TRAITEMENT POSSIBILISTE POUR LA DEFINITION DE CLASSES D'OBJETS FLOUS D'UN RESEAU SEMANTIQUE

Mohamed-Nazih OMRI

LAFORIA-IBP
Université Paris VI, Boîte 169
4, Place Jussieu, 75252 Paris Cedex 05
tel: 44 27 70 10
&
Laboratoire de Psychologie Cognitive, URA CNRS 1297
Université Paris VIII
2, rue de la liberté, 93526 Saint-Denis Cedex 02 France

RESUME: Dans cet article, nous proposons une méthode de représentation des connaissances procédurales qui repose sur la notion de classes d'objets flous. Les propriétés ou attributs décrivant une classe présentent la distinction entre ensemble de valeurs nécessaires (ensemble de valeurs issue d'une distribution de nécessité qui désignent cette propriété avec un certain degré) et ensemble de valeurs supplémentaires (ensemble de valeurs issue d'une distribution de possibilité qui permettent une confirmation éventuelle de la désignation de cette propriété avec un certain degré). On définit également pour chaque valeur d'attribut une zone sujet permettant de faciliter l'identification et l'intégration des objets et procédures fournis par un interlocuteur novice en termes du langage naturel, pour les ranger dans des catégories de classes d'objets d'un système d'aide à l'utilisation de dispositifs techniques à base floue de connaissances. L'intégration de valeurs floues dans la description de classes d'objets entraîne une valuation des liens hiérarchiques entre classes d'une part et entre instances et classes d'autre part. On parle d'héritage pour désigner le transfert de propriétés d'une classe vers une autre et d'instanciation dans le cas de transfert de propriétés d'une classe vers une instance.

Cette recherche a pour objectif la conception d'interfaces de dialogue entre experts et novices, pour attribuer à des énoncés imprécises une signification basée sur l'appartenance catégorielle.

MOTS-CLES: Classes floues, attributs flous, zones floues, imprécision, incertitude, représentation des connaissances procédurales, aide à l'utilisation de dispositifs techniques.

ABSTRACT: This paper presents a method for procedural knowledge representation based on fuzzy object class notion. This representation involves fuzzy and uncertain techniques. Both a fuzzy range of uncertain values and a fuzzy range of supplementary values can be specified for the attributes describing object's class. Various kinds of inclusion relations are defined between classes, as well as membership relation between instances and classes. For each attribute, we define an area which grows with the user's queries and which is supposed to users facilitate the analysis of the next user query. The transfer of properties from one class to an other class or from one class to an instance is made thanks to membership relation. We talk about heritage to designate the first kind of properties transfer and about instanciation to designate properties transfer from one class to an instance.

1. INTRODUCTION

Les systèmes d'aide en ligne classiques ne traitent que des requêtes exprimées par des valeurs précises alors qu'elles sont souvent formulées de manière imprécise et/ou incertaine. Les besoins de l'utilisateur étant eux mêmes imprécis. Il en résulte un manque de flexibilité et de souplesse dans la recherche d'informations.

Dans notre travail, nous nous sommes attachés à rechercher les meilleurs moyens et méthodes pour représenter et gérer convivialement des connaissances dans un réseau sémantique flou avec comme cadre théorique la théorie des sous-ensembles flous [16] et la théorie des possibilités [5, 18] qui permettent de traiter les informations imprécises et incertaines.

Nous avons introduit, dans la définition des classes d'objets de SIFADE [7], le concept de connaissances imprécises et incertaines pour représenter une connaissance (un but ou un objet) par deux couples de valeurs : ((valeur linguistique$_i$, degré de possibilité) (valeur linguistique$_j$, degré de nécessité)).

Deux raisons essentielles ont guidé un tel choix:

- Détenir une représentation aisée des objets complexes ayant des attributs à valeurs complexes.
- Obtenir une grande flexibilité et de la souplesse quant à l'analyse d'une requête sujet éventuellement imprécise.

Ce formalisme présente l'intérêt d'amiliorer la description de connaissances imprécises et incertaines et offre des moyens simples pour les manipuler, tenant compte du fait qu'il permet un enrichissement dynamique du réseau sémantique au fur et à mesure de son utilisation, donnant une satisfaction plus importante à l'utilisateur que les systèmes classiques surtout quand il s'agit de fournir une aide à l'utilisation de dispositifs techniques.

2. APPARIEMENT ENTRE LES CONNAISSANCES SUJETS ET CELLES DU RESEAU SEMANTIQUE DE L'EXPERT IDEAL

Etant donné qu'une interface de dialogue entre novice et expert nécessite des requêtes utilisateurs, le problème qui se pose est celui de l'identification ou de la correspondance entre le contenue d'une requête (l'étiquette de l'objet concerné et celle de la procédure à appliquer sur cet objet) à leurs entités correspondantes (classes d'objets et valeurs d'attributs) du réseau sémantique utilisateur.

L'exemple du système technique que nous considérons, est un logiciel dont les objets sont telles que "chaîne-de-caractères", et les procédures sont telles que "couper" ou "copier". Pour un utilisateur novice d'un tel logiciel, une liste de dénomination standard n'est toujours pas évidente et il doit souvent demander à un opérateur expert de lui montrer comment exécuter une action sur un objet telle que "Comment gommer les lettres". Le système SIFADE [7] que nous proposons prend la place de l'utilisateur expert. Quand le sens d'une instruction donnée est ambigu, SIFADE dialogue avec le sujet jusqu'à ce qu'il interpréte la requête dans le langage propre (langage naturel) de ce dernier.

Avec l'apprentissage de nouveaux mots du langage naturel, sous forme d'interprétation produite en accord avec le sujet, le système perfectionne son schéma de représentation par l'expérience, à l'aide de nouveaux sujets et, en plus, il améliore les dialogues prévus avec les utilisateurs : (i) les objets et procédures standards reconnus

par le logiciel seront représentés dans le réseau sémantique des utilisateurs [7], (ii) plus les requêtes utilisateurs sont exprimées en langage naturel et correspondent imparfaitement à leurs dénominations standards et plus le système établit des liens flous de connection entre ses connaissances principales (celles de l'expert idéal) et les nouvelles étiquettes des objets et procédures utilisateurs.

Par exemple, si une procédure peut être identifiée de plusieurs manières, il convient d'utiliser des degrés de confiance pour chacune des identifications. Pour cette raison, nous avons introduit la notion de sous-ensemble flou et de possibilité pour décrire les caractérisations des classes d'objets dont les valeurs de l'attribut procédure sont les procédures à appliquer sur ces objets. Les mesures de possibilité et/ou d'incertitude nous permettent de quantifier l'opinion des utilisateurs concernant le degré de similarité ou de convenance d'association entre les descriptions des objets et procédures.

Dans le paragraphe suivant, nous présentons les objets flous basés sur la représentation des connaissances. Nous discutons des degrés sur les liens hiérarchiques entre objets et procédures formulés par le sujet par rapport aux classes du réseau sémantique flou. Nous parlons enfin du transfert de propriétés d'une classe vers une autre classe (ou d'héritage), et du transfert de propriétés d'une classe vers une instance (ou d'instanciation).

La représentation que nous proposons, prend aussi en compte, les degrés d'inclusions entre classes d'objets flous et d'appartenances d'une instance à une ou plusieurs classes. Si Rossazza [9, 10] a introduit le concept d'objet flou et Dubois et Prade [5] ont introduit la notion de typicalité dans la hiérarchie de classes d'objets flous, peu de travaux ont été développés dans cette direction, spécialement pour la planification des actions.

Dans cette communication, tous les exemples sont pris dans le cadre d'une application basée sur l'utilisation d'un traitement de texte.

3. PRINCIPES DE REPRESENTATION

Pour représenter des connaissances de sens commun, il paraît indispensable d'utiliser différents natures de propriétés (ou de liens) : nécessaires, suffisantes et supplémentaires pour Torasso et Console [13], nécessaires et typiques pour Padgham [8]**.** Si la première organisation apparaît très puissante de point de vue théorique, souvent la condition de suffisance n'est pas utilisée en pratique, il y a une très grande difficulté à la définir [13]. Dans le cadre de notre formalisme, on propose la distinction entre propriétés nécessaires et propriétés supplémentaires. Les propriétés suffisantes ne seront pas prises en compte. Chaque propriété d'un attribut d'un objet est décrite par trois zones floues : une zone nécessaire, une zone supplémentaire (ou zone possible) et une zone utilisateur.

Dans ce paragraphe, * désigne soit N (pour Nécessaire) soit S (pour Supplémentaire) soit U (pour Utilisateur).

3.1 Les zones floues

La zone nécessaire ZN(P, A) d'une propriété P d'un attribut A, est l'ensemble de couples de valeurs (propriété nécessaire, cœfficient de nécessité) permises pour l'attribut A relativement à la propriété P. Cette zone est floue car certaines de ses valeurs

sont moins admissibles que d'autres. La valeur de ZN(Effacer, procédure) représentée dans l'exemple ci-dessous est obtenue à l'aide d'une distribution de nécessité donnée par un expert utilisant les buts **CouperParMenu** et **EffacerParMenu** pour désigner la propriété système **Effacer**.
Exemple :

attribut : **procédure**
domaine : [EffacerParMenu, Sélectionner, CouperParMenu, Coller]
valeur : Effacer
Zone nécessaire : [(CouperParMenu .75) (EffacerParMenu .9)]

La zone supplémentaire ZS(P, A) d'une propriété P d'un attribut A, est l'ensemble de couples de valeurs (propriété supplémentaire, cœfficient de possibilité) permises pour l'attribut A relativement à la propriété P. Cette zone est floue car certaines de ses valeurs sont moins possibles que d'autres. Elle est obtenue à l'aide d'une distribution de possibilité fournie par un expert du système relativement à la propriété système Effacer.

Exemple :

attribut : **procédure**
domaine : [EffacerParMenu, Sélectionner, CouperParMenu, Coller]
valeur : Effacer
Zone supplémentaire : [(CouperParMenu 1) (EffacerParMenu 1)]

La zone utilisateur ZU(P, A) d'une propriété P d'un attribut A, est l'ensemble de couples de valeurs (propriété sujet, cœfficient de possibilité), obtenues à partir d'une distribution de possibilité fournie par le sujet, qu'un attribut A peut prendre relativement à P. Reprenons l'exemple précédent,

attribut : **procédure**
domaine : [EffacerParMenu, Sélectionner, CouperParMenu, Coller]
valeur : Effacer
Zone utilisateur : [(Gommer .8) (Enlever .75)]

Cette zone exprime le fait que le sujet a utilisé les propriétés **Gommer** et **enlever** pour désigner le but **Effacer**.

Les différentes zones décrites précédemment permettent de définir les attributs constitutifs des classes d'objets.

3.2 Les attributs flous

Soit P_i, $i \in <1, k>$ l'ensemble des propriétés d'un attribut A, la structure d'un attribut d'un objet prend la forme suivante :

attribut : #:[**(**P_1 (ZN_1) (ZS_1) (ZU_1)**)**...**(**P_i (ZN_i) (ZS_i) (ZU_i)**)** ..**(**P_k (ZN_k) (ZS_k) (ZU_k)**)**]

Exemple :

procédure : [(**Effacer** ((CouperParMenu .8)) ((Déplacer 1)) ((gommer .75)))
(**Sélectionner** ((SelectionParcourant .9)) ((SelectionDelimitant .75)) ((Noircir 1)))]

Trois types d'attributs peuvent ainsi être définis : attribut nécessaire noté AN, attribut supplémentaire noté AS, et attribut utilisateur noté AU. Il sont définis par:

$$A^*(C) = Z^*(P_1, A) \times Z^*(P_2, A) \times ... \times Z^*(P_k, A).$$

Ces attributs sont flous car ils sont le résultat d'un produit cartésien de zones floues. L'ensemble de ces attributs décrit les classes d'objets et les instances de classes. Ces classes et ces instances sont définies par l'ensemble de leurs attributs nécessaires et de leurs attributs supplémentaires. Les propriétés de AU sont donc des propriétés sujets et par conséquent, elles ne peuvent participer à la définition théorique d'une classe ou d'une instance de classe.

3.3 Les classes d'objets floues

Soient AN_i, AS_i, AU_i, $i \in <1, n>$, l'ensemble des attributs nécessaires, supplémentaires et utilisateurs définissant une classe C. On définit respectivement une classe nécessaire CN, une classe supplémentaire CS et une classe utilisateur CU par :

$$C^* = A^*_1(C) \times A^*_2(C) \times ... \times A^*_n(C).$$

3.4 Les instances floues

Soient AN_i, AS_i, AU_i, $i \in <1, n>$, l'ensemble des attributs nécessaires, supplémentaires et utilisateurs définissant une instance I. On définit respectivement une instance nécessaire IN, une instance supplémentaire IS et une instance utilisateur IU par analogie à une classe par :

$$I^* = A^*_1(I) \times A^*_2(I) \times ... \times A^*_n(I).$$

4. LIENS HIERARCHIQUES

L'intégration de propriétés floues, telles que **Gommer** et **enlever,** dans la description d'objets va entraîner une valuation de ces liens dans l'intervalle [0, 1]. Une classe par exemple, au lieu d'être (ou ne pas être) une sorte d'autre classe, pourra l'être plus ou moins. On distingue deux types de liens: le lien **sorte-de** entre deux classes d'objets et le lien **est-un** entre une classe et une instance. Chaque type de lien va être décrit par un couple de valeurs (coefficient de nécessité, coefficient de possibilité) qui sera calculé en fonction du degré d'inclusion entre deux classes et d'appartenance d'une instance à une classe.

Exemple :

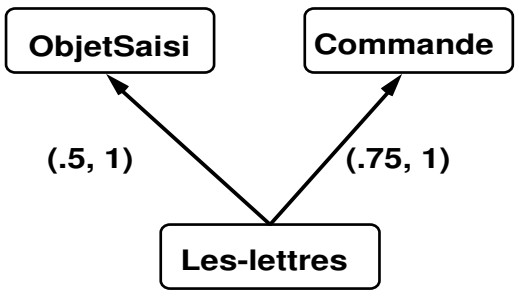

figure 4 : lien valué **sorte-de**

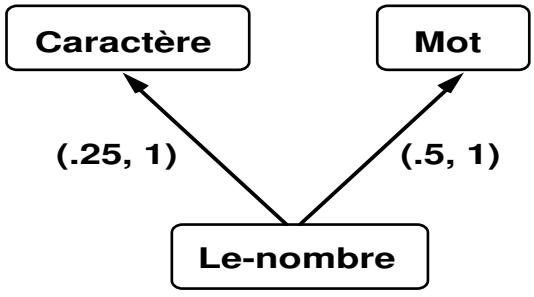

figure 5 : lien valué **est-un**

Dans la figure 4, l'objet "les-lettres" est une **sorte-de** "commande" avec un coefficient de nécessité égale à 0.75 et un coefficient de possibilité égal à 1. C'est aussi une **sorte-de** "objet-saisi" avec un coefficient de nécessité égale à 0.5 et un coefficient de possibilité égal à 1. Si dans la figure 5 on suppose que "le-nombre" est un objet terminal d'un réseau sémantique, On dit alors que c'est un représentant physique de l'objet **Mot** avec un degré de nécessité égal à 0.5 et un degré de possibilité égal à 1 et de l'objet **Caractère** avec un degré de nécessité égal à 0.25 et un degré de possibilité égal à 1.

4.1 Inclusion entre zones floues

Soient $p_1, p_2,..., p_k$ des valeurs linguistiques nécessaires pour la propriété P d'un attribut A d'une classe C_1, $n_1, n_2,..., n_k$ leurs degrés de nécessité et $s_1, s_2,..., s_k$ des valeurs linguistiques nécessaires pour la propriété S d'un attribut A' d'une classe C_2, $m_1, m_2,..., m_k$ leurs degrés de nécessité. On définit un degré d'inclusion entre deux zones floues par:

$$D(Z^*(P, A) \supset Z^*(S, A')) = \min_{1 \le j \le l} \min_{\substack{1 \le i \le k \\ p_i = s_j}} (n_i, m_j).$$

Il s'applique soit à la zone nécessaire soit à la zone supplémentaire.

Dans toutes les formules qui suivent , * désigne soit la lettre N (pour désigner nécessaire), soit la lettre S(pour désigner supplémentaire)

4.2 Inclusion entre attributs flous

Soient i le nombre de propriétés P d'un attribut A d'une classe C_1, j le nombre de propriétés S du même attribut A d'une classe C_2. On définit un degré d'inclusion entre deux attributs nécessaires et un degré d'inclusion entre deux attributs supplémentaires par :

$$D((A^*, C_1) \supset (A'^*, C_2)) = \min_{1 \le p \le j} \{ \min_{1 \le q \le i} \{ D(Z^*(P_q, A^*) \supset Z^*(S_p, A'^*))\}\}$$

4.3 Inclusion entre classes floues

On définit le degré d'inclusion d'une classe nécessaire CN dans une classe C et le degré d'inclusion d'une classe supplémentaire dans la classe C par :

$$D(C \supset C^*) = \min_{1 \le i \le n} \{ D((A^*_i, C) \supset (A'^*_i, C^*))\}.$$

où $A^*_1, A^*_2,..., A^*_n$ sont les attributs définissant la classe floue C et $A'^*_1, A'^*_2,..., A'^*_n$ sont les attributs définissant la classe floue C*.

4.4 Appartenance d'une instance à une classe

Dans le cas d'une classe et d'une instance on parle souvent de degré d'appartenance de l'instance à la classe. Ce degré mesure la représentation physique de la classe par l'instance. Le degré d'appartenance est calculé en fonction du degré d'inclusion entre zones et entre attributs comme dans le cas de deux classes.

L'évaluation de l'appartenance d'une instance I à une classe C est donnée par deux types de degré d'inclusion : un degré d'inclusion entre une classe et une instance nécessaires et un degré d'inclusion entre une classe et une instance supplémentaire.

$$D(I^* \in C) = \min_{1 \le i \le n} \{ D((A^*_i, C) \supset (A^*_i, I^*))\}.$$

n étant le nombre d'attributs de la classe C.

Dans le cas classique, si I est une instance de C et C est incluse dans E sur-classe de C alors I est une instance de E. D'après les définitions précédentes des degrés d'appartenance et d'inclusion nous pouvons démontrer que cette propriété est préservée, par le fait que :

si $D(I \in C) > 0$ et $D(E \supset C) > 0$ alors $D(E \supset I) > 0$.

5. HERITAGE ET INSTANCIATION

Dans les paragraphes précédents, on a calculé les valuations des liens hiérarchiques **est-un** et **sorte-de** à partir des descriptions des objets d'un réseau sémantique flou. Dans ce paragraphe, on définit une classe (resp. une instance de classe) comme une sorte de (resp. comme appartenant à) une ou plusieurs classes afin de réaliser le transfert de toutes ou quelques propriétés de leurs descriptions. Une fois la description complète de la sous-classe obtenue, on calcule la valuation du lien qui existe entre les deux classes.

Nous parlerons d'héritage pour désigner le transfert de propriétés d'une classe vers une autre et d'instanciation pour désigner le transfert de propriétés d'une classe vers une instance.

5.1 Héritage

Le transfert de propriétés d'une classe vers une autre se fait grâce au lien sorte-de. On distingue trois types d'héritage qui peuvent être simples ou multiples : héritage nécessaire, héritage normal et héritage supplémentaire.

5.1.1 Héritage multiple : On parle d'héritage multiple pour une classe C lorsque celle-ci possède plus d'une sur-classes. La classe C est donc décrite par la réunion de tous les attributs de ses sur-classes. Pour un attribut donné, la zone hérité d'une propriété donnée est l'intersection des zones correspondantes des propriétés dans chacun des attributs des sur-classes de C. On distingue trois types d'héritage multiple : héritage multiple nécessaire, héritage multiple normal et héritage multiple supplémentaire.

- héritage multiple nécessaire : les **zones supplémentaires** des propriétés des attributs de C reçoivent l'intersection des **zones nécessaires** des propriétés des attributs correspondants des sur-classes de C.

- héritage multiple normal : les **zones nécessaires** des propriétés des attributs de C reçoivent l'intersection des **zones nécessaires** des propriétés des attributs correspondants des sur-classes de C et les **zones supplémentaires** des propriétés des attributs de C reçoivent l'intersection des **zones supplémentaires** des propriétés des attributs correspondants des sur-classes de C.

- héritage multiple supplémentaire : les **zones nécessaires** des propriétés des attributs de C reçoivent l'intersection des **zones supplémentaires** des propriétés des attributs correspondants des sur-classes de C.

Exemple : héritage multiple normal.

Considérons les classes d'objets suivantes :

ChaineDeCaractère

....

Attribut :	procédure
Domaine :	{Déplacer, Modifier, Recopier, Selectionner, ...}
Valeur :	Déplacer
Zone-nécessaire :	{(Déplacer 1) (Modifier .5) (Remplacer .7)}
Zone-supplémentaire :	{(Remplacer .8)}

Caractère

....

Attribut :	procédure
Domaine :	{Déplacer, Modifier, Recopier, Selectionner, ...}
Valeur :	Déplacer
Zone-nécessaire :	{(Déplacer 1) (Modifier .6) (Recopier .3)}
Zone-supplémentaire :	{(Remplacer .3) (Selectionner .2)}

Mot

....

Attribut :	procédure
Domaine :	{Déplacer, Modifier, Recopier, Selectionner, ...}
Valeur :	Déplacer
Zone-nécessaire :	{(Déplacer 1) (Modifier .7)}
Zone-supplémentaire :	{(Selectionner .2) (Remplacer .5)}

Si par exemple, l'objet sujet **Les-signes** a été identifié à la fois à la classe ChaineDeCaractère, à la classe Caractère et à la classe Mot, ceci nous permet d'obtenir par un héritage normal :

Les-signes

....

Attribut :	procédure
Domaine :	{Déplacer, Modifier, Recopier, Selectionner, ...}
Valeur :	Déplacer
Zone-nécessaire :	{(Déplacer 1) (Modifier .5)}
Zone-supplémentaire :	{(Remplacer .3)}

Remarquons que l'héritage concerne uniquement les zones floues qui interviennent dans le calcul du degré d'inclusion.

5.1.2 Héritage simple : L'héritage simple étant un cas particulier de l'héritage multiple dans la mesure où la classe C possède une seule sur-classe. Trois types d'héritage peuvent ainsi être définis : héritage simple nécessaire, héritage simple normal et héritage simple supplémentaire.

5.2 Instanciation

L'instanciation peut être entre une instance et une classe (sa classe), ce qui correspond à une instanciation simple ou entre une instance et plusieurs classes (ses classes) ce qui correspond alors à une instanciation multiple.
On distingue trois types d'instanciations simples et trois types d'instanciations multiples : instanciation simple (resp. multiple) nécessaire, instanciation simple (resp. multiple) normale et instanciation simple (resp. multiple) supplémentaire.

Le processus de transfert de propriétés dans le cas d'une instanciation est le même que dans le cas d'héritage entre classes.

6. LA LIMITE DE L'APPROCHE : Problème de recherche d'informations.

La distinction entre propriétés nécessaires et propriétés supplémentaires pour l'amélioration de la description des connaissances procédurales de sens communs et les moyens qu'offre cette approche pour les manipuler semblent insuffisants pour permettre à un mécanisme d'identification et de recherche d'informations, basé sur l'appartenance catégorielle, de saisir la connaissance la plus adéquate vis-à-vis d'une situation donnée. En effet, quand un sujet désire réalisé un objectif donné dont il ne connait pas le nom de la procédure permettant de l'atteindre, un système à base floue de connaissances [2] muni d'un tel mécanisme, se trouve souvent devant plusieurs solutions possibles. Il ne dispose d'aucun moyen supplémentaire permettant de sélectionner la connaissace recherchée dans un temps limité. La recherche d'informations se fait presque aléatoire par rapport à un ensemble de données. L'analyse des objets de la base ne présente aucune priorité des uns par rapport aux autres et demande par fois, un parcours de tout le réseau sémantique pour tester tous ses objets.

Ceci a pour inconvénient de mèttre en cause la qualité des performances qu'on attend d'un tel système. Ce problème fait, actuellement l'objet de nos travaux.

7. CONCLUSION

L'approche décrite permet de définir des classes d'objets floues pour la représentation des connaissances procédurales et déclaratives eventuellement imprécises et incertaines. Cette représentation présente un grand intérêt pratique dû à la possibilité de réaliser des classifications dans un système d'aide en ligne à base floue de connaissances telque SIFADE [7]. Il est également important de souligner que dans cette approche, la distinction entre classes nécessaires, classes supplémentaires et classes sujets permet une prise en compte facile des connaissances imprécises et incertaines et offre des moyens simples pour les représenter, les interroger et les manipuler. Notre travail a pour objectif de répondre à ces besoins dans le cadre des systèmes d'aide à l'utilisation de dispositifs technique à base floue de connaissances.

Pour accroître la puissance de représentation de notre modèle, on envisagera de définir un degré d'instabilité (donnant une indication sur la stabilité) relatif à chaque classe d'objets et à chaque propriété du système et un degré de similarité ou de ressemblance entre deux objets de même nature afin de faciliter l'identification des objets et procédures sujet pour attribuer à des énoncés imprécises une signification basée sur l'appartenance catégorielle.

REFERENCES BIBLIOGRAPHIQUES

[1] Bouchon-Meunier B., Yager R.R., Zadeh L.A. (eds.) (1991). Uncertainty in knowledge bases. Lecture Notes in Computer Science, 521, Springer Verlag.

[2] Bouchon-Meunier B., Tijus C.A., Omri M.N., (1994). Fuzzy Sets Based System for User's Assistance: how SIFADE diagnoses user's goals. The Second World Congress on Expert Systems, Estoril / Lisbon.

[3] Card, S.K., Moran, T.P., Newell A. (1983). The Psychology of Human-Computer Interaction. Hillsdale, N.J.: Erlbaum.

[4] Carrol J.M., Mack R.L. (1985). Metaphor, computing systems and active learning. International Journal of Man-Machine Studies, 212, 39-57.

[5] Dubois D. et Prade H. (1985), Théorie des possibilité- applications à la représentation des connaissances en informatique- masson.

[6] Green T.R.G., Schiele F, Payne S.J. (1992). Formalisable models of user knowledge in human-computer interaction in working with computers: Theory versus Outcome Van Der Veer G.C., Green T.R.G., Hoc J.M, Murray D.M. (eds.) Londres Academic Press.

[7] Omri M.N. (1993). Interface Homme-Machine acceptant l'imprécision: SIFADE. Congrès INFORSID, Lille 93.

[8] Padgham L.(1988). A Model and Representation for Type Information and Its Use in Reasonning with Défaults. Proceedings of AAAI, vol 2, p 409-414.

[9] Rossazza J.P. (1990). utilisation de hiérarchie de classes floues pour la représentation des connaissances imprécises et sujettes à exceptions: système SORCIER. Thèse de 3ème cycle, Université Paul Sabatier.

[10] Rossazza J.P. (1992). Représentation centrée objet, 2ème Congrès National sur les applications des ensembles flous, Nîmes.

[11] Tauber M. (1988). On mental models and the user interface. Human-computer interaction in Working with Computers: Theory versus Outcome Van Der Veer G.C., Gren, T.R.G., Hoc J.M., Murray, D.M. (eds.) London. Academic Press.

[12] Tijus C.A., Poitrenaud S. (1992). Semantic Networks of Action. NATO conference on Psychological and Educational Foundations of Technology-Based Learning Environments. Kolymbari (Crete).

[13] Torasso P., Console L. (1989). Approximate Reasonning and Protypical knowledge. International journal of Approximate Reasonning, vol 3, no 2, pp 157-177.

[14] Vincent A. (1989). Filtrage sémantique dans une base de données imprécises et incertaines : un système souple autorisant la formulation de requêtes composites pondérées. Thèse de 3ème cycle, Université Paul Sabatier.

[15] Weber S. (1983). A general concept of fuzzy connectives, negations and applications based on norms and t-co-norms. Fuzzy Sets and Systems, 11, p115-134.

[16] Wright P., Barnard P. (1975). Just fill in this form: a review for designers, Applied Ergonomics, 6, 4, 213-220.

[17] Zadeh L.A. (1965). Fuzzy sets-information and control, vol.8, pp 338-353.

[18] Zadeh L.A. (1978). Fuzzy sets as basis for theory of possibility-Fuzzy sets and systems, vol. 1, N°1.