joueur**. Étant donnés les buts existants et le contenu du profil du joueur, le module de **Raisonnement** permet de générer un protocole *adapté* à ces informations. Le protocole généré sera ainsi placé dans le système d'action pour son exécution. Le module de raisonnement utilise le raisonnement à partir de cas (Kolodner, 1993) qui consiste à réutiliser les connaissances spécifiques des expériences précédemment expérimentées appelées **cas sources** et stockées dans une **base de cas**, pour résoudre de nouveaux problèmes. Un nouveau problème, appelé **cas cible**, est résolu en *sélectionnant* un cas source similaire à celui-ci. Le cas sélectionné sera *adapté* à la nouvelle situation du cas cible. Toute nouvelle expérience dont le degré de similarité aux cas sources excède un certain seuil sera *mémorisée* dans la base de cas, la rendant immédiatement disponible pour des problèmes futurs.

Atteindre un but revient à construire un protocole. Mais il ne s'agit pas d'une exigence rigide, les protocoles engendrés par le système de décision peuvent et doivent être modifiés lorsqu'ils se révèlent désuets dans le cas de l'apparition d'événements exceptionnels au niveau des actions du joueur. Les exceptions sont utilisées pour notifier une situation indésirable qui empêche l'exécution standard du protocole. **Le module de traitement d'exceptions** permet d'identifier, de signaler et de traiter ces exceptions.

4.1 Représentation des cas

Les connaissances sur la façon d'accomplir les buts sont représentées dans le système par un ensemble de cas sources. Dans notre architecture, un cas source contient deux composants : *une description* et un *protocole d'activités.*

- **La description de cas :** ce composant contient des *descripteurs* liés aux buts et au profil du joueur. Les descripteurs sont représentés par un couple de la forme « attribut, valeur ». Les descripteurs liés aux buts décrivent la nature des buts. Par exemple : *« Perception-auditive, haut »*, *« Perception-visuelle, moyen »*. Les descripteurs liés au profil contiennent des informations concernant les connaissances et les préférences du joueur - par exemple : *« Type-joueur, débutant »*, *« couleur, vert »*, *« Durée-session, [15.30] »*. Les descripteurs des cas sont utilisés pour calculer le degré de similarité entre le cas cible et les cas sources dans le processus de raisonnement (voir 4.3).
- **Le protocole** : ce composant contient une séquence d'activités.

Une fois que la structure d'un cas a été présentée, un autre problème de base est celui de l'organisation de la base de cas (Schank, 1982) (Schank, 1989).

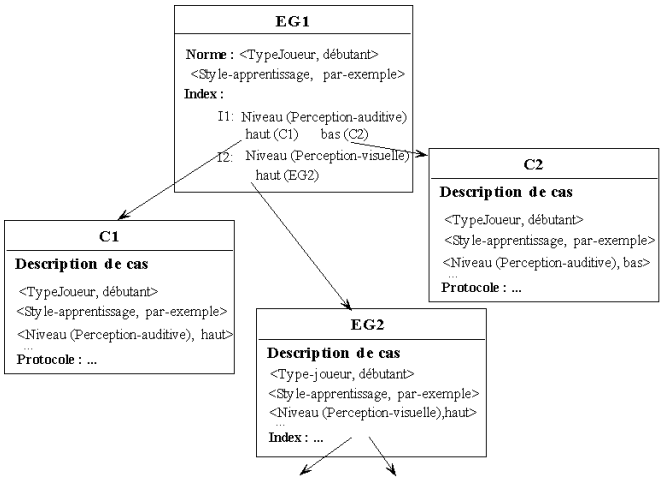


Fig. 2 –Structure de la base de cas.

L'organisation de la base de cas que nous adoptons est basée sur le modèle à mémoire dynamique de Schank (Schank, 1982). L'idée fondamentale est d'organiser les différents cas ayant des similarités propres sous la forme d'une structure plus générale appelée *épisode généralisé* ou *EG*. Un *EG* contient trois types d'objets :

- ***Norme (Norm)*** : elle représente les données communes à tous les cas indexés sous *l'EG*. Elle est représentée par une liste de couples de la forme « attribut, valeur ».
- ***Cas (Cases)*** : ils représentent des expériences individuelles.

- ***Index (Indices)*** : ils représentent les différences entre les cas d'un même *EG*. Chaque index est lié à un attribut concret d'un descripteur et contient une liste de paires « valeur, nœud ». Chaque paire d'index lie son *EG* avec un autre nœud (cas ou *EG*) correspondant.

Ainsi se forme un graphe hiérarchique (voir l'exemple de la figure 2) dont les nœuds sont soit des *EGs* soit des cas. Les arcs représentent les liens entre les index et les nœuds.

4.2 Profil du joueur

Le profil du joueur a des fonctionnalités multiples, utilisé à différents moments par le système de décision, plus particulièrement, dans le processus de raisonnement présenté à la section 4.3. Il intervient aussi dans l'interprétation des actions du joueur par le système d'observation et d'analyse de comportement. Plusieurs types d'informations concernant le joueur sont présents :

- Des informations générales
- Des connaissances du domaine
- Des préférences
- L'histoire

Les informations générales concernent l'identité du joueur, p. ex. : nom, prénom, date de naissance,... Les connaissances du domaine et les préférences donnent une description du profil du joueur identique à celle mentionnée dans la description de cas (voir la section 4.1). Ces informations sont donc représentées par une liste de couples de la forme « attribut, valeur».

L'historique est vu comme un agenda de tout ce qui est arrivé au joueur lors de chacune des sessions. Il doit permettre de retrouver exactement ce que le système avait prévu de faire faire au joueur, les activités qui lui ont été proposées et ce qu'il avait réalisé.

4.3 Processus de raisonnement

Le module de raisonnement du système de décision utilise le raisonnement à partir de cas (Kolodner, 1993), pour générer des protocoles en sélectionnant des cas sources, de la base de cas, similaires au cas cible et en les adaptant à la situation actuelle du cas cible. Le nouveau cas généré peut être mémorisé dans la base de cas s'il répond à certains critères détaillés par la suite. Nous avons recensé trois étapes dans le processus de raisonnement : *La remémoration*, *L'adaptation* et *La mémorisation*.

4.3.1 La remémoration

La remémoration des cas sources similaire au cas cible est une étape qui intègre deux processus, *appariement* et *recherche*. Le processus d'appariement utilise une fonction qui calcule le degré de similarité entre les cas. Le processus de recherche

consiste à élaborer des méthodes d'investigation optimales dans la base de cas qui tiennent compte de sa structure et de ses propriétés.

L'appariement est défini dans (Kolodner, 1993) par *"Matching is the process of comparing two cases to each other and determining their degree of match"*. La méthode d'appariement que nous avons utilisée est *le plus proche voisin* (k-nearest-neighbors). Cette méthode, utilisée dans (Elorriaga & Fern, 2000) (Voß, 1994) (Leake, 1996), possède l'avantage d'être *"très simple à implémenter"* (Bisson, 2000) et le fait qu'elle utilise directement la notion de similarité pour mesurer la correspondance entre chaque cas source et cas cible. La similarité est mesurée en termes d'attributs appropriés des descripteurs de cas (cible ou source). Rappelons que les descripteurs d'un cas concernent les buts et le profil du joueur.

Nous distinguons deux filtres dans le processus d'appariement, le premier filtre sélectionne les cas sources dont la similarité avec le cas cible est supérieure à un seuil défini par l'expert. Le deuxième filtre sélectionne les cas sources qui minimisent l'effort d'adaptation.

Le premier filtre utilise la fonction numérique Φ qui calcule le degré de similarité entre un cas cible *C* et un cas source *S*. La fonction Φ est définie comme étant la moyenne pondérée des valeurs de similarité sur chacun des descripteurs de *C* et *S*. Le schéma (1) montre la fonction de similarité Φ où :

- *D* est l'ensemble des descripteurs du cas cible *C*.
- W_d est le coefficient d'importance du descripteur *d*. Les descripteurs les plus importants doivent être pris en considération, dans le calcul de la similarité, avec des valeurs plus élevées que d'autres descripteurs moins importants.
- $\varphi_d(C,S)$ est la similarité entre deux valeurs du descripteur *d* associés à *C* et *S*.

$$\phi\ (C,S) = \frac{\sum_{d \in D} W_d * \varphi_d(C,S)}{\sum_{d \in D} W_d} \qquad (1)$$

Le deuxième filtre minimise l'effort d'adaptation des cas sources à la situation actuelle du cas cible. L'effort d'adaptation est calculé en fonction des buts et des descripteurs du cas cible non assurés par le cas source. Plus ces données sont importantes plus l'effort d'adaptation est important, moins la sélection du cas source est retenue.

La recherche dans la base de cas dépend fortement de sa structure. Dans une structure hiérarchique, la recherche est guidée par les attributs partagés de chaque *EG* (normes) et les index. Ce problème de recherche présente quelques propriétés spéciales. D'abord, la structure hiérarchique de la base de cas permet une recherche heuristique guidée par la fonction de similarité. Cette dernière est limitée aux attributs des descripteurs du cas cible inclus dans la norme de l'*EG*. Une autre caractéristique de la recherche réside dans l'absence d'un critère d'arrêt : un cas partageant tous les attributs du cas cible est rarement trouvé. Par conséquent, le critère de sélection est basé sur une comparaison entre la valeur de la fonction de similarité et le seuil. Pour

chaque cas source dont la valeur de la fonction de similarité avec le cas cible dépasse le seuil sera retenu comme cas candidat pour la deuxième étape (l'étape d'adaptation).

Ces propriétés nous permettent d'établir des techniques de sélection basées sur les méthodes classiques de recherche d'intelligence artificielle. Plusieurs algorithmes de recherche peuvent être combinés (p. ex.: *A* recherche, Hill-climbing search*).

Une stratégie de sélection établit des appels à des algorithmes de recherche de base avec leurs paramètres. Les paramètres d'un appel sont *l'identifiant de l'algorithme*, *la fonction heuristique* (quand c'est un algorithme heuristique), *la fonction de similarité* et *le seuil de sélection*.

4.3.2 L'adaptation

L'étape d'adaptation permet de modifier les cas candidats sélectionnés lors de l'étape de remémoration, pour qu'ils répondent aux mieux à la description du cas cible. Dans la plupart des systèmes, l'étape d'adaptation nécessite une intervention humaine pour compléter une solution partielle ou tout simplement pour générer une solution entièrement à partir des cas. Ceci est dû à la difficulté de l'implémentation de cette étape comme le souligne (d'Aquin *et al*. 2004) et à la nécessité d'une base de connaissances intensives (Elorriaga & Fern, 2000) et un coût en termes de temps et d'efforts.

Étant donné que notre application est destinée aux enfants autistes accompagnés, une intervention de l'accompagnateur au cours de la session peut perturber l'enfant avec un risque de rupture, ce qui motive une stratégie d'adaptation automatique. La stratégie que nous avons adoptée distingue deux types d'adaptation, *globale* et *locale*.

Dans l'adaptation globale, il s'agit de remplacer des sous-protocoles des cas candidats par d'autres répondant au mieux aux objectifs du cas cible. Pour cela, il faut recenser les buts non assurés par les cas candidats, et chercher des activités qui peuvent compléter ces buts. Ces activités seront ainsi insérées dans les protocoles des cas candidats correspondants. Le meilleur cas candidat similaire au cas cible est considéré comme le cas répondant aux besoins du cas cible.

L'adaptation locale permet de régler les paramètres de configuration des activités du protocole du cas candidat avec des valeurs mieux adaptées à la description du cas cible, en d'autres termes adaptées aux préférences et aux connaissances du joueur.

4.3.3 La mémorisation

La mémorisation consiste à mettre à jour la base de cas après chaque session. Il s'agit d'évaluer, en termes d'apports éducatif et informatique, le nouveau cas et de le sauvegarder dans la base de cas en tenant compte de sa structure et ses propriétés. Cette étape se compose donc de deux phases : l'évaluation et la sauvegarde.

Pendant la phase d'*évaluation*, chaque protocole est évalué dans deux dimensions : *éducative* et *informatique*.

Dans la dimension éducative, le souci des données concerne juste le profil du joueur : fondamentalement, les changements des préférences et connaissances recueillies pendant la session et les erreurs faites par le joueur. Nous pensons que l'automatisation de cette dimension est difficile dans le contexte de l'autisme, dans la

mesure où l'enfant est très susceptible aux perturbations de son environnement. Ces perturbations ne peuvent pas êtres contrôlées par le système (bruits, ouverture de la porte de la salle,…). L'accompagnateur évalue donc ce qu'a fait l'enfant durant la session en tenant compte des perturbations de l'environnement et prend la décision de l'utilité de sauvegarder le cas ou non.

L'objectif de l'évaluation dans la dimension informatique consiste à estimer la qualité de chaque nouveau cas pour créer un nouveau cas source. Notre principe est que seulement les cas sensiblement différents des cas sources seront mémorisés. Il y a deux possibilités : quand le nouveau cas est assez semblable à n'importe quel cas source (par exemple quand le nouveau cas a été obtenu par une légère transformation d'un cas source), il ne sera pas mémorisé. En revanche, quand le nouveau cas a été obtenu par une transformation substantielle d'un cas source (le degré d'adaptation, globale et locale, excède un seuil défini par l'expert) ou le cas était appliqué à une situation différente (le degré de similarité excède un autre seuil défini par l'expert), un nouveau cas sera construit et ajouté à la base de cas. Une réorganisation des liens de la structure de la base est produite comme effet secondaire.

La phase de *sauvegarde* nécessite des mécanismes robustes pour maintenir la structure et les propriétés de la base de cas à chaque ajout d'un nouveau cas. Ceci implique trois étapes : le choix d'un *EG* qui contiendra le nouveau cas, l'enchaînement du nouveau cas à *l'EG*, et la généralisation des cas/*EG*.

Le choix d'un *EG* candidat pour accueillir le nouveau cas est un processus de recherche semblable à celui du processus de recherche de l'étape de remémoration. Cependant, deux différences surgissent:

- Le nœud cible est un *EG*.
- L'algorithme peut élaguer les nœuds du graphe puisque le nouveau cas doit complètement satisfaire la norme de *EG* candidat.

L'enchaînement du nouveau cas à l'*EG* est atteint en choisissant un attribut d'index qui remplit quelques conditions : la paire « attribut, valeur » doit distinguer le nouveau cas parmi les autres descendants de l'*EG*, et la valeur doit être liée.

Afin de maintenir une structure optimale de la base de cas, cette phase devrait vérifier périodiquement la base de cas après chaque insertion pour détecter des situations dans lesquelles il est recommandé de généraliser. Quand un ensemble de nœuds appartenant à un *EG* et partageant un certain nombre de descriptions est détecté, une généralisation est déclenchée; elle crée une nouvelle abstraction (*EG*) et réorganise les liens parmi les nœuds impliqués.

4.4 Gestion d'exceptions

Le système de décision établit un lien entre les informations concernant le profil du joueur et les buts pour générer un protocole adapté. L'aptitude du protocole à répondre de façon satisfaisante à l'ensemble des actions du joueur au cours de la session, en d'autres termes au cours de l'exécution du protocole, dépend directement du caractère exhaustif des événements prévus par le protocole. Cette approche ne peut convenir que pour les domaines d'applications quasi-statiques. Cette hypothèse est

contraignante dans le contexte de l'autisme dans la mesure où, d'une part, il est difficile que le protocole puisse recenser toutes les actions possibles que l'enfant peut effectuer. D'autre part, des comportements atypiques (p.ex. rupture, évitement,…) de l'enfant peuvent apparaître. Dans ce contexte et dans un souci de fiabilité du système, aussi bien lors de la phase de génération du protocole qu'en cours de son exécution, il faut que le système de décision ait la capacité de réagir à tout moment aux actions (typiques et atypiques) de l'enfant afin de pouvoir adapter le protocole aux nouveaux besoins qui apparaîtront et de nouvelles possibilités qui s'offriront.

Les actions du joueur sont considérées par le système comme des événements qui peuvent êtres des événements classiques (associés aux actions typiques) ou des événements exceptionnels (associés aux actions atypiques). Les premiers peuvent être traités par le protocole standard généré par le module de raisonnement.

Dans le contexte de la programmation, les événements exceptionnels (souvent appelés exceptions (Goodenough, 1975)) sont gérés par des mécanismes de gestion d'exceptions (Weinreb, 1983) (Meyer, 1988) (Koenig & Stroustrup, 1989). Le principe de ces mécanismes est utilisé aussi dans le contexte d'agents (Souchon *et al.*, 2004). Dans notre contexte, les exceptions sont utilisées pour notifier une situation indésirable qui empêche l'exécution standard du protocole de se poursuivre, telles qu'un évitement ou une rupture de la part de l'enfant. Le module de traitement d'exceptions fournit des structures de contrôle et des mécanismes permettant de *détecter*, de *signaler* et de *traiter* les exceptions.

La détection d'une exception interrompt l'exécution standard du protocole. Sa continuation est alors prise en charge par le module de gestion d'exceptions qui procède au signalement de l'exception, c'est-à-dire à la recherche d'un traitement pouvant être appliqué pour gérer cet événement. De tels traitements sont définis dans des gestionnaires d'exceptions. L'exécution des actions correctives, définies par les gestionnaires d'exceptions, permet soit de reprendre l'exécution là où le protocole a été interrompu (repris), soit en un autre point, en abandonnant tout ou partie du protocole en cours d'exécution au moment de la détection de l'exception (terminaison).

5 Résultats

Dans cette section, nous présentons l'implémentation de notre architecture dans le cadre du projet *Autisme*. Dans un premier temps, nous avons développé une plate-forme qui assure le processus de raisonnement et le traitement d'exceptions. Pour le traitement d'exception, nous avons développé un agent qui observe les actions du joueur pendant la session et réagit en cas d'exceptions en modifiant le protocole.

Dans une seconde étape, nous avons développé une interface permettant à l'expert de définir les activités (voir la figure 3), les cas et les distances entre les différentes valeurs des attributs de chaque descripteur. Ces informations ainsi que le profil du joueur sont stockés dans un serveur de données.

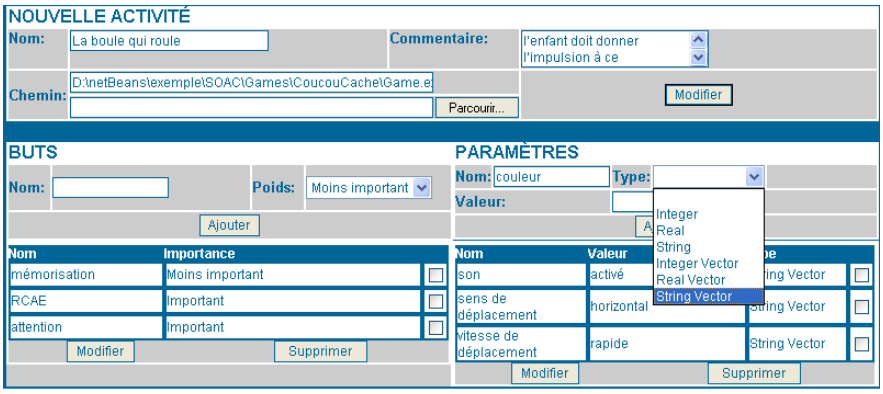


Fig. 3 – Fenêtre de création d'une activité.

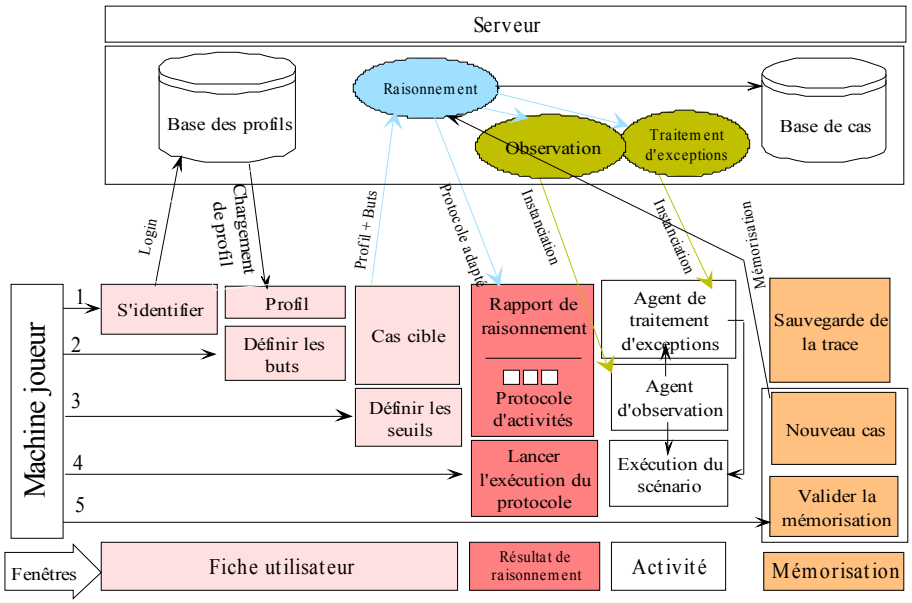


Fig. 4 –Implémentation.

La figure 4 illustre notre exemple. Chronologiquement, le joueur se connecte au serveur, son profil sera alors chargé. Ensuite, le joueur est prié de spécifier les buts qu'il veut atteindre. Une fois ces informations entrées, un cas cible est créé. Le joueur définit les seuils de raisonnement. Le cas cible ainsi que les seuils sont transmis par message au serveur. Le message est intercepté par l'agent de raisonnement. Le rôle de cet agent est de générer un protocole adapté à la situation actuelle du cas cible en utilisant le RàPC (voir la section 4.3).

Un rapport de raisonnement sera envoyé au joueur. Ce rapport contient le cas source choisi, les valeurs de la similarité et de l'effort d'adaptation entre le cas cible et

le cas source ainsi que les modifications faites au niveau des activités du protocole du cas source sélectionné. Le joueur lance les activités. Pendant la session, une instance d'agent d'observation suit en permanence les actions du joueur. Chaque exception détectée par l'agent observateur sera traitée par l'agent de traitement d'exceptions.

A la fin de la session, le joueur valide (dans la dimension éducative) ou non le cas. Dans le cas favorable, le logiciel procède à l'évaluation dans la dimension informatique pour sa mémorisation.

6 Conclusion

Dans cet article, nous avons développé une architecture permettant au système visé d'adapter de façon dynamique son exécution par observation et analyse de comportement. Le principe de l'architecture repose sur l'optimisation de l'extraction de l'information en fonction du contexte. L'observation et l'analyse de comportement consistent à déterminer le comportement du joueur à partir de ces actions, en tenant compte des consignes de l'expert. L'approche que nous avons adoptée est inspirée des travaux issus de l'Interaction en Langue Naturelle.

Nous avons utilisé le RàPC pour la prise de décision. Le RàPC nous a permis de doter le système d'un mécanisme intelligent de création d'activités adaptées au profil du joueur. Afin de doter le système de décision d'un mécanisme dynamique lui permettant de réagir à tout moment, même au cours de la session, nous avons utilisé le mécanisme de gestion d'exception qui tient compte des consignes de l'expert. Il s'agit de détecter des comportements particuliers, définis par l'expert, et de répondre par des actions au niveau du protocole.

Le système que nous avons développé est actuellement testé au service de pédopsychiatrie de l'hôpital de La Rochelle.

Références

ALLEN J., FERGUSON G. & STENT A. (2001). An architecture for more realistic conversational systems. In Proceedings of Intelligent User Interfaces (IUI-01). p. 1–8.

BISSON G. (2000). La similarité : une notion symbolique/numérique. Apprentissage symbolique-numérique (tome 2). In CEPADUES, Ed. p. 169–201.

D'AQUIN M., BRACHAIS S., LIEBER J. & NAPOLI A. (2004). Vers une acquisition automatique de connaissances d'adaptation par examen de la base de cas - une approche fondée sur des techniques d'extraction de connaissances dans des bases de données. In 12ème Atelier de Raisonnement à Partir de Cas - RàPC'04. p.41-52. Université Paris Nord, Villetaneuse.

DESPRES S. (2002). Contribution à la conception de méthodes et d'outils pour la gestion de connaissances. Habilitation à diriger des recherches**,** Université de Paris V.

ELORRIAGA J. & FERN NDEZ-CASTRO I. (2000). Using case-based reasoning in instructional planning : towards a hybrid self-improving instructional planner. In International Journal of Artificial Intelligence in Education, p. 416–449.

FUCHS B. (1997). Représentation des connaissances pour le raisonnement à partir de cas : le système Rocade. Thèse de doctorat, Université Jean Monnet de Saint-Etienne.

FUNK P. & CONLAN O. (2002). Case-based reasoning to improve adaptability of intelligent tutoring systems. In Workshop on Case-Based Reasoning for Education and Training at 6th European Conference on Case Based Reasoning, p. 15–23.

GIBSON J.J. (1977). The theory of affordances. In R. Shaw & J. Bransford (eds.), Perceiving, Acting and Knowing. Hillsdale, NJ: Erlbaum.

GIBSON J.J. (1979). The ecological approach to visual perception. Chapter 14: The theory of information pickup and its consequences. p. 238-263. Boston: Houghton Miflin Co.

GOODENOUGH J. B. (1975). Exception handling : issues and a proposed notation. In Communications of the ACM.

JOHNSON-LAIRD P. (1977). Procedural semantics. In Cognition, p. 189–214.

KOENIG A. R. & STROUSTRUP B. (1989). Exception handling for c++. In Proceedings of the C++ at Work conference, p. 322–330.

KOLODNER J. (1993). Case-based reasoning. In Morgan Kaufmann Pub.

LEAKE D. (1996). Cbr in context : The present and the future. In Case-BasedReasoning : Experiences, Lessons, and Future Directions.

MEYER B. (1988). Disciplined exceptions. In Technical report tr-ei-22/ex, Interactive Software Engineering.

SABAH G. (1990). Caramel : A computational model of natural language understanding using a parallel implementation. In Proceedings of the Ninth ECAI, p. 563–565.

SANSONNET J.-P. (1999). Présentation de vdl 0.1. In Rapport interne, p. 99–10. LIMSI-CNRS.

SCHANK R. (1982). Dynamic memory ; a theory of reminding and learning in computers and people. In Cambridge University Press.

SCHANK R. & RIESBECK C.K. (1989). Inside Case-Based Reasoning. Lawrence Erlbaum Associates, Inc. Hillsdale, New Jersey.

SOUCHON F., VAUTTIER S., URTADO C. & DONY C. (2004). Fiabilité des applications multiagents : le système de gestion d'exceptions sage. In Actes JFSMA'04, LAVOISIER, Ed., p.121–134.

TROUSSE B., JACZYNSKI M. & KANAWATI R. (1999). Une approche fondée sur le raisonnement à partir de cas pour l'aide à la navigation dans un hypermédia, In proceedings of Hypertexte & Hypermedia : Products, Tools and Methods, Paris.

VOß (ED) 1994. *Similarity Concepts and Retreival Methods*. FABEL project Technical Report number 13. Gesselschaft für Mathematik und Datenverarbeitung (GMB). Sankt Augustin.

WATSON I. & MARIR F. (1994). Case-Based Reasoning: A Review. The Knowledge Engineering Review, p. 327-354.

WEINREB D. L. (1983). Signalling and handling conditions. In Technical report, ambridge, MA, USA.

WITTGENSTEIN L. (1953). The philosophical investigations. New York : Macmillan.

WOODS W. (1981). Procedural semantics as a theory of meaning. In E. of discourse understanding, ed., a. joshi, b. webber and i. sag.

L'expérience tracée comme support potentiel de négociation de sens entre agents informatiques et humains

Arnaud Stuber, Salima Hassas, Alain Mille

Equipe Cognition, Expérience et Agents situés - LIRIS
Université Claude Bernard Lyon 1
Bâtiment Nautibus - 8 bd Niels Bohr - 69622 Villeurbanne
http://experience.univ-lyon1.fr/
{astuber, hassas,amille}@liris.univ-lyon1.fr

Résumé : Ce travail a pour objectif de permettre, à une communauté technique et scientifique, de partager efficacement des connaissances et des expériences durant la réalisation d'une tâche collaborative. Nous présentons notre réflexion pour la construction d'un tel système informatique, détaillons un formalisme de trace d'utilisation basé sur la théorie des langages formels, et présentons notre approche pour manipuler ces traces.

Mots-clés : Raisonnement à Partir d'Expérience, trace d'utilisation, travail collectif, Systèmes Multi-Agents, émergence de langage

1 Introduction

Ce travail présente les principes d'un système d'assistance personnalisée, avec pour objectif de favoriser le partage d'expérience au sein d'une communauté de travail afin de réaliser une tâche collective. Nous considérons des acteurs ayant des compétences, des formations et des expériences hétérogènes, et tenant des rôles différents voire multiples.

Les objets échangés sont des documents, servant de comptes-rendus des diverses étapes d'une activité collective; ils peuvent rassember des données techniques et des résultats issus d'outils spécifiques. L'assistant proposé s'applique à un portail matérialisant cet environnement documentaire et permettant le partage et l'échange de documents entre les acteurs. Pour capturer l'interaction entre un acteur et le système, nous choisissons de la représenter sous la forme d'une *trace d'utilisation*.

La section suivante présente les principes d'ensemble de notre approche, en les situant par rapport à des travaux connexes. La section 3 détaille la repré-

sentation de l'environnement de partage, les traces d'utilisation, les principes de manipulation, et la suite de notre travail pour en dégager des signafications utiles aux utilisateur et manipulables par le système. Nous discutons alors de leur manipulation en vue d'une assistance. En section 4, nous présentons brièvement notre prototype en cours de développement, et son contexte de réalisation.

2 Tracer l'expérience dans un contexte hybride d'agents humains et informatiques

Cette section introduit en premier lieu les principes de notre approche. Un positionnement par rapport à des travaux connexes est ensuite effectué.

2.1 Principes

Notre approche est fondée, d'une part, sur le paradigme de raisonnement à partir d'expérience tracée, pour constituer des bases d'expériences individuelles. D'autre part, le paradigme de systèmes multi-agents est adopté pour représenter directement un collectif interagissant via un environnement informatique partagé. Plus spécifiquement, nous utilisons les mécanismes d'émergence de langage pour permettre l'émergence de sens dans les traces, à partir des interactions entre les agents.

Pour assister la réalisation collective de tâche, nous devons considérer les interactions :

- entre les acteurs pour remplir la tâche;
- entre les acteurs pour partager et enrichir leurs expériences;
- entre un acteur et le système pour garantir sa participation;
- entre un acteur et le système pour réutiliser sa propre expérience.

Pour les gérer, un agent *alter ego* est associé à chaque acteur. Il doit donc assister la réutilisation de l'expérience individuelle, permettre le partage et l'échange de l'expérience et participer à l'émergence d'une expérience collective. Les interactions sont alors regroupées en deux catégories :

- entre un acteur et son assistant;
- entre les agents humains médiées par le système, donc entre les agents alter ego.

Le système considéré est donc un système multi-agent *hybride*.

Notre objectif est de parvenir à la co-construction de sens communs aux acteurs et aux agents alter ego, pour permettre une interprétation adéquate et partagée des traces. Notre approche s'appuye sur les mécanismes d'*émergence de langage*,

qui impliquent une représentation symbolique. Cette représentation est basée sur un *niveau syntaxique formel* pour situer les traces par rapport à l'environnement de l'application. Notre modèle de trace est conforme au modèle MUSETTE (Champin & Prié, 2003).

2.2 Travaux connexes

La personnalisation de l'assistance, qui est recherchée en introduisant les agents *alter ego*, nous place dans la problématique d'*agent interface* présentée dans (Maes, 1994; Lieberman, 1997).
Les principes d'émergence de langage sont présentés dans (Steels, 2000), ils vont constituer la trame du comportement interactif des agents alter ego.

Des travaux de Plaza & al. (Martin *et al.*, 1998; Plaza *et al.*, 1997; Plaza & Ontañon, 2001) ou bien encore (Kanawati & Malek, 2002) constituent d'importants points de départ pour des cas structurés. Toutefois, dans notre cas, l'expérience de collaboration est formée de l'ensemble des expériences individuelles (matérialisées par des traces), pour lesquelles aucune orientation n'est donnée a priori quand à leur lecture, empêchant ainsi de les découper sous forme de cas. Afin de fournir une assistance à la collaboration, les agents alter ego doivent disposer des moyens pour partager et manipuler l'ensemble de ces traces.
Le niveau syntaxique de description des traces, que nous présentons en 3.2, suit une logique similaire à celui présenté dans (Gorodetski & Kotenko, 2002) pour des systèmes de détection d'intrusion (IDS). Dans le même domaine, (Martín & Plaza, 2004) présentent un mécanisme ascendant pour l'élaboration des cas, il est similaire au mécanisme d'identification de signature que nous présentons en 3.2.

3 Représentation de l'expérience

Cette section présente d'abord notre modélisation de l'environnement considéré : le lien entre l'activité collective et les documents manipulés, les caractéristiques d'ensemble de l'outil permettant de les manipuler. Ensuite, la modélisation des traces d'utilisation est détaillée, en présentant également leur représentation symbolique. A partir de cela, nous introduisons notre approche pour assurer l'émergence de sens communs entre un utilisateur et son assistant qui serviront au partage d'expérience entre utilisateurs via le système.

3.1 Modélisation de l'environnement

L'environnement est composé d'un outil informatique commun à tous les acteurs pour l'activité considérée; il est utilisé pour manipuler des documents. Ces documents servent de supports durant les diverses tâches de l'activité; ils rassemblent des éléments utiles aux acteurs, au regard de leurs rôles.

3.1.1 Documents et activités

Les documents suivent des modèles propres à l'activité, et sont composés de différents champs, qui peuvent contenir du texte libre, des références à d'autres documents, des pièces jointes, et des termes spécifiques au domaine d'activité.

Les termes sont regroupés dans un thesaurus, avec une structure spécifique au domaine d'activité; nous ne faisons pas d'hypothèse forte sur sa structuration, seule la possibilité de rapprocher sémantiquement des termes nous intéresse.

Dans une première étude[1], nous avons retenu de décrire des processus coopératifs de la manière suivante : les activités observées peuvent être explicitées à l'aide des concepts d'*étape coopérative élémentaire*, de *cycle* et de *processus coopératif* ; toutefois, cette description n'est qu'indicative. Une étape coopérative élémentaire designe une tâche pouvant être menée individuellement, elle est représentée par au moins un document. Dans la pratique, ces étapes se regroupent en *cycles*, qui représentent des tentatives de résolution d'un processus coopératif considéré. Les cycles sont menés successivement ou conjointement jusqu'à l'obtention d'un résultat satisfaisant, des étapes peuvent appartenir à plusieurs cycles. Les processus peuvent également être combinés.

Nous suggérons cette démarche descriptive peu contraignante, car elle permet de contextualiser les documents les uns par rapport aux autres. D'autre part, elle constitue une structure macroscopique pour guider la construction de l'expérience de collaboration. Dans la suite, nous ne faisons pas l'hypothèse d'avoir une telle description à disposition.

3.1.2 Application cible

Les *actions* pouvant être réalisées dans l'environnement documentaire introduit ci-dessus sont : l'*édition*, la *consultation*, la *recherche* et l' *annotation*. Les annotations sont considérées comme un moyen privilégié d'échange d'information entre acteurs. Bien que faiblement structurées, donc peu manipulables par le système, leur contextualisation est utile pour les acteurs humains.

Nous ne faisons aucune hypothèse restrictive sur le fonctionnement de l'application; ainsi plusieurs actions peuvent être exécutées de front. La section 4 donne une illustration de ce type d'outil.

3.2 Modélisation des traces d'utilisation

Le modèle de trace présenté ici est fondé sur le modèle MUSETTE. Il se caractérise par une structure de trace de la forme *état-transition*. Les *objets d'intérêt* observables pour la tâche considérée sont décrits dans un *modèle d'utilisation*, ce sont des entités et des événements. Une *trace d'utilisation* décrit les changements d'états du système, observables par l'utilisateur et enregistrés dans les états. Ils sont provoqués par les opérations de l'utilisateur, qui sont représentées par des événements dans les transitions. Pour manipuler les traces enregistrées, l'assistant doit reconnaître des situations; une *signature de tâche expliquée* désigne un

1. Etude réalisée avec le soutien de la société *PCO Technologies*, http://www.pcotech.fr

motif récurrent dans les traces, qui est un événement marquant relié à au moins une situation. Lorsqu'une partie d'une trace d'utilisation vérifie une signature de tâche expliquée, elle constitue un *épisode d'utilisation*. La modélisation de l'environnement introduite en 3.1 définit le *modèle d'utilisation*.

Toutefois, prenons le cas d'un utilisateur effectuant une *action* sur un document. Pour réaliser son action, il dispose d'un *ensemble de fonctionnalités* qu'il applique au document cible, sur la base d'une démarche personnelle. Une trace d'utilisation présente donc une forte variabilité si l'on considère deux utilisateurs réalisant la même tâche selon des démarches distinctes.

On est ici en présence d'un verrou pour la manipulation des traces d'expérience, qui doit être levé afin de ne pas limiter leur rapprochement. Pour cela, les fonctionnalités de l'outil constituent un sens commun aux opérations des utilisateurs. La structuration de trace présentée ci-dessous explicite l'appel des fonctionnalités constituant une action; les opérateurs et leurs propriétés vis à vis des différents éléments de langage servent à exprimer les formes équivalentes d'une même action.

Pour les actions considérées, il est possible de décomposer l'exécution d'une action en : une *séquence d'Initialisation*, un ensemble de séquences correspondant aux fonctionnalités de cette action, et une *séquence de Cloture*. Parmi les séquence intermédiaires, certaines sont permutables entre elles, on parle alors de *séquences Alternatives*; d'autres marquent des événements qui ne peuvent pas être permutés, on parle de *séquences Statiques* (les séquences d'initialisation et de cloture sont par définition statiques). Cette description des actions et des fonctionnalités est rajoutée au *modèle d'utilisation*; lors du traçage, un lien vers l'élément approprié de cette description est placé dans chaque séquence, on parle de *marqueur*.

Ci-dessous la définition formelle des propriétés à l'aide d'un langage formel.

3.2.1 Un formalisme à base de langage formel pour interpréter les traces

- **Phrase:** une phrase est une suite d'actions. Si des actions sont menées conjointement dans plusieurs fenêtres de l'application, on applique la même approche à chaque fenêtre. On appelle ces phrases des *traces*.

- **Symboles terminaux:** les transitions (T) et les états (E) sont considérés comme des symboles terminaux. Parmi eux, T_I, T_A, T_S, et T_C sont des transitions qui contiennent respectivement les marqueurs pour la séquence d'*initialisation*, des séquences *alternatives*, des séquences *statiques* et la séquence de *cloture* d'une action;

- **Opérateurs:** considérons les opérateurs $\cdot$ et $+$ entre symboles terminaux, avec pour propriétés que $\cdot$ n'est pas commutatif, et que $\cdot$ est prioritaire sur $+$. Une succession d'états et de transitions impliquant uniquement des $\cdot$

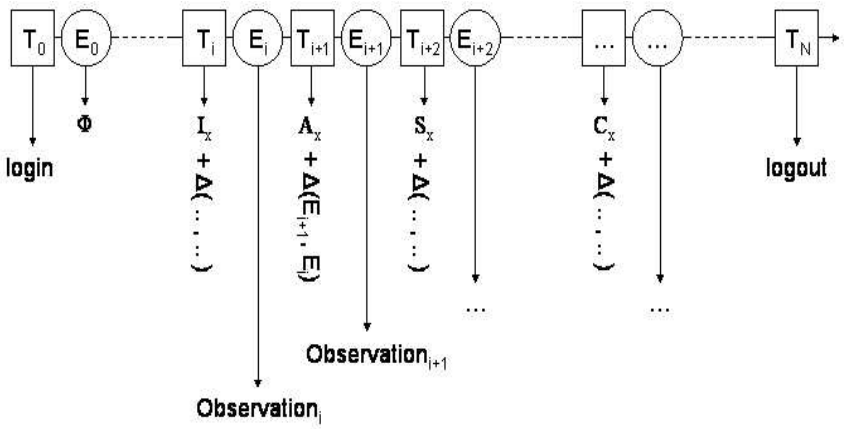


FIG. 1 – *Structure de trace : une suite d'*état/transition*, délimité par deux transitions représentant l'ouverture et la fermeture d'une session. Un* état *est une capture de l'état de l'application cible à un instant donné. Une transition contient un* marqueur *de l'opération observée, et la différence entre l'état suivant et l'état précédent. Les marqueurs sont définis dans la grammaire formelle.*

est alors indivisible, la notion de *séquence* est ainsi formellement définie. Inversement, + est utilisé entre des séquences; la commutativité entre séquences est fonction de leurs statuts, selon les principes de description des actions et des fonctionnalités selon les principes énoncés en introduction de cette grammaire.

- **Symboles non-terminaux:** une *trace* est une suite d'actions; elle se décompose en une séquence initiale (I), des séquences intermédiaires et une séquence de cloture (C). Les séquences intermédiaires sont des alternatives (A) et des séquences statiques (S); les alternatives ne sont permutables que si elles se suivent sans interruption.
- **Règles de production:**

$$\begin{aligned} Trace &::= Action^* \\ Action &::= Sequence_I + SequenceInterm^* + Sequence_C \\ SeqInterm &::= SeqPerm \mid Sequence_S \\ SeqPerm &::= Sequence_A \mid SeqPerm + Sequence_A \mid Action \\ Sequence_x &::= T_x \cdot E \cdot [T \cdot E]^* \quad avec \quad x \in \{I, A, S, C\} \end{aligned}$$

La figure 1 illustre la structure d'une trace d'utilisation. Dans une séquence, ici de taille minimale (i.e., une transition et un état), au moins une transition contient le *marqueur de fonctionnalité* et toutes contiennent une *desciption des changements* qu'elles expriment (la différence entre l'état précédent et le suivant). Les états décrivent le contexte entre les événements significatifs. Notons

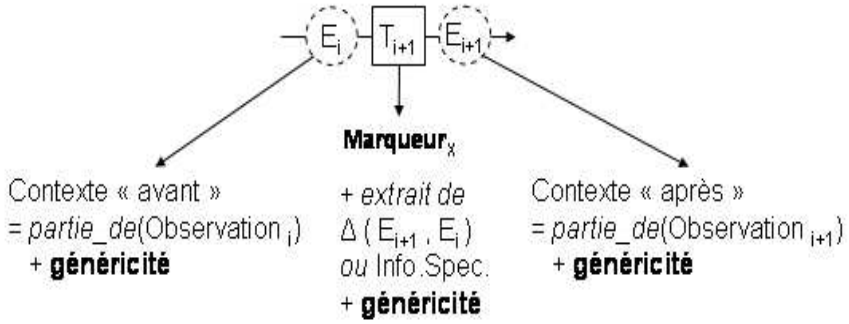


FIG. 2 – *Structure d'un symbole : elle correspond à la structure d'une* séquence *de trace (FIG. 1), avec un* marqueur *dans la première transition. Les états contiennent des observations partielles, avec une dimension* générique *permettant de comparer des éléments d'observation sur la base de leur types.*

la présence de transitions particulières (login, logout) en début et en fin de trace : elles fournissent des informations sur la trace et permettent de la délimiter.

Cependant, cette structuration des traces à l'aide d'une grammaire formelle ne permet pas de connaître, parmi les éléments de la trace, quels sont ceux qui sont à prendre en compte pour exprimer une expérience.

La section 3.3 présente un mécanisme de reformulation des traces sous une forme symbolique, pour permettre le rapprochement de épisodes d'utilisation et pour faciliter la lecture des traces par l'utilisateur. La section 3.4 présente ensuite notre approche pour parvenir à capturer le sens des actions de l'utilisateur et pour l'exprimer sous cette forme symbolique.

3.3 Représentation symbolique des traces

Les *signatures de tâches expliquées* constituent les situations connues des agents alter ego et porteuses d'un sens pour l'utilisateur. Pour exprimer les signatures, nous introduisons une couche intermédiaire, qui est formée d'un ensemble de motifs locaux, appelés *symboles*; ces symboles sont construits à partir des séquences, et donnent à l'acteur le moyen de préciser les objets qui doivent être considérés pour effectuer des rapprochements. Une signature est définie par un groupe de symboles, et par un groupe de *règles de contextualisation* contraignant et reliant ses symboles. Les signatures et les symboles vont évoluer durant les interactions, selon les mécanismes de négociation introduits en 3.4.

Les symboles ont une structure similaire à celle des séquences marquées : une suite de *transition-état* avec la possibilité d'ajouter l'état prédécesseur. La première transition contient le *marqueur* de fonctionnalité et toutes contiennent une *desciption des changements* qu'elles impliquent (la différence entre l'état précédent et le suivant). La figure 2 illustre cette structure.

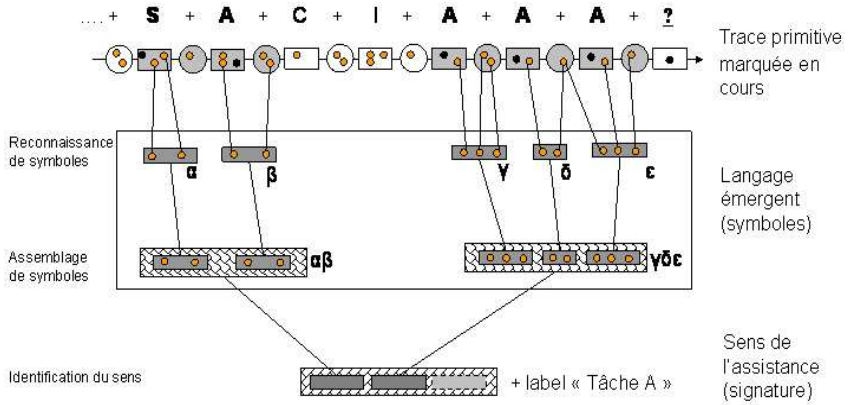


FIG. 3 – *Principes de reconnaissance ascendante de signature : les* marqueurs *de fonctionnalité guident l'identification de symbole. A chaque symbole reconnu est associé une étiquette (ici, α, β...) exprimant le sens en langue naturelle. Des signatures candidates sont cherchées, les règles de contextualisation servent à isoler celes qui sont vraissemblables, jusqu'à l'identification satisfaisante d'au moins une signature.*

Afin de jouer le rôle de grille de lecture de la trace, un symbole dispose de *noeuds génériques*, qui sont comparés avec les noeuds des traces uniquement sur la base de leurs types. Dans notre cas, les observations sont structurées en arbres, et la reconnaissance d'un symbole revient donc à rechercher un appariemment partiel entre ses arbres d'observations et leurs équivalents dans la trace.

La reconnaissance d'une signature est faite de manière ascendante à partir de la trace d'utilisation en cours. La figure 3 donne les principes de ce processus.

Après chaque opération de l'utilisateur, l'agent alter ego initie une reconnaissance de symboles, les candidats sont ceux avec le même marqueur de fonctionnalité. Un appariement est d'abord recherché entre la description des changements contenue dans un symbole et celle contenue dans la section de trace ciblée. Les symboles identifiés sont enregistrés au cours de la construction de la trace.

L'accumulation de symboles appariés et la vérification des règles de contextualisation permet la reconnaissance de signatures. Pour une signature donnée, l'épisode d'utilisation identifié dans la trace en cours est représenté par un ensemble de symboles reconnus. Un algorithme d'appariemment d'arbre est appliqué afin de retrouver des épisodes d'utilisation antérieurs, où figurent les mêmes symboles. Les épisodes retrouvés sont classés selon leur similarité conceptuelle avec l'épisode en cours, en se basant sur les informations de contexte contenues dans les états. Le résultat est alors soumis à l'utilisateur.

3.4 Emergence de sens

Les principes d'extraction d'épisode, présentés ci-dessus, ne garantissent pas l'adéquation entre le sens de l'assistance voulu par l'utilisateur et la signature qui représente ce sens pour l'agent alter ego. Néanmoins, l'agent peut profiter des interactions avec l'utilisateur pour réduire les ambiguités de sens.

Pour cela, nous proposons de voir les interactions introduites en 2 comme des processus de négociation de sens entre les acteurs de ces interactions; et de transposer les mécanismes d'émergence de langage présentés dans (Steels, 2000).

D'ors et déjà, lorsqu'un utilisateur manipule un système, il interagit avec lui en tenant compte des réactions; par exemple, lors de l'utilisation d'un moteur de recherche sur internet, nous corrigeons notre requête en fonction de l'influence apparente des mots clés. Nous avons donc déjà une démarche de négociation avec le système, mais elle reste à sens unique. Dans cette section, nous nous intéressons tout particulièrement à l'interaction entre un acteur humain et son assistant.

Pour illustrer cette approche, considérons une interface graphique permettant de manipuler des signatures, des symboles, et des traces. La figure 4 donne une illustration d'une telle interface.

Les principes de fonctionnement sont :

- le cadre supérieur donne les symboles reconnus dans la trace en cours pour une signature donnée.
- le cadre situé en-dessous donne une liste des épisodes retrouvés, qui vérifient cette signature;
- pour un épisode sélectionné, le détail de son interprétation symbolique est affiché dans le cadre situé plus bas.
- dans la partie inférieure, les arbres représentent respectivement le detail d'un symbole sélectionné pour la trace en cours et pour l'épisode antérieur. Ils permettent à l'utilisateur d'agir sur la définition d'un symbole en précisant ce qui doit être pris en compte.

Après une suite d'opérations, l'utilisateur fait appel à son agent alter ego, qui lui présente, au travers de l'interface graphique, les interprétations symboliques qu'il a réalisées, ainsi que les épisodes d'utilisation qu'il a pu rapprocher. L'utilisateur réagit (via cette interface) aux interprétations et aux rapprochements proposés. Notre approche est de voir ces interactions comme un ensemble de jeux de langage destinés à négocier le sens des signatures et des symboles, pour construire un langage symbolique commun entre l'utilisateur et son alter ego; ces éléments de langage serviront par la suite à l'échange d'expérience entre les agents ego selon des jeux de langage similaires.

Vers des jeux de langage pour la négociation du sens

Les jeux synthétisés dans (Steels, 2000), qui permettent l'émergence d'un langage, sont les jeux d'*Imitation*, de *Discrimination* et de *Nommage*.

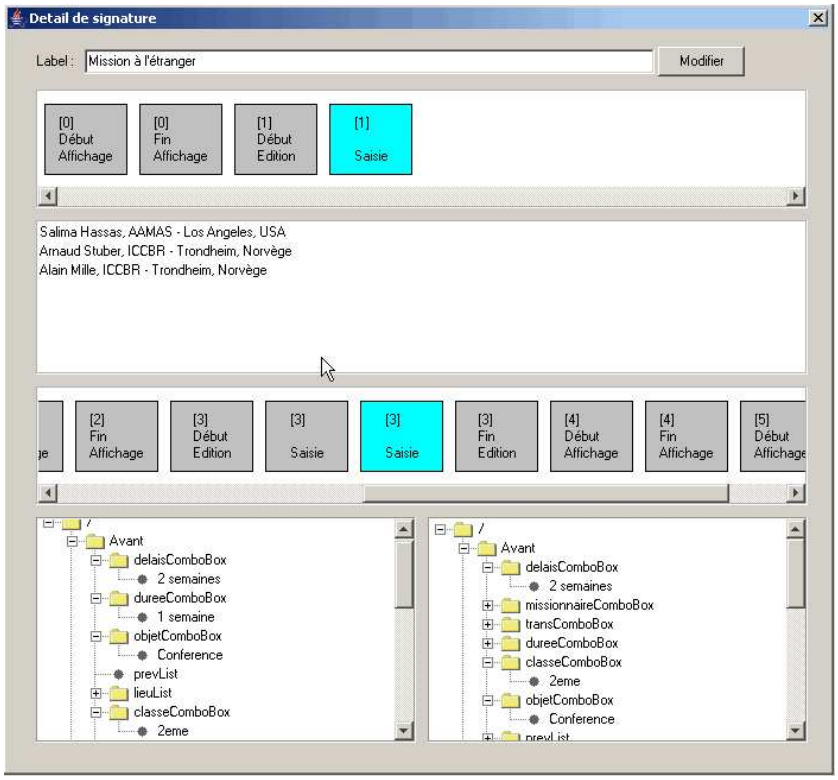


FIG. 4 – *Interface graphique de l'assistant*

Ils ont pour objectifs respectifs de co-construire des éléments de langage (des *mots*) supports à la conversation entre acteurs, de construire individuellement des prédicteurs permettant de sélectionner un sens parmi d'autres à partir d'une observation locale, et enfin de co-construire des lexiques entre les mots et les sens. L'auto-adaptation aux limites de perception et d'action des différents acteurs est une conséquence directe de ces jeux; de même un couplage structurel est observé entre les mots émergents, les sens identifiés et les lexiques permettant une communication sémantique à l'aide de ces mots. Ces principes sont appliqués aux deux types d'interaction introduits en 2.

Sur figure 3, les étiquettes -en langue naturelle- des symbole et des signatures forment deux répertoires de sens. La structure interne d'un symbole (ses arbres génériques) ainsi que la composition d'une signature (ensemble de symboles) constituent chacun à leur niveau des mécanismes de catégorisation. Les liens entre un sens -signature ou symbole- et sa description détaillée -respectivement, des symboles ou des arbres génériques- constituent les éléments de lexique pour

chaque niveau.
A partir de cela, nous considérons que la pertinence d'ensemble des interprétations et des rapprochements effectués peut être vue comme un jeu d'imitation; l'alter ego cherche à interpréter les actions de l'utilisateur et à donner des épisodes antérieures qui *imitent* au mieux la situation en cours. Le jeu de discrimination est porté par les structures de signature ou de symbole manipulées : il en résulte des performances du système, et est guidé par l'utilisateur. Le jeu de nommage est implicite aux symboles et aux signatures, lors de leur créations respectives. Les interactions capturées initient des itérations de jeux, dont certaines vont pouvoir être menées complètement et ainsi faire évoluer le langage.

4 Application

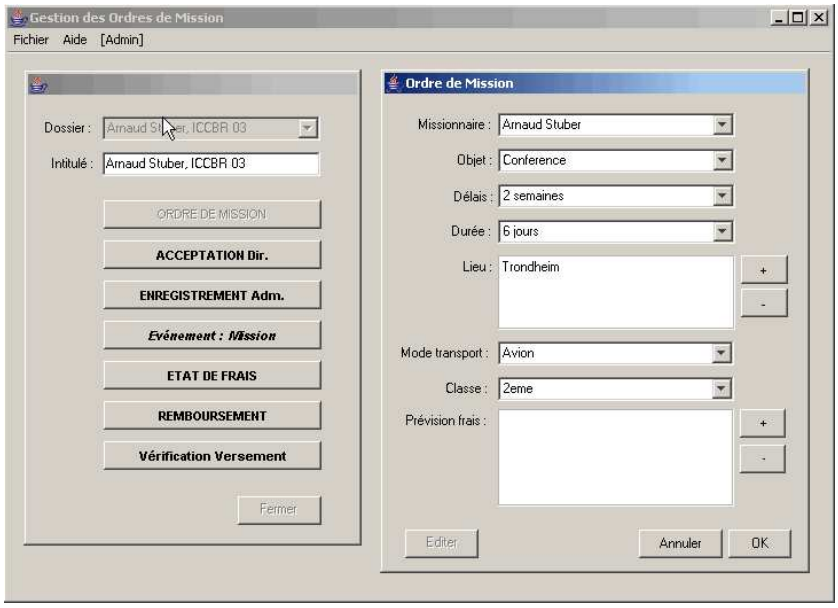


FIG. 5 – *Application prototype : portail de gestion des dossiers administratifs pour la prise en charge des déplacements.*

Pour nos expérimentations, nous considérons l'activité coopérative d'*enregistrement de mission* présente au LIRIS. Cette activité est comparable à toute procédure administrative impliquant divers acteurs aux expériences et compétences limitées. Nous développons un environnement informatique en *Java*, auquel appartient l'assistant représenté sur la figure 4. Cet environnement permet à la communauté de gérer les différents formulaires. Les description symboliques sont faites en RDF/XML. La figure 5 illustre ce portail, avec à gauche une fe-

nêtre représentant les étapes de la procédure, et à droite une fenêtre avec le détail d'un document.

5 Conclusion et Perspectives

Dans cet article, nous présentons la problématique de notre étude, qui est de permettre à une communauté scientifique et technique le partage efficace de connaissances et d'expériences lors de la réalisation d'une tâche collective complexe. Nous cherchons à mettre l'expérience collective à disposition des utilisateurs, par le biais d'un assistant contextuel personnalisé. L'expérience collective est la combinaison des expériences individuelles acquises à partir des traces d'utilisation laissées par les acteurs dans le système. La modélisation de l'environnement de partage et le formalisme de représentation de trace sont présentés.

Nous présentons alors la suite de notre travail, qui se concentre sur la nécessaire capture du sens de l'interaction entre un utilisateur et son assistant, pour permettre à ce dernier de s'adapter et ainsi d'augmenter ses performances.

Références

Champin P.-A. & Prié Y. (2003). *Musette: uses-based annotation for the Semantic Web*, In O. P. S. Handschuh (ed.), Ed., *Annotation for the Semantic Web*.

Gorodetski V. & Kotenko I. (2002). Attacks against computer network: Formal grammar-based framework and simulation tool. In *Proceedings of the 5th International Conference "Recent Advances in Intrusion Detection"*, p. 219–238, Zürich, Switzerland: Springer Verlag.

Kanawati R. & Malek M. (2002). A multi-agent system for collaborative bookmarking. In *Proceedings of the first international joint conference on Autonomous agents and multiagent systems*, p. 1137–1138: ACM Press.

Lieberman H. (1997). Autonomous interface agents. *ACM Conference on Human-Computer Interface [CHI-97], Atlanta, March 1997*.

Maes P. (1994). Agents that reduce work and information overload. *Communications of the ACM*, **37**, 30–40, 146.

Martín F. J. & Plaza E. (2004). Ceaseless case-based reasoning. In P. Funk & P. A. González-Calero, Eds., *ECCBR*, volume 3155 of *LNCS*, p. 287–301.

Martin F. J., Plaza E. & Arcos J. L. (1998). Knowledge and experience reuse through communication among competent (peer) agents. *International Journal of Software Engineering and Knowledge Engineering*.

Plaza E., Arcos J. & Martín F. (1997). Cooperative case-based reasoning. In *Lecture Notes in Artificial Intelligence*, p. 180–201.

Plaza E. & Ontañon S. (2001). Ensemble case-based reasoning: Collaboration policies for multiagent cooperative cbr. In *In Case-Based Reasoning Research and Development: ICCBR 2001 - Lecture Notes in Artificial Intelligence 2080*, p. 437–451.

Steels L. (2000). Language as a complex adaptative system. *Lecture Notes in Computer Science. Parallel Problem Solving from Nature - PPSN*, **4**.

Agent RàPC pour la Gestion Coopérative de Bases Bibliographiques Personnelles

Hager Karoui

LIPN, CNRS UMR 7030, Université Paris XIII
99, avenue Jean-Baptiste Clément
F-93430 Villetaneuse, FRANCE
hager.karoui@lipn.univ-paris13.fr

Résumé. Cet article décrit un système égal à égal (Peer-to-Peer) pour la gestion collaborative de bases de données bibliographiques distribuées. Le but de ce système est double : d'une part, il vise à fournir de l'aide pour les utilisateurs afin de gérer leurs bases bibliographiques locales ; d'autre part, il offre la possibilité d'échanger des données bibliographiques entre des groupes d'utilisateurs partageant les mêmes centres d'intérêts d'une manière implicite et intelligente. Chaque utilisateur est assisté par un agent personnel lui fournissant de l'aide telle que : remplir des enregistrements bibliographiques, vérifier la correction des informations introduites et plus encore, la recommandation des références bibliographiques pertinentes. Pour réaliser cela, l'agent personnel a besoin de collaborer avec ses pairs pour obtenir des recommandations intéressantes. Chaque agent applique une approche de raisonnement à partir de cas pour prévoir les pairs susceptibles de fournir les recommandations demandées. Cet article donne une vue générale sur notre système et se concentre principalement sur la description de l'approche de calcul des recommandations.

Mots clés. Système égal à égal, Système de recommandation, Raisonnement à partir de cas, Partage de données bibliographiques, Coopération d'agents.

1 Introduction

Maintenir une base bibliographique annotée et à jour est une activité importante dans la vie des équipes de recherche. Cependant, la multiplication des sources de documents (ateliers, conférences, journaux, …) aussi bien que la disponibilité en ligne de la majorité des documents, ont contribué à rendre la tâche de recherche d'information plus compliquée et plus coûteuse en temps. Actuellement, en sus du problème classique de la surcharge d'information, les chercheurs ont un accès direct aux articles qui sont rarement pourvus de leurs données bibliographiques complètes. Il est devenu fréquent de commencer une nouvelle session de recherche pour trouver

la référence exacte d'un article intéressant obtenu précédemment. Pour une équipe de recherche, il est fortement possible qu'une information obtenue ou connue d'un ou plusieurs chercheurs puisse être utile à un autre. De plus, il se peut que des collègues puissent avoir des traces utiles à propos de la qualité des papiers et aussi des références sur un thème donné. Il est évident que le partage des connaissances bibliographiques peut non seulement enrichir la connaissance de chaque membre, mais aussi réduire le temps et l'effort nécessaire pour gérer les bases bibliographiques personnelles.

Dans cet article, nous proposons un système multi-agents de type égal à égal pour la gestion collaborative de bases bibliographiques. Chaque utilisateur est assisté par un agent assistant logiciel personnel. Un agent assistant observe les interactions de l'utilisateur avec sa base locale. Il détecte les thèmes d'intérêt courants de l'utilisateur et trouve les informations manquantes dans la base de données locale (exemple : le lieu d'une conférence citée, le nombre de pages d'un papier donné, etc.). Les agents communiquent entre eux afin d'obtenir des informations manquantes mais aussi pour recommander à leurs utilisateurs des références associées à leurs thèmes courants. Les agents peuvent aussi vérifier la correction des références locales en les comparant avec les enregistrements de leurs pairs.

Une description détaillée de l'architecture du système et des services est donnée dans la section 2. Le thème central de ce papier est la description de la technique du calcul de la recommandation. Cela est fait en appliquant une approche de raisonnement à partir de cas (RàPC). La méthodologie de RàPC a été utilisée avec succès par plusieurs systèmes de recommandations (Burcke, 2004). Le cycle de raisonnement appliqué est détaillé dans la section 3. Les travaux similaires sont décrits en section 4 et finalement, nous concluons et nous donnons quelques directions pour le travail futur en section 5.

2 Description du système

La Figure 1 illustre l'architecture globale du système où chaque utilisateur maintient localement une base de données bibliographique personnelle. L'utilisateur gère sa base de données locale en utilisant un module de gestion qui fournit les fonctions de gestion classiques telles que : l'ajout, la suppression, l'édition et la recherche de références. De plus, il fournit des fonctionnalités pour la sélection d'une liste de références à ajouter à un travail en cours (par exemple construire une bibliographie d'un rapport) et exporter des listes en différents formats (BibTeX, html, pdf, etc,). Dans le but de faciliter les opérations d'exportation/transformation, les références sont stockées dans un format XML. Chaque référence *r* est décrite par un enregistrement contenant les informations suivantes :

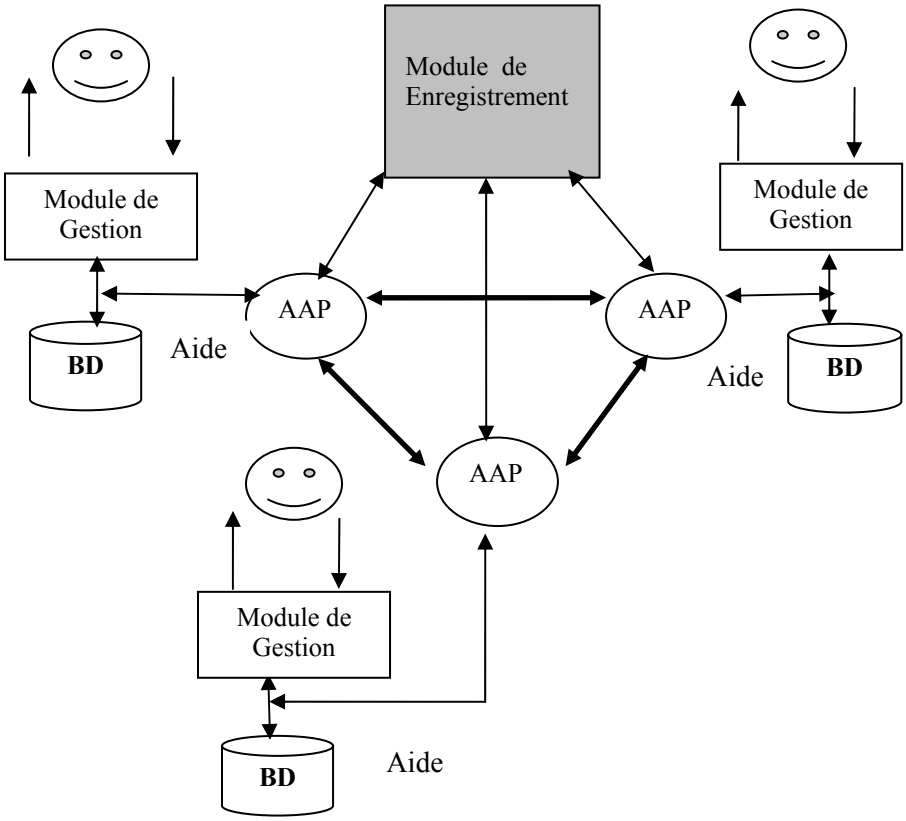


Fig. 1– Architecture du système

- Données bibliographiques (notées *r.biblio*) : ce sont des données classiques composant une référence bibliographique telles que le type (Article, In Proceedings, Report, etc.), la liste des auteurs, le titre, etc.
- Mots clés (notés *r.keywords*) : une liste de mots clés donnée par l'utilisateur décrivant la référence.
- Thèmes (notés *r.topics*) : une liste de thèmes auxquels la référence est liée. La même hiérarchie de thèmes est utilisée par tous les utilisateurs. Dans ce travail, nous utilisons une hiérarchie de thèmes en arbre. Il est raisonnable de supposer qu'une même équipe de recherche utilise les mêmes arbres de thèmes pour indexer les références bibliographiques (par exemple l'utilisation de la hiérarchie de thèmes ACM dans des équipes de recherche d'informatique). Cependant, nous soulignons que la même hiérarchie peut être utilisée différemment par différents utilisateurs. Par exemple, quelqu'un peut indexer tous les articles reliés au RàPC au thème : « RàPC », alors qu'un autre utilisateur peut indexer les mêmes articles différemment : quelques

uns reliés à « l'organisation de mémoire dans le RàPC » et les autres dans « la maintenance de cas dans RàPC». Un troisième peut indexer les mêmes références comme toutes reliées à « l'apprentissage ».
- Evaluation (notée *r.eval*) : une évaluation faite par chaque utilisateur indiquant la qualité de la référence selon son propre point de vue. L'évaluation est déterminée suivant une échelle allant de "très intéressant" à "très mauvais".
- Annotation (notée *r.annotation*) : un champ de texte libre.

Chaque agent personnel (PAA : Personal Assistant Agent) offre les services suivants :

- Edition de références : lorsqu'un utilisateur édite une référence, l'assistant local peut l'aider en remplissant certains champs. Par exemple, quand un utilisateur tape un acronyme d'un nom de conférence, l'assistant lui fournit le nom complet et peut remplir d'autres informations reliées telles que la date de la conférence, le lieu.
- Correction des références : ce service se déclenche quand des références sont ajoutées ou éditées dans la base bibliographique. Une fois que des champs de données tels que le titre d'une conférence, nom d'auteur, … concernant une référence sont remplis et validés, le système vérifie la correction des entrées en lançant l'agent correspondant. Une première vérification de la correction des données est faite en examinant la base locale de l'utilisateur. Si les données sont écrites correctement alors la tâche est confirmée. Autrement, si des erreurs sont détectées, l'agent propose une correction. S'il y a quelques champs vides, l'agent suggère les données manquantes s'il en dispose. Si les données ne sont pas trouvées dans la base locale, l'agent collabore alors avec ses homologues pour les trouver.
- Recommandation : ce service tend à partager des connaissances bibliographiques entre les utilisateurs.

Le module d'enregistrement détient des informations sur les différentes connexions /déconnexions des agents et d'autres informations systèmes.

Le but consiste à tirer parti des expériences passées d'un seul utilisateur ou même d'un groupe d'utilisateurs pour recommander des références plus pertinentes. Une approche de Ràpc est utilisée pour calculer les différentes recommandations d'une manière coopérative, i.e. les différents agents assistants doivent collaborer pour obtenir des recommandations variées et pertinentes. L'assistant local doit suggérer des recommandations intéressantes et variées à son utilisateur selon son activité courante. L'utilisateur pourra alors choisir d'accepter ou de refuser les recommandations proposées.

3 Calcul des recommandations

3.1 Description informelle

Chaque agent assistant applique le cycle de calcul suivant :

- Premièrement, il calcule la liste des thèmes courants qui intéressent le plus l'utilisateur (cf. section 3.2).
- Pour chaque thème courant, l'agent envoie une requête de recommandation à un ensemble d'agents pairs susceptibles de posséder des références reliées à ce thème. Un message de demande de recommandation contient un identificateur de thème et une liste de mots clés décrivant l'ensemble de références enregistrées sous ce thème dans la base de données locale (cf. section 3.3).
- L'agent peut recevoir de chaque agent contacté, deux listes de références : des références recommandées et des références non recommandées. La dernière liste est formée des références qui satisfont la requête mais qui sont mal évaluées par l'utilisateur associé.
- L'agent assistant fusionne et filtre les résultats reçus en supprimant les références dupliquées ainsi que celles déjà stockées dans la base de données locale (donc déjà connues de l'utilisateur).
- Les références les mieux classées (celles ayant les plus hautes valeurs de similarité) seront ainsi recommandées à l'utilisateur. Ce dernier peut accepter d'ajouter toutes ces références ou une partie au thème indiqué ou à d'autres thèmes.
- Les utilisateurs peuvent aussi rejeter toutes les références fournies ou quelques unes.

Lorsqu'il reçoit une demande de recommandation, un agent cherche dans la base de données locale des références correspondant à la demande reçue.
Le processus de recherche commence à partir du thème indiqué dans la demande. Rappelons que tous les utilisateurs partagent la même hiérarchie de thèmes. Quand il y a un nombre insuffisant de références, l'agent continue la recherche dans les sous-thèmes suivant une stratégie de recherche en profondeur d'abord. S'il n'y a pas assez de références, l'agent continue la recherche dans les super-thèmes. La stratégie consiste à préférer les références reliées aux sous-thèmes supposées être plus spécifiques que celles reliées aux super-thèmes supposées être plus générales. Le processus de la recherche se termine quand suffisamment de références ont été trouvées ou si la similarité entre le thème de la demande et celui recherché tombe au dessous un certain seuil. La pertinence d'une référence est calculée à l'aide d'une mesure de similarité qui prend en compte la similarité entre les thèmes (du message de la demande de recommandation et celui recherché) aussi bien que la correspondance entre la description des mots clés de la référence et la liste de mots de clés de la demande.

3.2 Calcul des thèmes d'intérêt courants

Comme mentionné précédemment, tous les utilisateurs partagent une même hiérarchie de thèmes ayant une structure d'arbre. Tous les thèmes ne présentent pas le même intérêt pour chaque utilisateur. De plus, pour chaque utilisateur, l'ensemble de thèmes intéressants change au cours du temps. Les recommandations fournies ne doivent pas être intrusives : les références à suggérer doivent être pertinentes et doivent se présenter à l'utilisateur au bon moment. De plus, si les utilisateurs sont réellement intéressés par le domaine de recommandation, ils seront plus enclins à évaluer ces

recommandations. Dans le but de calculer la liste des thèmes courants d'un utilisateur, nous appliquons un algorithme simple qui mesure *la température* de chaque thème. Les thèmes ayant une température au dessus d'un certain seuil σ seront considérés comme des "thèmes d'intérêts courants". Initialement, tous les thèmes ont une température égale à zéro. Chaque action exécutée par l'utilisateur invoquant un thème *t* va augmenter la température alors d'une quantité spécifique. Les actions typiques qui modifient la température sont : l'ajout, l'édition ou la recherche d'une référence associée à un thème. Différentes actions ajoutent différentes valeurs à la température courante du thème. Une "fonction de refroidissement" est aussi appliquée afin de diminuer la température des thèmes non utilisés par l'utilisateur. Puisque la hiérarchie des thèmes peut être utilisée différemment par plusieurs utilisateurs, nous avons besoin de déterminer les thèmes les plus spécifiques à l'activité de l'utilisateur. Pour ce faire, une fonction de propagation de température est appliquée. A partir des feuilles de l'arbre des thèmes, chaque thème propage sa température courante au thème parent. Les thèmes ayant une température au dessus d'un certain seuil fixé σ_r cessent de propager leur température et seront ajoutées à la liste des "thèmes d'intérêt courants". Les *n* thèmes ayant la température la plus élevée qui ont été ajoutés aux listes des "thèmes d'intérêt courants", après le parcours de l'arbre d'une façon ascendante, seront retournés. L'heuristique consiste à retourner les thèmes les plus spécifiques qui révèlent un certain niveau des intérêts de l'utilisateur.

3.3 Livraison des recommandations

Un agent réclame à ses pairs des recommandations en leur adressant un message de demande de recommandation. C'est un triplet $R = <A, T, L>$ où A est l'identificateur de l'agent émetteur, *T* est le thème cible et *L* est la liste des mots clés calculée à partir de l'ensemble des listes de mots clés décrivant les références associées directement ou indirectement au thème *T*. Une référence est indirectement reliée à un thème *T* si elle est reliée (directement ou indirectement) à un thème *T'* plus spécifique que *T*.

Une problématique importante dans n'importe quel système égal à égal concerne la formation du comité de pairs. L'idée consiste à doter chaque agent d'une capacité de sélection de sous ensemble des pairs disponibles pouvant fournir les résultats les plus pertinents (dans notre exemple, les recommandations). L'objectif est d'améliorer les performances de tout le système en réduisant la charge du réseau et des agents. Il faut également améliorer la qualité des recommandations fournies en évitant de fausser les résultats par des informations bruitées. Différentes approches sont proposées dans la littérature. Certaines s'appuient sur la notion de réputation d'agent (Ammar, Gupta et Judge, 2003). D'autres appliquent des techniques d'apprentissage automatique pour permettre à chaque agent de déterminer s'il a besoin d'élargir son comité de pairs et si c'est le cas, quel agent inviter (Ontanon et Plaza, 2003). Bien que cette problématique soit importante, nous avons décidé d'utiliser dans un premier temps une approche naïve qui consiste à diffuser les requêtes de recommandations à tous les pairs disponibles. Nous supposons que tous les agents ont le même degré de confiance et appliquent le même processus de calcul de recommandation.

Quand il reçoit une requête, un agent commence la recherche dans sa base locale pour des références qui correspondent au couple (*T,L*). Informellement, la liste de mots clés contenue dans la demande pourra être traitée comme une requête. Le thème cible *T* indique le point de départ de la recherche des documents dans la base de données locale. L'agent recherchera dans sa base locale les références qui correspondent à la requête reçue. L'adéquation référence/requête est évaluée par une simple fonction de similarité $Sim_{Ref}(R,r)$ qui mesure la similarité entre une demande R et une référence r. Une référence r satisfait une demande R si leur similarité est au dessus d'un seuil donné spécifié σ_r. La fonction de similarité est une agrégation pondérée de deux fonctions basiques de similarité : la similarité des thèmes (Sim_{Topics}) et la similarité des mots-clés ($Sim_{Keywords}$). Formellement, nous avons :

$$Sim_{Ref}(R,r) = \alpha\, Sim_{Topics}(R.T, r.topics) + \beta\, Sim_{Keywords}(R.L, r.keywords)\ .$$

où α et β sont les poids des similarités de base. Bien évidemment, nous avons $\alpha + \beta = 1$. La fonction de similarité de mots clés utilisée est une fonction simple qui mesure le nombre de mots en commun entre deux listes. Formellement:

$$Sim_{Keywords}(A, B) = |(A \cap B)| / |(A \cup B)|$$

La mesure de similarité des thèmes utilise la structure hiérarchique fondamentale des thèmes. L'heuristique appliquée est la suivante : la similarité entre deux thèmes dépend de la longueur du chemin qui lie les deux thèmes et de la profondeur des thèmes dans la hiérarchie (Jaczynski et Trousse, 1998), (Jaczynski, 1998). Rappelons que dans un arbre, il existe seulement un chemin entre deux nœuds. De plus, une correspondance avec des nœuds spécifiques plus proches des nœuds feuilles conduit à une valeur plus importante de similarité que les nœuds correspondant à des niveaux plus hauts de l'arbre. Nous avons :

$$Sim_{Topics}(T_1, T_2) = 1 - (path(T_1, MSCA(T_1, T_2)) + path(T_2, MSCA(T_1, T_2))) / (path(T_1, root) + path(T_2, root))\ .$$

où path(a,b) retourne la longueur du chemin entre les nœuds a et b, root est le thème racine de l'arbre et MSCA(a, b) retourne l'ancêtre commun le plus spécifique des nœuds a et b dans l'arbre des thèmes.

La Figure 2 montre un exemple simple d'un arbre de thèmes où la racine est « *Computing Methodologies* ». Nous considérons ici quatre niveaux de thèmes. Chaque thème possède une liste de références associées.

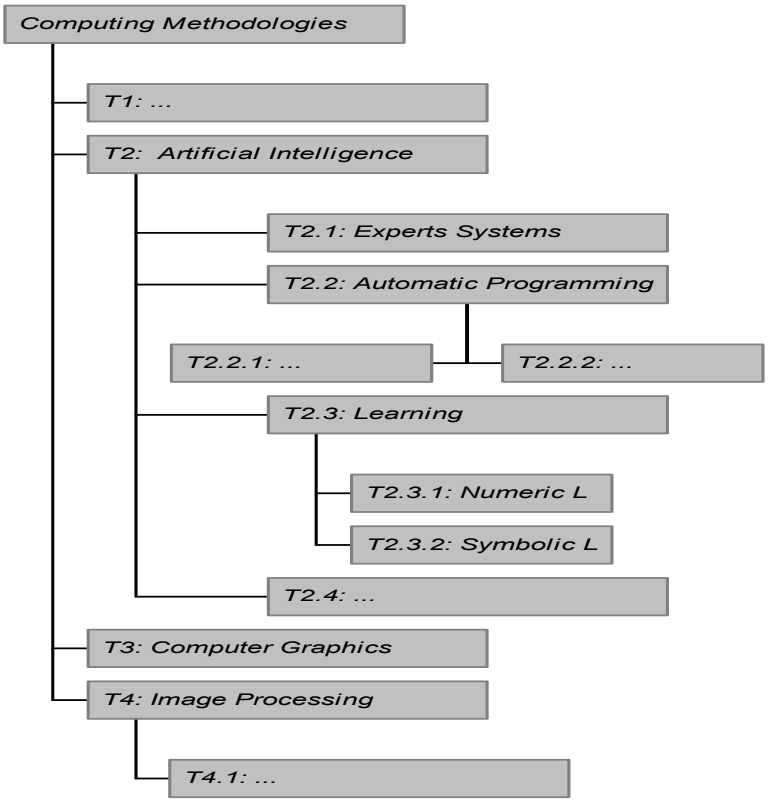


Fig. 2 – Exemple d'arbre de thèmes

En utilisant l'arbre de thèmes de la Figure 2, nous donnons quelques exemples de calcul de similarité de thèmes :

$$Sim_{Topics}(T_{2.3.1}, T_{2.3.2}) = 1-((1+1) / (3+3)) = 2/3$$
$$Sim_{Topics}(T_{2.3.1}, T_{2.2}) = 1-((2+1) / (3+2)) = 2/5$$
$$Sim_{Topics}(T_{2.2}, T_{2.3}) = 1-((1+1) / (2+2)) = 1/2$$

Les valeurs de similarité des thèmes sont calculées en appliquant la fonction Sim_{Topics} . Nous notons que la valeur de Sim_{Topics} ($T_{2.3.1}$, $T_{2.3.2}$) est plus élevée que celle de Sim_{Topics} ($T_{2.2}$, $T_{2.3}$) car ils sont plus proches des nœuds feuilles. Les thèmes ($T_{2.3.1}$, $T_{2.3.2}$) sont plus similaires que ($T_{2.3.1}$, $T_{2.2}$) car le chemin reliant $T_{2.3.1}$ et $T_{2.3.2}$, est plus court que celui reliant $T_{2.3.1}$ et $T_{2.2}$.

Utilisant ces fonctions de similarité, l'agent local essaye de retourner les *m* références les plus pertinentes qui correspondent à la requête reçue. A partir du thème cible (*R.T*), l'agent cherche les références reliées qui sont similaires au dessus d'un certain seuil σ_r. S'il n'y a pas assez de références trouvées, il examine les références reliées à des thèmes plus spécifiques, puis il cherche dans des thèmes plus généraux. Le processus de recherche se termine quand *m* références pertinentes ont été trouvées ou quand il n' y a plus de thème à visiter.

Les références pertinentes qui sont évaluées positivement par l'utilisateur local, forment la liste des références recommandées tandis que celles évaluées négativement forment la liste des références non recommandées. Chaque référence retournée est associée avec un score représentant ses similarités avec la demande initiale. L'agent ayant envoyé la demande de recommandation, recevra un couple de références recommandées et des références non recommandées de chaque agent. Il fusionne et filtre les listes reçues, il élimine d'abord les références dupliquées et celles déjà connues par l'utilisateur (celles déjà stockées dans la base de données locale). Ensuite, il applique une méthode de tri pour classer les références recommandées et les références non recommandées suivant leurs valeurs de similarité. Dans une étape suivante, nous tiendrons compte d'une autre valeur représentant l'importance de l'agent émetteur. Finalement, les *k* références recommandées et les mieux classées seront proposées à l'utilisateur.

4 Travaux similaires

Un travail intéressant directement lié à notre approche est Bibster system (Broekstra et al., 2004). Le but de ce système est de permettre à un groupe de personnes de rechercher des références bibliographiques dans les bases de données personnelles d'autres utilisateurs. Une architecture égal à égal a été utilisée. Cependant, seulement la recherche est supportée par le système. Notre système va plus loin que le système Bibster car nous utilisons une recherche de références basée sur une mesure de similarité, qui tient en compte la similarité des thèmes et la similarité des mots clés. De plus, la recherche est effectuée par des agents logiciels (les agents assistants personnels) au lieu d'être lancée par les utilisateurs eux-mêmes. Ainsi, l'utilisateur reçoit des recommandations (dans notre exemple, des références pertinentes) variées de son agent. Ce dernier effectue sa recherche implicitement en se basant sur l'observation du comportement de son utilisateur.

Les systèmes de signets collaboratifs adressent un problème similaire (Malek et Kanawati, 2000). Cependant, les systèmes de signets collaboratifs égal à égal (Malek et Kanawati, 2001) ne possèdent pas de hiérarchie unifiée de thèmes, ce qui complique le calcul de l'évaluation. Un autre travail similaire est le moteur de

recherche d'information I-SPY (Smyth et Balfe, 2004). Son but est de permettre à un groupe de personnes de même centre d'intérêt, de partager leurs résultats de recherche d'une façon implicite. Le système est construit d'une manière centralisée où une matrice enregistre pour chaque requête soumise, les documents sélectionnés par l'utilisateur. Quand une nouvelle requête est soumise, les résultats ayant été sélectionnés par d'autres utilisateurs en réponse à des requêtes passées similaires sont fournis par le système. Dans notre système, l'ensemble des thèmes peut être vu comme un ensemble de requêtes pré-spécifiées. Le chevauchement entre les requêtes utilisateurs est plus probable de se produire que dans le cas des requêtes en vocabulaire ouvert.

Dans (McGinty et Smyth, 2001), McGinty et Smyth décrivent une architecture de raisonnement à partir de cas collaboratif CCBR, qui permet aux expériences de résolution de problème d'être partagées entre plusieurs agents en échangeant des cas. Cette approche a été appliquée dans la planification de route personnalisée et promet une solution en permettant à un agent d'utilisateur donné d'emprunter des cas d'autres agents similaires et qui sont familiers avec le territoire cible. Il y a un compromis entre la similarité des agents et la couverture de leurs bases de cas. Un agent distant est utile à un agent cible s'il a une couverture différente, mais pour être considéré similaire, ils doivent partager un ensemble de problèmes communs. Plaza et. al. proposent dans (Plaza, Arcos et Martin, 1996) différents modes possibles de coopération à travers des agents homogènes avec des capacités d'apprentissage. Ils présentent deux modes de coopération entre les agents : RàPC distribué (DistCBR) et RàPC collectif (ColCBR). Dans la coopération de RàPC distribué, un agent d'origine délègue l'autorité à un autre agent pair pour résoudre le problème. Par contre, le RàPC collectif maintient l'autorité de l'agent d'origine puisqu'il décide quelle méthode de RàPC appliquer et utilise simplement l'expérience accumulée par les autres agents pairs. Ils montrent que le résultat de la coopération est souvent meilleur que lorsqu'il n'y a pas de coopération du tout. Cependant, ces protocoles sont dépendants du domaine et sont le résultat d'un processus de modélisation de connaissances. Dans (Plaza et Ontanon, 2003), Plaza et Ontanon présentent diverses stratégies de collaboration pour les agents qui apprennent en utilisant le RàPC. Les agents utilisent un mécanisme de marché (troc) pour améliorer leurs performances individuelles aussi bien que celles de tout le système multi-agents. Deux politiques ont été présentées : Politique de comité et Politique de conseil restreint. Dans la première politique de collaboration, les agents membres du système multi-agents sont vus comme un comité. Un agent ayant à résoudre un problème, l'envoie à tous les agents du comité. La solution finale est la classe avec le nombre maximum de votes. La politique suivante est celle du conseil restreint où un agent essaie de résoudre le problème lui-même et s'il échoue à trouver une "bonne" solution, il peut alors demander conseil aux autres agents dans le système multi-agents. Ils concluent que la politique du comité est meilleure que celle isolée ou celle du conseil restreint, cependant, cette précision a un coût élevé puisqu'un problème est résolu par chaque agent. A cause du biais des exemples d'un agent, qui entraîne la diminution de la précision du système, Plaza et Ontanon proposent une stratégie de collaboration basée sur l'échange de cas pour améliorer les performances du système multi-agents en diminuant les biais individuels des bases de cas. Le mécanisme d'échange de cas est basé sur un accord d'échange « bartering agreement » où le résultat de l'échange diminue le biais individuel de la base des cas des agents.

5 Conclusion

Cet article décrit un travail en cours visant à doter un groupe de personnes ayant les mêmes centres d'intérêts, d'un outil intelligent et implicite pour partager leurs connaissances bibliographiques et exploiter leurs expériences passées. Actuellement, nous travaillons sur l'implémentation d'une première version de ce système. L'approche du RàPC décrite dans cet article est limitée à l'utilisation de la recherche basée sur la similarité, qui conduira à la définition d'une approche de RàPC plus profonde, permettant aux agents d'activer leurs temps de réponse en exploitant les similarités entre les demandes de recommandations courantes et passées. Un autre problème important à traiter pour la coopération des agents, est celui de la formation du comité : comment apprendre à former un comité optimal pour chaque demande de recommandation? Enfin, nous appliquerons des schémas de RàPC collaboratifs pour améliorer la qualité des recommandations individuelles de chaque agent et profiter des expériences des autres.

Références

Ammar M., Gupta M. & Judge P. (2003). A reputation system for peer-to-peer networks. In 13 international workshop of networks and operating system support for digital and video, Montreal.

Broekstra J., Ehrig M., Haase P., Harmelen F., Menken M., Mika P., Schnizler B. & Siebes R. (2004). Bibster -a semantics-based bibliographic peer-to-peer system. In Proceedings of SemPGRID'04, 2nd Workshop on Semantics in Peer-to-Peer and Grid Computing, pages 3-22, New York, USA.

Burcke R. (2004). Hybrid recommender systems with case-based reasoning. In Advances in Case-based Reasoning, 7th European Conference, ECCBR 2004, LNAI 3155, pages 91-105.

M. Jaczynski. Modèle et plate-forme à objets pour l'indexation par situations comportementales : application à l'assistance à la navigation sur le Web. Thèse de doctorat. Université de Nice-Sophia Antipolis, 1998

Jaczynski M. & Trousse B. (1998). WWW Assisted Browsing by Reusing Past Navigations of a Group of Users. In Proceedings of the 4th European Workshop on Case-Based Reasoning, Lecture Notes in Artificial Intelligence, Dublin , september 23-25.

Malek M. & Kanawati R.(2001). Cowing: A collaborative bookmark management system. In Proceedings of CIA'02: International workshop on cooperative information agents, LNAI 2182, pages 34-39.

Malek M. & Kanawati R. (2000). Informing the design of shared bookmark systems. In Proceedings of RIAO'2000: Content-based Multimedia Information Access, 28 April 2000, Paris.

McGinty L. & Smyth B. (2001). Collaborative case-based reasoning: Applications in personalised route planing. In ICCBR, pages 362-376.

Ontanon S. & Plaza E. (2003). Learning to form dynamic committees. In Proceedings of the second international joint conference on Autonomous agents and multiagent systems, pages 504-511, Melbourne, Australia. ACM Press, NEW YORK, USA.

Plaza E., Arcos J. L. & Martin F. (1996). Cooperation modes among case-based reasoning agents, Proceedings of the ECAI'96 Workshop on Learning in Distributed AI Systems. Budapest, p. 90-99.

Plaza E. & Ontanon S. (2003). Cooperative Multiagent Learning, Lecture Notes on Artificial Intelligence 2636, p. 1-17. Springer Verlag, London.

Smyth B. & Balfe E. (2004). Case-based collaborative web search. In Proceedings of the 7th European Conference on Advances in Case-based Reasoning, Madrid, pages 489-503.

Un système coopératif pour le tri des résultats de moteurs de recherche

Rushed Kanawati

LIPN - CNRS UMR 7030
99 av. J.B. Clément 93430 Villetaneuse
rushed.kanawati@lipn.univ-paris13.fr

Résumé : Ce papier décrit un système coopératif d'aide au tri des résultats de recherches sur le Web. L'approche proposée repose sur le principe de partage implicite des expériences de recherche d'information au sein d'un groupe d'utilisateurs qui ont des centres d'intérêts communs. Un agent assistant personnel est associé à chaque membre du groupe. L'agent observe les requêtes soumises par l'utilisateur aux moteurs de recherche sur le Web et intercepte les résultats retournés. Le couple (requête, résultats) est alors transmis aux autres agents assistants qui vont chacun proposer un classement des documents retournés en fonction de leurs propres expériences. La méthodologie du raisonnement à partir de cas est utilisée par les agents pour calculer les classements. L'agent émetteur fusionne les listes triées proposées par les tous les agents et présente le résultat à l'utilisateur. Dans ce papier nous décrivons l'architecture général du système, le Cycle RàPC appliqué par chaque agent et le protocole de coopération employé. Les premiers résultats expérimentaux sont aussi rapportés.

Mots-clés : Agents RàPC, Système égal à égal, Recherche d'information, Tri de résultats, Web.

1 Introduction

La présentation et le tri des résultats des moteurs de recherches est un problème important dans le domaine de la recherche d'information sur le Web. En soumettant une requête à un moteur de recherche l'utilisateur attend en retour un ensemble de documents *triés* en fonction de leur pertinence par rapport à ses besoins informationnels. Les réponses des moteurs de recherche sont rarement conformes aux besoins de l'utilisateur. Les sources du problème sont multiples : Les requêtes utilisateurs sont souvent vagues. Les mots clés utilisés pour l'indexation des documents ne sont pas les mêmes utilisés dans les requêtes des utilisateurs. Le réel besoin informationnel de l'utilisateur est difficile à identifier à partir des requêtes posées. Le besoin informationnel est défini par le thème de recherche de l'utilisateur mais aussi par la nature des résultats attendus (on cherche un document spécifique, une adresse spécifique, des documents sur un thème, ou encore une recherche exploratoire). A tout cela s'ajoute le problème de la disparité des qualités des ressources indexées sur le W*eb*.

Différents travaux se sont intéressés récemment au problème du tri de résultats de recherche. Nous les classifions selon les trois axes suivants :

1. Approches fondées sur l'exploration de la structure du Web. L'idée est d'inférer une mesure de la qualité (ou de l'autorité) d'une ressource à partir des liens hypertextes entrants et sortants qui la relie aux autres ressources (e.g pages web). Les ressources qui ont une grande autorité sont alors placées en tête de la liste des résultats. L'exemple le plus connu de ces approches est l'algorithme *PageRank* employé par le moteur de recherche *Google* (Brin et. al. , 1998). D'autres exemples sont : l'approche *Salsa* (Lemep et.al. 2000), l'algorithme *HITS* (Ding, 2002), l'approche de Kleinberg (Keleinberg, 1999) et l'algorithme baysien proposé dans (Borodin et. al., 2002).
2. Approches fondées sur l'exploration des données d'usage. L'idée ici est d'améliorer le tri en analysant le comportement de l'utilisateur et d'inférer une sorte de *profil utilisateur* qui sera utilisé pour trier les résultats retournés par les moteurs de recherche. Des exemples de telles approches sont décrits dans (Chen et.al. , 2000), (Arezki et. al., 2004).
3. Approches coopératives ou approches orientées communauté d'utilisateurs. L'idée est d'appliquer des techniques de filtrage coopératif pour le tri des résultats. Un premier exemple est le système proposé dans (Chidlovski et. al., 2000) où les auteurs proposent un couplage entre un méta-moteur de recherche et un système de recommandation explicite de documents. Les recommandations faites dans ce dernier système sont utilisées pour trier les résultats retournés par le méta-moteur de recherche. Un autre exemple est le système I-SPY (Freyen et. al., 2004).

Dans ce papier nous proposons une approche qui combine l'exploration des données d'usage et l'approche coopérative. L'idée est de permettre à un groupe d'utilisateurs qui ont des centres d'intérêts communs (e.g. une équipe de R&D) de partager d'une manière implicite leurs expériences de recherche d'information. D'autres systèmes d'aide à la recherche d'information ont adopté une telle approche : (Kanawati et. al, 1999), (Trousse et. al., 1999). Ces systèmes sont construits selon une architecture centralisée. Or la centralisation des données d'usage pose des problèmes de sécurité et de protection des vies privées des utilisateurs. Dans notre système nous proposons une architecture complètement décentralisée où chaque utilisateur sauvegarde sur sa machine locale, ses propre trace d'usage. Le partage d'expérience est fait par des agents logiciels assistant interposés.

Le système proposé fonctionne comme suit : chaque utilisateur est associé à un agent assistant personnel. Cet agent trace dans un fichier les requêtes posées et les sélections des documents faites par l'utilisateur. L'agent intercepte les résultats retournés par un moteur de recherche et demande aux autres de reclasser ces résultats en leur envoyant la requête q et les résultats obtenus R. Chaque agent cherche dans sa base de traces des requêtes similaires à q et utilise les sélections faites par l'utilisateur pour proposer un nouveau classement de la liste R. La méthodologie de raisonnement à partir de cas (RàPC) (Aamodt et. al., 1994) est employée à cet effet. L'agent

demandeur reçoit les différents classements proposés par les différents agents et infère le classement à présenter à l'utilisateur.

La suite de l'article est organisée comme suit. La section 2 décrit l'architecture générale du système. L'architecture interne d'un agent assistant est détaillé et la constitution de la base de trace est expliquée. L'algorithme de tri est détaillé en section 3. Le cycle RàPC employé par chaque agent est décrit en détails. Les premiers résultats expérimentaux sont rapportés dans la section 4. Finalement une conclusion est donnée en section 5.

2 Architecture générale du système

Le système proposé est construit selon une architecture multi-agents de type égal-à-égal (Peer-To-Peer). Chaque utilisateur est associé à un agent assistant personnel. Un agent d'enregistrement est gère l'adhésion des agents assistants au groupe.

Un agent assistant est composé de quatre principaux modules : un module *d'interaction avec l'utilisateur*, un module de *gestion de la communauté*, un module de *communication* et un module de *tri*. Le fonctionnement du denier module représente la contribution principale de ce papier et sera détaillé en section 3. Le module de communication offre les facilités classiques de communication entre agents. Le rôle du module de gestion de la communauté est de permettre à l'agent de choisir un sous-ensemble des agents les plus compétents pour lui fournir de l'aide. Dans la version actuelle du système aucune stratégie de formation de communauté n'est implémentée. Chaque agent diffuse sa demande d'aide à tous les autres agents. Le module d'interaction avec l'utilisateur peut être vu comme un *proxy* qui s'interpose entre l'utilisateur et les moteurs de recherche utilisés. Son rôle est a) d'observer et de tracer le comportement de l'utilisateur, 2) de notifier aux autres modules les événements nécessaires pour leur fonctionnement et 3) de transférer les requêtes utilisateurs aux moteurs de recherche, et de présenter les résultats à l'utilisateur après avoir été triés par le module du tri.

Les traces du comportement de l'utilisateur sont enregistrées dans un fichier de traces (log). Ce fichier est composé d'un ensemble d'enregistrements. Un enregistrement contient les informations suivantes :

1. Identificateur de requête (*Qid*). C'est un identificateur unique attribué par le module d'interaction à chaque nouvelle requête soumise par l'utilisateur.
2. La requête (*Q*). Dans la version actuelle, nous considérons des requêtes simples composées de listes de mots clés.
3. La liste des résultats (*R*) : C'est la liste des documents retournés par le moteur de recherche en réponse à la requête *Q*. Chaque document est représenté par un couple : adresse du document (i.e. URL) et un vecteur de mots clés représentant le document. Ce vecteur est extrait automatiquement à partir du résumé envoyé par les moteurs de recherche.
4. La sélection (*S*). C'est une sous-liste ordonnée de *R* qui représente les documents sélectionnés par l'utilisateur. Nous faisons l'hypothèse qu'un document

sélectionné par l'utilisateur est jugé pertinent pour la requête *Q*. Cette hypothèse est justifiée par le fait que l'utilisateur lit d'abord le résumé avant de sélectionner un document.

Un nouvel enregistrement est crée pour chaque requête. L'enregistrement est mis à jour avec chaque sélection d'un nouveau document appartenant à la liste *R*. Un enregistrement sera fermé si aucune mise à jour n'est effectuée pendant un certain période de temps. Le fichier de trace fournit les données brutes qui seront utilisées par le module de tri pour calculer le classemen des résultats de nouvelles requêtes. Ce calcul est expliqué dans la section suivante.

3 Approche de tri coopératif

Présentation générale

Etant donné une requête *Q* et une liste de résultats *R* retournée par un moteur de recherche, l'objectif du module de tri est de trouver une permutation *R** de *R* de sorte que les documents les plus pertinents pour l'utilisateur soient placés en tête de la liste *R**. L'approche que nous proposons est la suivante : pour chaque couple <*Q, R*>, l'agent assistant A_i explore sa propre base de traces afin de calculer une permutation R_i*. En même temps l'agent A_i demande aux autres agents A_j de lui proposer des permutations R_j*. La permutation R* est donnée par une combinaison linéaire des toutes les permutations calculées et la liste originelle R. La combinaison des listes permutées est un processus similaire à celui appliqué classiquement par les méta-moteurs de recherche afin de fusionner les résultats retournés par les différents moteurs de recherche (Dwork et. al, 2001) (McCabe et. al., 1999). Noter que le processus ici est plus simple puisque toutes les listes contiennent les mêmes documents (mais dans des ordres éventuellement différents) et on n'a pas à traiter des évaluations hétérogènes des documents.

Pour calculer une permutation R_k* un agent A_k applique la méthodologie du raisonnement à partir de cas (RàPC). Le principe du RàPC est de résoudre un problème, appelé aussi *cas cible*, en réutilisant des solutions éprouvées lors de la résolution des problèmes similaires dans le passé. L'expérience de résolution de problèmes passés est stockée dans une base de cas dits *cas sources*. Un cas source, dans sa forme la plus simple est composé d'un couple <problème, solution>. Le cycle RàPC est composé de quatre étapes : une étape de *recherche* qui permet de retrouver les *k* cas sources les plus similaires au cas cible, une phase de *réutilisation* dont l'objectif est d'adapter les solutions des cas sources retrouvés afin de proposer une solution au cas cible. La solution calculée peut être révisée durant la phase de *révision*. Et enfin, le nouveau cas résolu peut être ajouté à la base de cas, pendant la dernière phase dite *d'apprentissage*, afin d'enrichir l'expérience du système.

Dans notre système un cas cible est tout naturellement composé du couple <*Q,R*> : la requête soumise par l'utilisateur et la liste des réponses retournées par le moteur de recherche. Les cas sources seront extrait des données de traces décrites en

section 2. Dans les sections suivantes nous dérivons la structure et la procédure d'extraction des cas sources. Ensuite nous décrivons les deux premiers phases du cycle RàPC.

-

Structure et extraction des cas sources

Chaque enregistrement fermé *Rec* dans le fichier de traces peut engendrer au plus un cas source. Un cas *c* est composé des attributs suivants :

1. La requête (noté *c.Q*). C'est une copie de l'attribut requête de l'enregistrement *Rec*. (*c.Q=Rec.Q*).
2. Le tri (noté *c.K*). C'est la liste des documents retournés par le moteur de recherche en réponse à la requête *Q* où les documents sélectionnés par l'utilisateur sont placés en tête (dans l'ordre de sélection). Formellement $c.K = Rec.S \oplus [Rec.R - Rec.S]$ où $\oplus$ est l'opérateur de concaténation de listes.

Lorsque un enregistrement est déclaré fermé, un cas source sera immédiatement extrait et rajouté à la base de cas. Cependant une attention particulière doit être faite afin de ne pas augmenter la taille de la base de cas par l'ajout des cas inutiles. En effet, la maintenance de la base de cas est un aspect central dans tout système de RàPC (Leake & Wilson, 2002). Mais la discussion de cet aspect dépasse le cadre de ce papier. Des heuristiques simples sont proposées afin de limiter l'ajout des cas inutiles. Aucun cas source n'est extrait dans les situations suivantes :

1. L'enregistrement a une liste de résultats vide..
2. L'enregistrement a une liste de sélections vide
3. La sélection est un préfixe de la liste de résultats. Autrement dit, l'utilisateur semble approuver le classement fait par le moteur de recherche.

La phase de recherche

L'objectif de cette phase est de retourner les *k* cas sources les plus similaires à un cas cible. *k* étant un paramètre du système. Afin d'accélérer la sélection des cas sources nous proposons un algorithme de recherche qui opère en deux étapes :

Etape 1. Lorsque l'utilisateur soumet une requête *Q*, l'agent local envoie cette requête aussi aux autres agents. Chaque agent chercher dans sa base locale les cas sources dont la similarité de l'attribut requête avec *Q* dépasse un certain seuil σ_q. La similarité entre deux requêtes *q1* et *q2* est donnée par la fonction : $Sim_{query}(q1, q2) = ||q1 \cap q2|| / ||q1 \cup q2||$. On désigne par Γq l'ensemble des cas sources retenus à l'issue de cette étape. Noter que cette étape se fait en parallèle de chargement de la liste des réponses à partir d'un moteur de recherche.

Etape 2. A l'arrivée de la liste de réponse *R*, les cas retenus dans Γq seront examinés pur déterminer les cas les plus similaires au problème <Q, R> en utilisant

la similarités entre les listes de réponses *R* et *c.K*. La similarité entre deux listes de documents *R1* et *R2* est donnée par la somme des similarités entre chaque document de *R1* et chaque document de *R2*. La similarité est normalisée en divisant la somme obtenue par la somme des tailles de *R1* et *R2*. La similarité entre deux documents est donnée par la fonction suivante :

Sim_{doc} *(d1, d2) :*
Si URL(d1)== URL(d2) Alors
Return 1
Sinon
Return $||Contenu(d1) \cap Contenu(d2)|| / ||Contenu(d1) \cup Contenu(d2)||$

Où *URL(d)* est une fonction qui retourne l'adresse du document *d* et *contenu(d)* retourne le vecteur de mots clés décrivant le document *d*. Les *k* cas les plus similaires seront alors retournés par la phase de recherche et seront utilisés par la phase suivante. On désigne l'ensemble des cas sources retournés par ΓR

La phase de réutilisation

Etant donné un cas cible composé d'un couple requête, liste de résultat <Q, R> et un ensemble de cas sources remémorés durant la phase de recherche ΓR, l'objectif de la phase de réutilisation est de calculer une permutation R_k* où les documents de *R* qui sont jugés être les plus pertinents pour l'utilisateur sont placés en tête.

L'idée de base est de permettre à chaque cas source $c \in \Gamma R$ de voter sur l'ordre relative de chaque couple de documents $d_i, d_j \in R$. Pour faciliter la compréhension du principe de vote, considérions la situation idéale où un couple d_i, d_j figure aussi dans la liste c.K. Le vote du cas *c* concernant ce couple est donné par la formule :

$vote_c(d_i, d_j) = rang(d_i, c.K) - rang(d_j, c.K)$;

où *rang(d, c.K)* retourne le rang du document *d* dans la liste ordonnée *c.K*.

Dans la réalité, les documents de *R* ne figurent pas forcément dans les listes *c.K* de tous les cas sources $c \in \Gamma R$. Pour résoudre ce problème, avant de calculer le vote d'un cas *c* sur les couples $d_i, d_j \in R$, nous commençons par associer à chaque document $d \in R$ un document $d^h \in c.K$ appelé document homologue. Par définition le document homologue à un document *d* est le document le plus similaire à *d* dans *c.K*. La similarité entre documents est calculée par la même fonction définie en section 3.3. Un seuil de similarité minimal σh doit être satisfait par les documents homologues. Si plusieurs documents dans *c.K* sont à égale distance de *d*, le document homologue sera le document le plus haut placé dans *c.K*. Si aucun document homologue à d_i n'est trouvé dans la liste *c.K* alors le cas *c* ne vote pas sur l'ordre de n'importe couple impliquant le document d_i

Pour calculer la permutation R_k* l'agent A_k applique l'algorithme suivant : La liste R_k* est initialisé à [d_1] où d_1 est le document en tête de *R*. Pour chaque document d_i suivant dans R on calcule la somme des votes des cas $c \in IR$ avec les documents d_k dans R_k*. Trois cas de figure peuvent avoir lieu :

1. La somme des votes est négative. Dans ce cas le document d_i est inséré avant le document d_k dans R_k*.
2. La somme des votes et nulle. Dans ce cas d_i est inséré immédiatement après d_k dans R_k*.
3. La somme des votes est positive. Dans ce cas, on calcule la somme des votes sur couple d_{k+1} et d_i et on applique à nouveau les mêmes règles.
4.

L'exemple suivant illustre l'algorithme proposé. Considérons une requête Q à laquelle un moteur de recherche répond par une liste R de 5 documents R= {d1, d2, …, d5}. Si aucun cas similaire au cas cible <Q, R> n'est trouvé par l'agent A_k, la permutation R_k* sera tout simplement la liste R elle-même. Autrement dit l'agent ne change pas l'ordre donné par le moteur de recherche. Maintenant, considérons la situation où l'utilisateur choisit parmi les documents proposés le document d5. Un cas source sera extrait dont la partie c.K est égale à c.K= {d5,d1,d2,d3,d4}. Si l'utilisateur soumet la même requête et le moteur répond par la même liste R. le cas source *c* sera retourné par la phase de recherche (similarité égale à 1). Les votants sont le moteur de recherche et le cas c. Les quatre premiers documents seront dans le même ordre que dans R puisqu'ils ont les mêmes ordres à la fois dans R et dans c.K. Par contre pour le document d5, le cas *c* le place avant d1 et le moteur le place après. La somme des votes est égale à :

Votes(d5,d1) = rang(R,d5)- rang(R,d1) + rang (c.K,d5) rang(c.K, d1) = 1.5 >0

Donc un deuxième vote sur le couple d5 et d2 doit avoir lieu. La somme des votes Votes(d5,d2) = 0.5. En conséquence il faut voter sur l'ordre du couple (d5, d3). Ici Votes(d5, d3) = -0.5. Par conséquent, le document d5 sera inséré dans la liste R_k* entre d2 et d3. la liste R_k* est égale à R_k*={d1, d2, d5, d3, d4}.

4 Expérimentation

Afin de valider notre approche nous avons exécuter l'algorithme sur *n* jeux de données fictives. Nous avons généré un ensemble *D* de 5000 documents. Chaque document est représenté par 1) un identificateur qui joue le rôle de l'adresse et 2) un vecteur de mots clés. Les tailles des vecteurs des mots clés varient aléatoirement entre 5 et 15 mots. Les mots clés sont sélectionnés aléatoirement parmi 2000 termes (représentés par des entiers). Pour une requête *Q*, notre simulateur de moteur de recherche retourne une liste de au plus 10 documents. La pertinence d'un document *d* pour la requête *Q* est tout simplement mesuré par : Pertinence (d,Q) = ||d ∩ Q ||/ ||d ∪ Q||. Par contre au lieu de renvoyer les listes de réponses dans l'ordre de pertinence,

on envoie une liste de 10 documents dont les 5 meilleurs sont placés dans la deuxième moitié de la liste. On simule que l'utilisateur sélectionne systématiquement trois des cinq meilleurs documents. Nous avons calculé combien de requêtes similaires faut-il avoir pour retrouver les cinq meilleurs documents en tête de la liste. Nous avons varié le seuil de similarité de 1 à 0.7. L'expérimentation est répétée dix fois et les valeurs moyennes sont relevées dans le tableau suivant :

Similarité	1	0.9	0.8	0.
Nombre de requêtes	4	5	9	12

Les résultats montrent qu'il suffit de poser la même requête 4 fois pour retrouver les meilleurs documents en tête du classement. Si la similarité des requêtes posées ne dépassent pas 0.7 il faut environ 12 requêtes pour avoir le même résultat. Ces résultats sont encourageants mais il faut évidemment valider le système dans des situations d'utilisation réelles où les utilisateurs ne sont pas toujours capables de sélectionner systématiquement les bons documents.

5 Conclusion

Ce papier décrit une approche d'aide au tri des résultats de moteurs de recherche sur le Web. L'approche présentée combine l'exploration, ou le fouille des données d'usage avec une approche coopérative. Peu de travaux dans la littérature ont abordé ce problème avec une approche similaire. Une exception est le système I-SPY (Freyen et. al., 2004). La différence principale entre notre système et I-SPY est que ce dernier est construit selon une architecture centralisée où les données d'usage de tous les utilisateurs sont groupés et traités sur une même machine. Des problèmes de performances, de et de protection de la confidentialité ne sont pas abordés. En outre, le système I-SPY ne modifie pas le classement des résultats que pour les requêtes qui sont déjà soumises telle qu'elles (similarité égale à 1). Outre la validation dans des situations réelles d'usage, plusieurs problèmes intéressants restent à résoudre. Nous citons en particulier le problème de formation de communauté, et le problème de prise en compte de la co-existence au sein du groupe des utilisateurs qui ont des comportements et des objectifs différents tout en ayant les mêmes centres d'intérêts. Ces deux problèmes sont sur notre agenda de travaux futurs

Références

AAMODT A., PLAZA E. (1994), Case-based Reasoning: Foundational issues, Methodological variations and system approaches. AI communications 7(1):39-59

AREZKI, R. PONCELET P., DRAY G. AND PEARSON D.W. (2004). PAWebSearch: An Intelligent Agent for Web Information Retrieval. In proceedings of the International Conference on

Advances in Intelligent Systems: Theory and Applications (AISTA 2004), Luxembourg, November 15-18, 2004.

BRIN S. AND PAGE L. (1998) The anatomy of large scale hypertextual web search engine. In proceedings of th 7th International conference on the world wide web. Brisbane, Australia.

BORODIN A. ET. AL., (2002) Finding authorities and hubs from link structures on the world wide web. In proceedings of the 10th International conference on the word wide web, Hong Kong

CHEN Z AND MENG X. (2000) Yarrow: real-time client side Meta-search learner. In proceedings of the AAAI workshop on artificial intelligence for web search (AAAI'00). 12-17 July Austin. AAAI press pp. 12-27

CHIDLOVSKI B. ET; AL. (2000) Collaborative Re-Ranking of search results. In proceedings of the AAAI workshop on artificial intelligence for web search (AAAI'00). 12-17 July Austin. AAAI press pp. 18-22

DING C. (2002) PageRank, HITS and a unified framework for link analysis. In proceedings of the 25th AM SIGIR conference, Tampere, Finland

DWORK C. ET. AL. (2001) Rank aggregation methods for the web. In proceedings of the 10th International conference on the word wide web, Hong Kong pp. 613-622

FREYEN J., SMYTH B. COLE M. BALLE EVELN AND BRIGGS P.(2004) Further Experiments on Collaboration Ranking in Community based Web search. Artificial Intelligence Review, 21(3-4), pp. 229-252.

KANAWATI R., JACZYNSKI, M., TROUSSE B., ANDREOLI J.M. Applying the Broadway Recommandation Computation Approach for Implementing a Query Refinement Service in the CBKB Meta-search Engine, Conférence française sur le raisonnement à partir de cas (Ra`PC'99), Palaiseau, France, juin, 1999.

KEINBERG J. (1999) Authoritative links in a hyperlink environment. Iin journal of the ACM (JACM)44.

LEAKE D.,WILSON D.C. (2002) Maintaining case-based reasoners: diemensions and dierections Computational Intelligence 27(2)

LEMPLE R., MORRIN S. (2000) The stochastic for link structure analysis SALSA and the TKC effect. In proceedings of the 9th International conference on the word wide web

MCCABE M. ET. AL. (1999). A unified environment for fusion of information retrieval approaches. In ACM CIKM conference pp. 330-334

TROUSSE B., JACZYNSKI M., KANAWATI R., Using User Behavior Similarity for Recommandation Computation: The Broadway Approach, 8th international conference on human computer interactions (HCI'99), Munich, August, 1999.