Une expérience de représentation d'une ontologie dans le médiateur PICSEL

Chantal Reynaud, Brigitte Safar, Hélène Gagliardi

Une expérience de représentation d'une ontologie dans le médiateur PICSEL

C. Reynaud[*†], B. Safar*, H. Gagliardi**

[*] L.R.I., C.N.R.S., Université Paris 11,
Bât. 490, 91405 Orsay cedex,
[†]Université Paris 10, 200 av. de la
république, 92001 Nanterre cedex

http://www.lri.fr

{cr, safar, gag}@lri.fr

Résumé : La construction d'un serveur d'information, relatif à un certain domaine d'application, au-dessus de sources d'information distribuées et hétérogènes, repose sur l'exploitation de connaissances correspondant à la description du domaine du serveur, encore appelée *ontologie du domaine*. Cet article traite du processus de représentation de telles ontologies dans le cadre du projet PICSEL. Deux étapes sont décrites. La première est guidée par le langage et les fonctionnalités du serveur d'information. La seconde, qui vise à affiner et réorganiser les connaissances représentées à la première étape, est guidée par la façon dont les fonctionnalités du serveur sont mises en œuvre. Les exemples cités dans l'article sont issus du domaine des produits du tourisme, domaine d'expérimentation choisi au sein du projet PICSEL.

Mots clés : ontologie, représentation de connaissances, logique de description, recherche d'informations, médiateur.

Abstract : Building an information server, specific to a given domain, over distributed and heterogeneous information sources, is based on knowledge describing the server domain, called domain ontology. We address the issue of representing such ontologies in the framework of the PICSEL research project. Two steps are described. The first one is directed by the representation formalism and the functionalities of the information server. The second one aims at refining and optimizing knowledge obtained in step one. It is guided by the way functionalities of the server are implemented. The examples in the paper are coming from the tourism products domain.

Key words : ontology, knowledge representation, description logics, information retrieval, mediator.

1. INTRODUCTION

Le projet PICSEL, mené par l'équipe IASI du LRI dans le cadre d'un contrat CNET-CTI, vise à construire des serveurs d'information au-dessus de sources d'information distribuées et hétérogènes relatives à un même domaine d'application. Ces serveurs, encore appelés *médiateurs*, sont l'interface entre des utilisateurs et des sources d'information et permettent de donner l'impression d'interroger un système centralisé et homogène. Tout médiateur se compose de deux parties : un moteur de requêtes générique et des bases de connaissances spécifiques à un serveur donné. Parmi les bases de connaissances, se trouvent la description du domaine d'application du serveur, appelée aussi *ontologie du domaine,* et la description abstraite du contenu de chaque source d'information accessible. Le rôle d'une ontologie est double. Elle permet de fournir aux utilisateurs un vocabulaire de base approprié pour formuler leurs requêtes, et également d'établir une connexion entre les différentes sources d'information à intégrer.

Les travaux de recherche sur les médiateurs tels Tsimmis (Garcia-Molina *et al*. [1997]), Information Manifold (Levy *et al*. [1996]) ou Infomaster (Duschka & Genesereth [1997]) se sont intéressés en priorité aux aspects algorithmiques, motivés par l'obtention d'un système opérationnel implémentant des algorithmes ayant de bonnes propriétés. En général, ces travaux ne traitent des ontologies qu'au travers du choix du langage de représentation utilisé.

Depuis peu, les ontologies sont l'objet d'un regain d'intérêt de par leur rôle clé dans le cadre du Web sémantique. En effet, elles permettent de réunir un ensemble de termes partagés, précisément définis, utilisables pour la description du contenu de sources d'information accessibles via le Web. Ce besoin, aujourd'hui nettement identifié, a conduit à proposer de nouveaux langages (XML Schema, RDF, RDF Schema). Leur objectif est de faciliter la représentation du contenu de documents électroniques et l'échange d'ontologies via le Web. Le pouvoir d'expression de ces langages n'est cependant pas très important et aucun mécanisme de raisonnement automatique ne leur est associé. Ces constats ont conduit au développement des langages OIL (Fensel *et al*. [2000]) et DAML (Hendler & McGuiness [2000]), deux langages pour la représentation d'ontologies qui étendent RDF Schema avec un ensemble de primitives de modélisation plus riche. Leur sémantique, basée sur une logique de description, est simple, claire et bien définie. Des mécanismes de raisonnement automatiques leur sont associés. En particulier, OIL a été conçu de façon à permettre des traitements automatiques, via une mise en correspondance avec la logique de description $\mathcal{SHIQ}$ et le système FaCT (Fast Classification of Terminologies). Un des objectifs à l'origine de OIL était de proposer un langage standard pour la construction d'ontologies. Sa conception a alors été faite de façon totalement indépendante de l'exploitation des ontologies construites.

Notre travail traite des ontologies dans le cadre du médiateur PICSEL. Le langage de représentation des connaissances utilisé est CARIN-$\mathcal{ALN}$, un langage riche permettant de décrire de façon très fine l'ontologie d'un domaine et le contenu de sources d'information. CARIN-$\mathcal{ALN}$ est un formalisme logique qui

combine un langage de règles logiques avec une logique de description. Comme tout langage formel, il a l'avantage de pouvoir faire l'objet de traitements automatisés. Dans PICSEL, les choix en matière de représentation des connaissances ont été faits de façon à faciliter l'expression de connaissances *dans le cadre d'un médiateur* et pour que les traitements automatisés effectués *dans ce cadre* soient dotés de bonnes propriétés. Le langage garantit, en particulier, la décidabilité du calcul complet de l'ensemble des plans de requêtes.

L'objet de cet article n'est pas de justifier le choix du langage utilisé dans PICSEL. Nous renvoyons le lecteur à (Goasdoue *et al*. [2000]) sur ce sujet. Nous supposons, au contraire, que ce choix a été effectué et qu'il fournit au concepteur du système un cadre de travail qui le contraint et le guide pour construire l'ontologie. Notre objectif est de décrire le processus de construction d'une ontologie dans PICSEL, étant donné ce cadre, c'est-à-dire en nous intéressant plus spécifiquement à l'aspect représentation de l'ontologie. La distinction entre modélisation et représentation est toujours délicate. Dans le cadre de ce travail, la modélisation est vue comme un problème centré sur l'identification des connaissances alors que la représentation considère ces connaissances comme acquises et étudie les formalismes offrant le meilleur compromis expressivité/efficacité. Nous mettons en évidence deux étapes importantes du processus de représentation. La première étape est guidée par le langage et les fonctionnalités du médiateur. La seconde, qui vise à affiner et à optimiser la représentation des connaissances obtenues à la première étape, est guidée par la façon dont ses fonctionnalités sont mises en œuvre dans l'application.

Ce travail est à rapprocher des travaux sur les ontologies dans le cadre des médiateurs ou du Web sémantique. Cependant, contrairement à ces travaux, on s'intéresse ici au processus même de représentation d'une ontologie étant donné un certain langage. Par ailleurs, ce travail est complémentaire des travaux réalisés en ingénierie des connaissances qui mettent surtout l'accent sur les aspects modélisation. Ces travaux cherchent à déterminer comment trouver les bons concepts, comment trouver les bonnes relations entre concepts, etc. Ils proposent des méthodes ou techniques de construction d'ontologies à partir d'analyse de corpus disponibles (textes (Aussenac *et al*. [2000] Aussenac *et al*. [2002]) (Maedche et Staab [2000]), documents structurés (Goasdoue-Reynaud [1999]), documents semi-structurés) ou des techniques de réutilisation (Clark *et al*. [2000]).

Le plan de l'article est le suivant. Nous décrivons dans la section 2 le langage de représentation des connaissances dans PICSEL. Les sections 3 et 4 sont consacrées à la présentation des deux étapes du processus de représentation d'une ontologie. En section 5, nous concluons et décrivons quelques perspectives. Les exemples cités sont issus du domaine des produits du tourisme, domaine d'expérimentation choisi au sein du projet PICSEL, fruit d'une collaboration avec l'agence de voyage Dégriftour[1].

[1] http://www.degriftour.fr

2. LA REPRESENTATION DES CONNAISSANCES DANS PICSEL

La tâche principale de PICSEL est de déterminer quelles sources d'information sont pertinentes pour répondre à une requête utilisateur. Réaliser cette tâche pose le problème de la reformulation des requêtes en fonction de la description abstraite des sources, encore appelées *vues*. Ce problème a été beaucoup étudié dans le domaine des bases de données relationnelles (Jeffrey & Ullman [1997]). Il est indécidable lorsque les requêtes sont récursives. Dans (Beeri *et al*. [1997]), les auteurs ont montré que la reformulation de requêtes, en termes de vues, peut être indécidable lorsque les vues et les requêtes sont des conjonctions contenant des expressions d'une logique de description, sauf si de sévères restrictions sont imposées. Les choix au niveau de la représentation des connaissances dans PICSEL ont donc été faits sur la base de ces résultats. Il faut pouvoir garantir la *décidabilité* de la réponse aux requêtes et des raisonnements *corrects* et *complets*. Un objectif majeur du projet est en effet d'aboutir à un système opérationnel et efficace. Toutefois, le langage doit aussi rester suffisamment expressif pour pouvoir exprimer des requêtes et décrire le contenu de sources de façon naturelle et précise. Le respect de l'ensemble de ces contraintes a conduit à adopter le formalisme de représentation des connaissances CARIN-$\mathcal{ALN}$ (Levy & Rousset [1998]), un formalisme logique combinant un langage de règles logiques de type Datalog (clauses de Horn sans fonction) avec la logique de description $\mathcal{ALN}$.

2.1. La représentation des connaissances du domaine

Les connaissances du domaine sont exprimées de manière déclarative au travers d' une composante terminologique et d'une composante déductive.

2.1.1. La composante terminologique

La composante terminologique de CARIN-$\mathcal{ALN}$ comprend des définitions et des inclusions de concepts.

Une définition de concept $NC := ConceptExpression$ associe un nom de concept NC à une expression de concept de la logique de description $\mathcal{ALN}$. Les concepts de base d'une terminologie sont ceux qui n'apparaissent pas en partie gauche d'une définition. Un nom de concept NC dépend d'un concept NC' si NC' apparaît dans la définition de NC. Un ensemble de définitions de concept est dit cyclique s'il y a un cycle dans la relation de dépendance des noms de concept. Dans PICSEL, on impose que l'ensemble des définitions de concepts soit non cyclique.

Une inclusion de concepts $C_1 \subseteq C_2$ est une assertion exprimant le fait que le concept C_1 est subsumé par le concept C_2. La vision logique d'une inclusion $C_1 \subseteq C_2$ est l'implication logique $\forall X [C_1(X) \Rightarrow C_2(X)]$. Comme inclusion de concepts, on autorise les inclusions de la forme : $A \subseteq ConceptExpression$, où A est un concept de base, et $A_1 \cap A_2 \subseteq \bot$, où A_1 et A_2 sont des concepts de base.

La logique de description $\mathcal{ALN}$ contient les constructeurs de conjonction ($C_1 \cap C_2$), de restriction de concept (ALL *R C*) qui représente l'ensemble des éléments qui ne sont en relation par le rôle[2] *R* qu'avec des éléments du concept *C*, les restrictions de cardinalité ($\geq$ n *R*), (respectivement ($\leq$ n *R*)), qui représentent l'ensemble des éléments qui sont en relation par *R* avec au moins, (respectivement au plus), n éléments distincts, et la négation $\neg$ (sur les concepts de base uniquement).

2.1.2. La composante déductive

La composante déductive du langage CARIN-$\mathcal{ALN}$ comprend :

a) Un ensemble de règles : $\forall X [q(Y) \Leftarrow p_1(X_1) \wedge \ldots \wedge p_n(X_n)]$ avec $Y = y_1, ...y_r$ et $X_1 = x_{11}$[3], $...x_{1i}$, ..., $X_n = x_{n1}$, ..., x_{nk} permettant de définir des relations q (d'arité quelconque) en fonction d'autres relations $p_1, p_2, ..., p_n$. Les variables de Y sont appelées variables distinguées. Les variables des X_i, distinctes des variables de Y, sont considérées logiquement comme quantifiées existentiellement. On distingue les *relations de base* qui sont les relations qui n'apparaissent pas en conclusion de règles. Parmi les relations de base, certaines – unaires – sont des expressions de concepts, d'autres – binaires – sont des expressions de rôles intervenant dans la composante terminologique.

b) Un ensemble de contraintes d'intégrité : $p_1(X_1) \wedge \ldots \wedge p_m(X_m) \Rightarrow \perp$ où les p_i sont des relations n-aires.

2.2. La description du contenu des sources

Le contenu de chaque source *S* est représenté à partir du vocabulaire $\mathcal{V}_S$ constitué d'autant de noms de relations locales v_i que de relations du domaine dont on sait que la source *S* fournit des instances. Ces relations locales sont appelées des vues. La description du contenu d'une source *S* en terme de ses vues est constituée de :

a) un ensemble $\mathcal{I}_S$ d'implications logiques $v_i(X) \Rightarrow p(X)$ reliant chaque vue à la relation du domaine dont elle peut fournir des instances ;

b) un ensemble $\mathcal{C}_S$ de contraintes sur les instances des vues de la forme : $v \subseteq C$ où C est une expression de concept ou bien $l_1(X_1), \ldots, l_n(X_n) \Rightarrow \perp$, où chaque $l_i(X_i)$ est soit une vue v(X), soit la négation d'une vue (au plus une négation de vue par contrainte).

Exemple 1 : Les implications suivantes expriment le fait que la source SKI permet d'obtenir des instances du concept *chambre* ainsi que des instances des

[2] En logique de description, la notion de rôle a une signification bien particulière. Elle correspond à une relation binaire.

[3] Des variables de noms différents sont considérées comme distinctes, dans l'ensemble du papier.

rôles *nbLitsUnePlaceAssocié*, *nbLitsDeuxPlacesAssocié*, *prestationChambre-Associé* et de la relation (prédicat ordinaire) *stationOu SeTrouveLogement*.

I_{SKI} : SKI_chambre $\Rightarrow$ chambre
SKI_nbLitsUnePlace $\Rightarrow$ nbLitsUnePlaceAssocié
SKI_nbLitsDeuxPlaces $\Rightarrow$ nbLitsDeuxPlacesAssocié
SKI_Prestations $\Rightarrow$ prestationChambreAssocié
SKI_Station $\Rightarrow$ stationOuSeTrouveLogement

Les contraintes suivantes traduisent le fait que les chambres trouvées en accédant à la source SKI sont toutes dans un lieu de résidence situé dans une station de ski et que *nbLitsUnePlace*, *nbLitsDeuxPlaces*, *Prestations*, *Station* concernent les chambres de la source SKI.

C_{SKI} : SKI_chambre $\subseteq$ (ALL lieuResidenceAssocié
lieuResidenceDansStationSki)
SKI_nbLitsUnePlace(x,y), $\neg$ SKI_chambre (x) $\Rightarrow \bot$
SKI_nbLitsDeuxPlaces(x,y), $\neg$ SKI_chambre (x) $\Rightarrow \bot$
SKI_Prestations(x,y), $\neg$ SKI_chambre (x) $\Rightarrow \bot$
SKI_Station(x,y), $\neg$ SKI_chambre (x) $\Rightarrow \bot$

3. L'ETAPE DE REPRESENTATION GUIDEE PAR LE LANGAGE

La démarche adoptée pour construire l'ontologie a consisté à concevoir dans un premier temps le modèle sous forme papier. Ce travail qui, dans PICSEL, a été entièrement manuel, ne sera pas décrit dans cet article qui traite uniquement du processus de représentation mis en œuvre une fois le modèle construit. La représentation du modèle en CARIN-$\mathcal{ALN}$ a été effectuée en utilisant le cadre de représentation du langage qui, en contraignant la représentation, a permis de la guider. Nous décrivons dans les sections qui suivent comment le modèle de l'ontologie a pu être représenté en CARIN-$\mathcal{ALN}$ et comment ce cadre de travail a conduit à affiner et à enrichir le modèle.

3.1. Ce qui est représenté dans la composante terminologique

3.1.1. La définition des concepts et des rôles

Le modèle de l'ontologie comprend une hiérarchie principale de concepts dont la racine est le concept *produit*, qui représente ce qui peut se vendre dans le domaine du tourisme et qui regroupe les logements, trajets, locations de véhicule, stages. Le modèle comprend, par ailleurs, des hiérarchies secondaires disjointes décrivant des catégorisations d'objets de sous-domaines du domaine d'application (lieu, loisir, prestation, service, équipement). Tous ces concepts sont représentés en CARIN-$\mathcal{ALN}$ par des relations unaires.

Chaque concept est défini au travers de ses relations avec d'autres concepts. Pour un concept donné, le modèle précise le concept qui le généralise (concept père dans la hiérarchie) et éventuellement ses propriétés spécifiques ou bien l'ensemble des propriétés nécessaires et suffisantes d'une entité pour appartenir à ce concept. Les propriétés sont représentées en CARIN-$\mathcal{ALN}$ à l'aide de rôles, ou relations binaires, et de constructeurs de restriction de concept (ALL) et de cardinalité ($\geq$, $\leq$). Les exemples ci-dessous illustrent la représentation en CARIN-$\mathcal{ALN}$ des concepts. L'exemple 2 représente la définition du concept *produit* : un produit a un seul prix, une seule date de début, éventuellement des services ou prestations associés. L'exemple 3 représente la définition du concept *activitéSportive* : une *activitéSportive* est une *activité* dont la nature associée est qualifiée de *loisirSportif*. *Produit* et *activitéSportive* sont tous deux des concepts dits définis en logique terminologique. L'exemple 4 illustre l'inclusion du concept de base (non explicitement défini) *équipementCulturel* dans le concept *équipement*.

Selon ce cadre de représentation, un concept « isolé », sans relation avec un autre concept, est dépourvu de sens. Il est éliminé de la représentation.

Exemple 2 : (produit := (and ($\geq$ 1 prixAssocié) ($\leq$ 1 prixAssocié)
(ALL prixAssocié Nombre)
($\geq$ 1 dateDébutAssocié) ($\leq$ 1 dateDébutAssocié)
(ALL dateDébutAssocié Date)
(ALL serviceProduitAssocié Service)
(ALL prestationProduitAssocié Prestation)))

Exemple 3 : (activitéSportive := (and activité
(ALL natureActivitéAssocié loisirSportif)))

Exemple 4 : (équipementCulturel $\subseteq$ équipement)

Les propriétés, ou rôles, issus du modèle de l'ontologie et représentés dans cette étape sont de deux types :

a) les propriétés qu'un utilisateur du serveur souhaiterait préciser lors de l'expression d'une requête ou lors de la description du contenu d'une source. Exemple de requête : une chambre d'hôtel avec 2 lits une place et située dans un hôtel avec piscine,

b) les propriétés nécessaires pour structurer et articuler les concepts entre eux. Ainsi, la propriété *lieuDeRésidenceAssocié* permet de relier un *Logement* (ex. : une chambre d'hôtel) à un *lieuDeRésidence* (ex. : un hôtel). Ce lien est important car les critères de choix d'un logement d'un usager peuvent porter sur les caractéristiques du lieu de résidence (adresse, catégorie, équipements) où le logement est situé.

Ce cadre de travail a facilité la représentation des concepts et des rôles de l'ontologie. D'une part, le mécanisme d'inclusion de concepts permet à un concept d'apparaître comme une spécialisation d'un autre concept et d'hériter de ses propriétés sans qu'il soit nécessaire de le définir plus précisément. Ainsi, si le concept *sportDeMontagne* a été défini comme un sport praticable dans un *lieuAvecMontagne*, on peut introduire des spécialisations de ce concept par

inclusion (*skiAlpin*, *luge*) qui hériteront de ses propriétés. Par ailleurs, le langage rend possible la représentation de points de vue multiples sur un concept qui, de ce fait, hérite des propriétés de l'ensemble des concepts plus généraux. Ainsi, le concept *Bateau* peut être défini comme étant à la fois un *MoyenDeTransport* et un *LieuDeRésidence*. Enfin, toute une hiérarchie de concepts peut être construite sans qu'aucune propriété ne soit explicitée pour aucun d'entre eux. Ainsi, la hiérarchie des équipements décrit des catégories d'équipements de toutes sortes dont les équipements sportifs, parmi lesquels on trouve, entre autres, les piscines qu'on peut spécialiser en *piscineAVague* et *piscineEauDeMer* sans qu'il soit nécessaire de définir ces concepts plus précisément.

Le concepteur est assisté durant tout le processus de représentation de l'ontologie par le « classifieur » qui classifie automatiquement les concepts à partir de leur définition. La hiérarchie résultant de cette classification peut être visualisée. Cela aide à détecter d'éventuelles anomalies, guide pour effectuer des modifications et peut faire émerger des concepts généraux par la mise en évidence de propriétés communes à plusieurs concepts. Par exemple, la création du concept *Logement* a été effectuée en factorisant les propriétés communes des concepts *Chambre* et *Appartement*. Ces deux concepts héritent des propriétés associées au concept général *Logement* et seules leurs propriétés spécifiques leur restent attachées.

Le cadre de travail a également conduit à renommer certains éléments, comme les noms de rôles. En effet, en logique terminologique, un rôle n'est pas défini en fonction des concepts qu'il lie. Il est alors tout à fait possible d'utiliser le nom *serviceAssocié* en tant que rôle de *produit*, de *lieuDeRésidence* ou de *station*. De ce fait, nous avons fait le choix d'utiliser des noms de rôles identiques lorsque les concepts concernés étaient liés par un lien de généralisation-spécialisation et nous avons préféré utiliser des noms de rôles distincts dans le cas contraire, afin d'éviter d'éventuelles ambiguïtés pour l'utilisateur du système.

3.1.2. L'expression de contraintes

Deux formes de contraintes ont été représentées dans la composante terminologique de CARIN-$\mathcal{ALN}$:

a) l'expression d'incompatibilités (ou disjonctions) entre concepts de base (non définis). L'exemple 5 ci-dessous traduit le fait qu'un équipement culturel ne peut pas être un équipement sportif.

Exemple 5 : équipementCulturel $\cap$ équipementSportif $\subseteq \bot$

b) l'expression de contraintes de typage sur les rôles par le biais du constructeur de restriction de concepts (cf. 3.1.1.). Dans l'exemple 2 définissant le concept *produit*, la restriction (ALL dateDébutAssocié Date) signifie que le rôle *dateDébutAssocié* ne relie des éléments de type *produit* qu'à des éléments de type *Date*.

3.2. Le recours aux règles

L'emploi de règles déductives permet de définir des propriétés sur les concepts par des prédicats déductibles n'appartenant pas à la composante terminologique. Suivant les cas, ce recours aux règles est une nécessité (cf. cas 1 à 4) ou une facilité d'écriture (cf. cas 5).

Cas 1. Pour exprimer des relations autres que des relations unaires et binaires. La règle R1 définit la relation de nom *VolAR* et d'arité 4 à partir d'un certain nombre de relations de base. Elle exprime qu'un vol aller-retour (*VolAR*) est composé de deux vols consécutifs dans le temps. Le premier vol part d'une ville « villeDép » à la date « dateDép1 » et atterrit dans la ville « villeArr ». Le second vol part d'une ville « villeArr » à la date « dateDép2 » et atterrit à « villeDép ».

R1 : VolAR(villeDép, dateDép1, villeArr, dateDép2) $\Leftarrow$ Vol(v1), lieuDépart(v1, villeDép), lieuArrivée(v1, villeArr), dateDépart(v1, dateDép1), vol(v2), lieuDépart(v2, villeArr), lieuArrivée(v2, villeDép), dateDépart(v2, dateDép2), Antérieure (dateDép1, dateDép2).

Cas 2. Pour construire une définition de relation disjonctive. Les règles R2 et R3 traduisent qu'un produit pour jeune (*produitJeune*) est un produit auquel on associé soit au plus un service, soit des services bon marché. Cette relation ne peut être définie dans la partie terminologique du langage du fait de l'absence du constructeur de disjonction $\vee$.

R2 : ProduitJeune(x) $\Leftarrow$ produit(x), ($\leq$ 1 produitServiceAssocié)

R3 : ProduitJeune(x) $\Leftarrow$ produit(x),(ALL produitServiceAssocié bonMarché)

Cas 3. Pour exprimer des contraintes « non exclusives ». Le constructeur de restriction de concepts (ALL R C) permet d'exprimer des contraintes sur le domaine des concepts en relation avec d'autres, mais ces contraintes sont « exclusives ». Ainsi, dans la définition suivante « lieuAvecMontagne := (and lieu (ALL loisirPraticable sportDeMontagne))», la restriction se lit « tous les loisirs praticables dans un lieuAvecMontagne sont des sportDeMontagne », ce qui exclut la natation ou le tennis. Si on souhaite représenter qu'il est possible de faire du ski et de la natation dans un lieuAvecMontagne équipé d'une piscine, on ne doit pas utiliser de restriction de concept, mais des règles. Les loisirs praticables dans un lieu peuvent y être définis comme dépendant à la fois des caractéristiques physiques du lieu considéré (cf. R4 et R5) et de ses équipements sportifs (cf. R6).

R4 : loisirPraticable(l, s) $\Leftarrow$ lieuAvecMontagne(l), sportDeMontagne(s).

R5 : loisirPraticable(l, s) $\Leftarrow$ lieuAvecPlage(l), sportNautique(s).

R6 : loisirPraticable(l, s) $\Leftarrow$ lieuAvecPiscine(l), natation(s).

Cas 4. Pour exprimer une relation inverse. Dans PICSEL, les définitions de concepts sont non cycliques et les rôles sont orientés. Par exemple, le rôle *situationAssocié* relie le concept *lieuDeRésidence* au lieu dans lequel il est situé. On s'interdit d'introduire, dans la partie terminologique, le rôle inverse qui donnerait les *lieuDeRésidence* se trouvant dans un lieu. En revanche, cette

définition est possible par une règle (cf. R7).

R7 : SeTrouvantDansLieu(l, r) ⇐ lieuDeResidence(r), situationAssocié(r, l), lieu(l).

Cas 5. Pour traduire un raccourci d'enchaînement de rôles. Une règle peut associer directement à un concept une propriété qui lui est déjà associée indirectement, par enchaînement de plusieurs rôles. Le but est d'éviter les redondances dans la composante terminologique. Ainsi, la règle R8 permet de dériver le prédicat *StationOuSeTrouveLogement* qui associe directement à *Logement*, à partir des propriétés présentes dans la composante terminologique, le nom de la station de ski où se trouve son lieu de résidence associé.

R8 : StationOuSeTrouveLogement(l,n) ⇐ logement(l),

lieuDeResidenceAssocié(l,r), stationLieuResidenceAssocié(r, st),

stationSki(st), nomAssocié(st,n).

3.3. L'utilisation de contraintes

Comme on l'a vu en 3.1.2., l'expression de contraintes de disjonction entre concepts de base (non définis) se fait en CARIN-$\mathcal{ALN}$ dans la composante terminologique. Les contraintes portant sur d'autres types de prédicats doivent être représentées par des contraintes d'intégrité (cf. 2.1.2).

3.3.1. Contraintes traduisant des dépendances fonctionnelles

La contrainte ci-dessous exprime le fait qu'un numéro de téléphone est propre à un seul lieu de résidence.

Exemple 6 : NumTél(x_1,y), lieuDeRésidence(x_1), NumTél(x_2,y), lieuDeRésidence(x_2), $x_1 \neq x_2 \Rightarrow \perp$

3.3.2. Contraintes de typage sur les arguments de prédicats

Les contraintes de typage s'écrivent P(X), $\neg C_i(x_i) \Rightarrow \perp$ avec X = x_1, x_2, …, x_i, …x_n. Elles permettent de préciser le domaine de valeurs (ensemble des éléments de C_i) de chaque argument x_i de la relation P. Ainsi, selon l'exemple 7, la relation *serviceAssocié* établit toujours un lien avec un service.

Exemple 7 : ServiceAssocié (x,y), ¬ service(y) ⇒ ⊥

Les contraintes de typage peuvent remplacer une restriction de concept dans une définition. Ainsi, lorsqu'un rôle est utilisé dans la définition de plusieurs concepts, avec la même restriction de concept (ex. : (ALL serviceAssocié service)), l'expression d'une contrainte (cf. exemple 7) évite de répéter la restriction de concept dans chaque définition. Lorsque les restrictions de concept sont différentes, le concept apparaissant dans la contrainte doit être un concept généralisant tous les concepts restreints. Dans ce cas, toutefois, la contrainte ne remplace pas les restrictions de concept qui sont plus précises. Ces dernières sont conservées, la contrainte est néanmoins utile et s'appliquera en l'absence de restriction plus précise (cf. exemple 8).

Exemple 8 :

ChambreD'Hôtel := (and Chambre (ALL lieuDeRésidenceAssocié Hôtel))

CabineBateau := (and Chambre (ALL lieuDeRésidenceAssocié Bateau))

Le premier concept qui généralise à la fois *Hôtel* et *Bateau* est *LieuDeRésidence*. On peut donc écrire la contrainte qui suit :

lieuDeRésidenceAssocié (x,y), $\neg$ lieuDeRésidence(y) $\Rightarrow \bot$.

Cette contrainte s'applique chaque fois que le rôle *lieuDeRésidenceAssocié* est utilisé dans une définition de concept sans qu'aucune restriction de concept ne soit indiquée.

3.3.3. Contraintes d'exclusion entre prédicats quelconques

Ces contraintes sont de la forme $p_1(X_1), \ldots, p_n(X_n) \Rightarrow \bot$ avec $X_1 = x_{11}, \ldots x_{1i}, \ldots, X_n = x_{n1}, \ldots, x_{nk}$. Elles sont utiles pour exprimer des contraintes sémantiques sur le domaine d'application. Ainsi, l'exemple 9 exprime qu'il n'y a pas de vol direct entre deux pays en conflit. Elles servent également au traitement des exceptions. Dans l'exemple 10, *CabineBateau* est une spécialisation du concept *Chambre* et hérite à ce titre de toutes ses propriétés associées, donc, entre autres, des prestations. La contrainte de l'exemple 10 traduit le fait que *Terrasse* fait exception à cette règle car les terrasses ne font pas partie des prestations associées à une cabine de bateau.

Exemple 9 : Pays(p_1), Pays(p_2), EnConflit(p_1,p_2), TrajetReliant(t, p_1, p_2), VolDirect(t) $\Rightarrow \bot$. *Pays* et *VolDirect* sont des concepts, *EnConflit* est un rôle, *TrajetReliant* est un prédicat.

Exemple 10 : CabineBateau(l), prestationChambreAssocié(l, p), Terrasse(p) $\Rightarrow \bot$. *CabineBateau* et *Terrasse* sont des concepts, *prestationChambreAssocé* est un rôle.

3.4. Les choix de représentation dus aux limites du langage

Le langage CARIN-$\mathcal{ALN}$ a été choisi pour garantir l'obtention d'un système efficace. Ce langage s'avère, en règle générale, relativement riche, notamment parce qu'il s'agit d'un langage hybride combinant à la fois logique de description et règles Datalog. Lorsqu'une connaissance ne peut s'exprimer à l'aide de la composante terminologique, il est souvent possible, comme le montrent les sections qui précèdent, de la représenter sous forme de règle ou de contrainte. Néanmoins, d'autres solutions ont parfois dû être imaginées pour pallier certaines limites du langage, par exemple l'absence du quantificateur existentiel $\exists$ et de la négation $\neg$.

3.4.1. CARIN-$\mathcal{ALN}$ ne permet pas l'utilisation du quantificateur existentiel

Ce manque a pu être comblé par l'utilisation de multiples rôles spécialisés. Ainsi pour définir une StationFamiliale comme une station équipée à la fois d'une garderie enfants et d'un centre médical, deux rôles distincts :

garderieSurPlaceAssocié et *médecinSurPlaceAssocié* (cf. exemple 11) ont été introduits alors que le seul rôle *serviceAssocié* aurait pu suffire s'il avait été possible de l'utiliser dans des expressions comportant le quantificateur ∃ du type : (∃ *serviceAssocié* garderieEnfants) et (∃ serviceAssocié Médecin) en concevant *garderieEnfants* et *Médecin* comme des concepts disjoints.

Exemple 11 :

StationFamiliale := (and Station
(≥ 1 garderieSurPlaceAssocié) (ALL garderieSurPlaceAssocié garderieEnfants)
(≥ 1 médecinSurPlaceAssocié) (ALL médecinSurPlaceAssocié Médecin))

3.4.2. La négation autorisée dans CARIN-$\mathcal{ALN}$ est très restreinte

En particulier, elle ne peut pas s'appliquer aux concepts définis. Nous avons contourné cette limitation en multipliant les noms de concepts et en exploitant les contraintes d'intégrité. Ainsi, dans l'exemple 12, le concept *lieuSansMontagne* a été introduit de façon à exclure la pratique d'un *SportDeMontagne* ailleurs que dans un *lieuAvecMontagne*. *lieuAvecMontagne* et *lieuSansMontagne* sont par ailleurs déclarés disjoints par une contrainte d'intégrité (cf. exemple 12 b).

Exemple 12 :

lieuSansMontagne(l), sportDeMontagne(s), loisirPraticable(l, s) ⇒ ⊥
lieuAvecMontagne(x), lieuSansMontagne(x) ⇒ ⊥

A l'issue de cette étape, nous disposons d'une première version opérationnelle de l'ontologie composée d'environ 200 concepts et 300 rôles. Celle-ci respecte les contraintes du langage et le modèle de l'ontologie préalablement construit. Son contenu a été défini en phase de modélisation et complété dans cette première étape de représentation compte tenu des fonctionnalités du médiateur à mettre en place, fonctionnalités pour lesquelles on ignore, à ce stade, la façon dont elles seront mises en œuvre. La représentation obtenue est directement utilisable par le moteur de requêtes et peut être le support de l'interface utilisateur. Nous décrivons alors en section 4 comment l'ontologie a été optimisée du fait de la prise en compte du rôle des connaissances de l'ontologie dans la mise en oeuvre des fonctionnalités.

4. L'ETAPE D'OPTIMISATION DE LA REPRESENTATION

La représentation obtenue a été utilisée comme support de l'interface guidant l'utilisateur dans l'expression de requêtes et, également, par le moteur pour calculer des plans de requêtes. Nous décrivons, dans les sections 4.1 et 4.2, l'impact de la mise en œuvre de ces deux modules sur l'ontologie.

4.1. L'ontologie, support de l'interface d'aide à l'expression de requêtes

L'ontologie met à la disposition des utilisateurs tout le vocabulaire d'un domaine d'application, concepts élémentaires et rôles associés, et permet d'exprimer toutes sortes de requêtes portant sur des concepts « complexes », généralement non représentés dans l'ontologie, mais qu'il est possible de définir à l'aide du vocabulaire disponible.

L'ontologie a d'abord été utilisée comme support d'une première interface composée d'un module de navigation et d'un module d'édition. Le module de navigation permet à l'utilisateur de visualiser la hiérarchie des concepts, d'avoir accès à la liste des concepts et des rôles de l'ontologie et pour chaque concept, de consulter sa définition, ses généralisants directs, ses spécialisations et les rôles qui lui sont associés. Le module d'édition aide à la formulation de requêtes. L'utilisateur construit sa requête, dans le langage CARIN, en sélectionnant concepts, rôles et constructeurs au sein de listes déroulantes. Cette interface permet ainsi de naviguer dans l'ontologie et de consulter le vocabulaire utilisable. Elle est très utile pour aider à comprendre la signification de l'ensemble des concepts et rôles représentés et pour guider dans le choix des termes à utiliser. En revanche, lorsque l'ontologie est très détaillée et très riche, son utilisation peut devenir complexe, d'autant plus qu'elle exige une bonne maîtrise du pouvoir d'expression du langage de requêtes (CARIN).

Face à ces difficultés, une deuxième interface a été développée (Reynaud-Safar [2002]) où l'utilisateur est moins libre de ses choix dans la construction de sa requête, davantage guidé, et surtout soulagé de toutes les contraintes liées au langage de requêtes. L'approche consiste à prédéfinir des requêtes portant sur les thèmes le plus souvent recherchés (*SéjourAuSoleil*, *SéjourWeekEnd*, *CourtSéjour*, etc.) et à les proposer aux utilisateurs du système. Ces thèmes ne correspondent pas aux sous-concepts de la racine *produit* de la hiérarchie, mais à des combinaisons de produits de tourisme élémentaires, définissant des concepts « complexes », par exemple un stage auquel s'ajoute un trajet (aller-retour entre la ville la plus proche du domicile et le lieu du stage).

Chaque thème est représenté en CARIN par une relation définie par une règle (cf. exemple 13) dont le corps est une conjonction de concepts et de rôles de la partie terminologique et, éventuellement, de prédicats ordinaires eux-mêmes définis par des règles.

Exemple 13 :

SéjourAuSoleil(s,p) ⇐CombinéSéjour(s), LogementAssocié(s,l), Logement(l),
LieuRésidenceAssocié(l,r),LieuDeRésidence(r),SituéDans(r,p),
LieuAuSoleil(p).

CombinéSéjour, *Logement*, *LieuDeRésidence*, *LieuAuSoleil* sont des concepts. *LogementAssocié*, *LieuRésidenceAssocié*, *SituéDans* sont des rôles.

Dès qu'un thème est sélectionné, l'interface exploite le corps de la règle définissant le prédicat correspondant : elle guide l'utilisateur en lui proposant de spécialiser les concepts intervenant dans la définition ou d'introduire de

nouvelles propriétés sur ces concepts. Chaque choix de l'utilisateur est répercuté en CARIN dans la requête initiale.

Cette approche a conduit à recenser les thèmes souvent recherchés et à enrichir l'ontologie pré-existante par l'ajout de prédicats correspondants. Cet enrichissement devrait s'accroître encore prochainement lors de la mise en place d'une interface similaire pour aider à la description du contenu des sources.

4.2. L'ontologie utilisée pour calculer des plans de requêtes

Le calcul des plans de requêtes passe par la réécriture des requêtes des utilisateurs en termes de vues sur les sources. Cette reformulation s'effectue en utilisant les connaissances exprimées dans la composante terminologique et les règles de la composante déductive. Les contraintes n'interviennent pas directement dans la reformulation, mais uniquement pour tester la satisfiabilité des réécritures.

Ainsi, à partir d'une requête exprimée sous la forme d'une conjonction d'atomes, le moteur alterne des phases de réécriture de chaque atome en chaînage arrière (dites phases d'expansion) avec des phases de vérification de contraintes en chaînage avant jusqu'à atteindre *toutes* les réécritures possibles et satisfiables ne contenant plus que des vues (les réécritures terminales). Sans entrer dans le détail du fonctionnement d'un moteur terminologique, il faut souligner que dans les phases d'expansion, celui-ci essaie toutes les possibilités : la réécriture directe par une vue, le remplacement d'un prédicat ordinaire par le corps des différentes règles qui le définissent et celui d'un concept terminologique par sa définition. De plus, pour chaque concept terminologique défini, le moteur « déplie » complètement le concept (le remplace par sa définition et remplace récursivement tous les concepts intervenant dans celle-ci jusqu'à atteindre les concepts de base), puis construit, au sein de ce « déplié », tous les regroupements possibles de concepts et de rôles susceptibles d'être réécrits par une vue. Ce dernier mécanisme est très coûteux. Dans le cas le pire, il est exponentiel dans la taille des expressions terminologiques utilisées dans la requête à reformuler. Ainsi, plus les définitions contiennent de rôles et de restrictions de concept, plus le processus de reformulation est coûteux. Le modèle de l'ontologie construit étant très fin, de très nombreux rôles caractérisent chaque concept. Leur représentation s'avère alors être pénalisante en temps de calcul.

Pour pallier ce problème, nous avons adopté une solution basée sur le fait que tous les rôles ne sont pas utiles dans toutes les fonctionnalités du médiateur. La présence de rôles est essentielle, d'une part, pour la construction de l'ontologie où le classifieur se base sur les rôles intervenant dans la définition d'un concept nouveau pour le rattacher à telle ou telle catégorie, d'autre part, pour la mise à disposition de l'utilisateur d'un maximum de caractéristiques qui peuvent l'aider à mieux interpréter la signification des concepts représentés et aussi l'intéresser pour exprimer précisément ce qu'il recherche. En revanche, les rôles ne sont pas tous utiles pour le calcul des plans de requêtes. En nous basant sur cet état de fait, nous avons cherché à optimiser la représentation de l'ontologie, le but étant de limiter la taille du « déplié » (et donc le nombre de regroupements possibles).

Pour cela, nous faisons l'hypothèse suivante : les concepts au sommet des hiérarchies sont des concepts génériques, abstraits (produit, lieu, loisir, prestation, service, lieuDeRésidence, etc.). A priori, une requête porte sur la recherche d'instances de concepts non abstraits. Ainsi, un utilisateur ne cherche pas une instance de produit mais une instance de logement ou de trajet, etc., vérifiant des propriétés particulières. Les propriétés définies sur un concept abstrait ne sont en réalité utilisées que sur ses spécialisations. Notre approche consiste alors à utiliser l'heuristique suivante : « *Tous les concepts racines des différentes hiérarchies de l'ontologie deviendront des concepts non définis dans l'ontologie optimisée, de même que leurs concepts fils lorsqu'ils ne dépendent que de concepts racines non définis (non définis initialement dans l'ontologie ou du fait de l'application de cette heuristique)*». Ainsi, les concepts, fils de concepts racines, qui ne dépendent que de concepts racines non définis deviennent aussi non définis. En effet, leurs rôles établissent des relations avec des concepts racines de hiérarchie, donc génériques. Il s'agit de propriétés représentées dans un souci de structurer les concepts entre eux, de les articuler (exemple : la propriété *lieuDeRésidenceAssocié* reliant *logement* à *lieuDeRésidence*). Après application de l'heuristique, ces propriétés n'apparaîtront que dans les définitions des spécialisations des concepts considérés.

En revanche, la présence de définitions de concepts peut s'avérer très utile pour le moteur de requêtes lorsqu'elles permettent de déterminer des disjonctions entre concepts. Ainsi, dans l'exemple 14, les deux concepts définis, *chambreD'Hôtel* et *chambreD'Hôte*, seront considérés comme disjoints par le moteur car les restrictions de concepts qui interviennent dans leur définition renvoient aux concepts, *Hôtel* et *Gîte*, définis comme étant disjoints dans la terminologie. La présence du rôle *LieuDeRésidenceAssocié* dans la définition des concepts est donc essentielle dans ce cas.

Exemple 14 :

chambreD'Hôtel := (and Chambre (ALL lieuDeRésidenceAssocié Hôtel))
chambreD'Hôte := (and Chambre (ALL lieuDeRésidenceAssocié Gîte))

Tous ces constats nous ont conduit à restructurer l'ontologie, avec un double objectif : préserver toute la richesse du modèle et maintenir l'efficacité du moteur. En considérant, qu'a priori, le classifieur n'est plus utilisé une fois l'ontologie construite, nous avons choisi de ne pas conserver dans la composante terminologique les rôles des concepts initialement définis, transformés en concepts non définis lors du processus d'optimisation et de les représenter par des contraintes de typage. Ces derniers restent néanmoins toujours accessibles pour l'expression des requêtes et pour la description des sources, ils sont consultables par l'utilisateur mais ils ne sont pas exploités dans les phases d'expansion.

Cette nouvelle représentation nous permet d'obtenir une hiérarchie dans laquelle de nombreux concepts n'ont plus de définition et sont décrits par de simples inclusions de concepts. Les concepts définis apparaissent surtout dans le bas de la hiérarchie (cf. exemple 15). Grâce à cette nouvelle structuration, le coût du calcul des plans de requête devient acceptable.

Exemple 15 : Dans la nouvelle hiérarchie, les concepts *produit*, *logement*, et *chambre* n'ont plus de définition mais les concepts *chambreD'Hôtel*, *chambreD'Hôte* et *cabineBateau* restent définis comme des spécialisations de chambre (cf. exemple 14) : chambre $\subseteq$ logement $\subseteq$ produit

	Format de représentation	**Nature des connaissances représentées**	**Calcul des plans de requêtes**		**Inter-face**
			Phase de réécri-ture	**Phase de vérifi-cation**	
Compo-sante termino-logique	Définitions de concepts *NC* :=*Concept-Expression*	Concepts définis			
	Inclusions de concepts $C_1 \subseteq C_2$	Concepts de base			
	Disjonctions de concepts $A_1 \cap A_2 \subseteq \perp$	Disjonctions entre concepts de base			
Compo-sante déductive	Contraintes d'intégrité	Dépendances fonctionnelles			
		Typage des arguments des prédicats			
		Contraintes sémantiques			
	Règles	Relations n-aires (n>2)			
		Expression de disjonctions			
		Raccourci d'enchaînement de rôles			
		Contraintes « non exclusives »			
		Relations inverses			
		Requêtes prédéfinies			

TABLEAU 1 — Récapitulatif des différentes catégories de connaissances représentées dans l'ontologie et, en gris, indication des étapes des traitements au sein desquelles elles sont utilisées.

Le tableau 1 récapitule les différentes catégories de connaissances représentées dans l'ontologie et les étapes où elles sont exploitées (en gris dans le tableau). Cette nouvelle organisation des connaissances donne actuellement de bons résultats mais les tests effectués jusqu'à présent ne sont pas suffisants pour fournir une mesure quantitative des gains obtenus. Par ailleurs, la phase

d'optimisation de l'ontologie peut certainement être encore améliorée. Le travail présenté doit être vu comme le résultat d'un premier travail d'optimisation d'une ontologie, travail que nous envisageons de poursuivre.

5. CONCLUSION ET PERSPECTIVES

Dans cet article, nous avons décrit le processus de représentation d'une ontologie au sein du médiateur PICSEL. Deux étapes ont été distinguées. La première étape consiste à construire une première version opérationnelle de l'ontologie respectant le modèle préalablement défini ainsi que les contraintes du langage CARIN-$\mathcal{ALN}$. Nous avons décrit, dans cet article, comment cette première étape a été mise en œuvre pour construire une ontologie d'un domaine d'application réel, le domaine des produits du tourisme. La seconde étape consiste à utiliser cette première version de l'ontologie pour le calcul des plans de requêtes et comme support de l'interface utilisateur. Cette deuxième étape a conduit à affiner la représentation de l'ontologie par l'ajout de prédicats supplémentaires qui se sont révélés utiles lors de l'implémentation de l'interface et par une réorganisation des connaissances représentées. En effet, désirant préserver toute la richesse du modèle tout en visant l'obtention d'un système efficace, une organisation des connaissances basée sur leur rôle dans l'application s'est avérée être une solution tout à fait satisfaisante.

Distinguer ces deux étapes est important car chacune a un objectif différent. Dans la première, il s'agit avant tout de représenter le modèle de l'ontologie en analysant la nature des connaissances à représenter. Dans la seconde, on recherche le meilleur formalisme de représentation compte tenu de la mise en œuvre des fonctionnalités du médiateur. Si l'on reprend une expression bien connue en ingénierie des connaissances, la première étape se situe plus au niveau connaissances, alors que la seconde est plus au niveau symbolique. Le processus de représentation des connaissances, souvent considéré comme un simple processus de traduction dans un formalisme donné, s'avère être en fait un processus complexe pour lequel des conseils méthodologiques font généralement défaut. Notre objectif, au travers de cet article, est de contribuer à l'énoncé de conseils de ce type, destinés aux futurs concepteurs d'ontologies exploitées par des médiateurs.

Toutefois, malgré l'explicitation et la description du processus de représentation d'une ontologie PICSEL, construire une ontologie s'avère très long et difficile. Ce problème représente un obstacle scientifique important pour le développement des approches médiateurs. Dans le cadre du projet PICSEL II, nous nous donnons pour objectif d'automatiser la construction d'ontologies factorisant et abstrayant un ensemble de sources d'information XML. Il s'agira, à partir d'une première ébauche simple d'une ontologie, de l'affiner et de l'enrichir automatiquement, en exploitant le contenu des sources d'information disponibles.

6. REMERCIEMENTS

Les auteurs remercient Marie-Christine Rousset, responsable du projet PICSEL, pour ses encouragements dans l'écriture de cet article, ses remarques et conseils avisés ainsi que François Goasdoué, auteur du moteur de requêtes, qui par sa disponibilité, nous a permis d'optimiser la représentation de l'ontologie.

7. RÉFÉRENCES

AUSSENAC N., BIEBOW B., SULZMAN S. (2000), «Revisiting Ontology Design: A method Based on Corpus Analysis», *EKAW 2000*, LNAI 1937, 172-188.

AUSSENAC N., BIEBOW B., SULZMAN S. (2002), «Modélisation du domaine par une méthode fondée sur l'analyse de corpus», - dans cet ouvrage.

BEERI C., LEVY A. Y. et ROUSSET M.-C. (1997), «Rewriting queries using views in description logics», *Proceedings of the Sixteenth Symposium on principles of Database Systems* (PODS'97).

CLARK P., THOMPSON J., HOLMBACK H. et DUNCAN L. (2000), «Exploiting Thesaurus-Based Semantic net for Knowledge-Based Search», AAAI, 988-995.

DUSCHKA O., GENESERETH M. (1997), «Query planning in Infomaster», *Proceedings of the 1997 ACM Symposium on Applied Computing*, San Jose, USA.

FENSEL D., HORROCKS I., VAN HARMELE F., DECKER S., ERDMANN M., KLEIN M. (2000), «OIL in a nutshell», *In ECAI Workshop Notes – Applications of Ontologies and Problem-Solving Methods*, 4-1 – 4-12.

GARCIA-MOLINA H., PAPAKONSTANTINOU Y., QUASS D., RAJARAMA A., SAGIV Y., ULLMAN, J., WIDOM J. (1997), «The TSIMMIS project: Integration of heterogeneous information sources», *Journal Intelligent Information Systems*, 8(2), 117-132.

GOASDOUE F., LATTÈS V., ROUSSET M.-C. (2000), «The Use of CARIN Language and algorithms for information integration: the PICSEL Project», *International Journal of Cooperative information Systems*, 4(9), 383-401.

GOASDOUE F. et REYNAUD C. (1999), «Modeling Information Sources for Information Integration», *EKAW 99*, LNAI 1621 Springer-Verlag, 121-138.

HENDLER J. et McGUINNESS D. L. (2000), «The DARPA agent markup language», *IEEE Intelligent Systems*, 6(15), 72-73.

JEFFREY D., ULLMAN J. (1997), «Information integration using logical views», *Proceedings de ICDT'97*.

LEVY A. Y., ROUSSET M. – C. (1998), «Combining Horn rules and description logics in CARIN», *Artificial Intelligence*, 104, 165-209.

LEVY A., RAJAMARAN A., ORDILLE J. (1996), «Querying heterogeneous information sources using source descriptions», *Proceedings of the Int. Conf. On Very Large Data Bases (VLDB)*, 251-262.

MAEDCHE A. et STAAB S. (2000), «Mining Ontologies from Text», *EKAW 2000*, LNAI 1937, 189-202.

REYNAUD C. et SAFAR B. (2002), «Aide à la formulation de requêtes dans un médiateur», *RFIA 2002*, Angers.