Une approche basée agent pour la découverte de services Web

Berdjouh Chafik*, Kazar Okba**

* Centre de Formation Professionnelle El-Meghaier
Wilaya El-OUED, ALGERIE
Berdjouh2006@yahoo.fr

**Département d'informatique
Faculté des sciences et sciences de l'ingénieur
Université Mohamed Khider 07000 Biskra, ALGERIE
kazarokba@yahoo.fr

Résumé. Les services Web sont des technologies émergentes et prometteuses pour le développement, le déploiement et l'intégration d'applications Internet. Ils sont basés sur trois briques principales que sont SOAP (Simple Object Access Protocol), WSDL (Web Service Description Langage) et UDDI (Universal Description, Discovery and Integration). Le langage utilisé qui sous-tend ces protocoles est XML (eXtensible Markup Language), ce qui rend les Web Services indépendants des plates-formes et des langages de programmation. Ils sont devenus un moyen très efficace dans l'interopérabilité des systèmes. Le besoin d'introduire la sémantique dans les services Web se fait sentir, afin d'automatiser les différentes phases de leur cycle de vie, en l'occurrence la phase de découverte.

Le concept des services Web sémantiques, est le fruit de la convergence du domaine des services web avec le Web sémantique, en effet son ultime objectif est de rendre les services web plus accessibles à la machine en automatisant les différentes tâches qui facilitent leur utilisation. Dans ce travail, on étudie la problématique de découverte sémantique des services en proposant une méthode basée sur les agents.

Abstract. Web services are emerging and promising technologies for the development, deployment and integration of Internet applications. They are based on three main bricks that are SOAP (Simple Object Access Protocol), WSDL (Web Service Description Language) and UDDI (Universal Description, Discovery and Integration). The language used behind these protocols is XML (eXtensible Markup Language), which makes Web services independent of platforms and programming languages. They have become very effective in the interoperability of systems. The need to introduce semantics in Web services is felt to automate the different phases of their life cycle, namely the discovery phase.
The concept of semantic web services, is the result of convergence in the field of web services with the Semantic Web, in fact its ultimate goal is to make web services more accessible to the machine by automating tasks that facilitate their use. In this work, we study the problem of semantic discovery of services by providing a method that is based on agents.

Mots-Clés : service Web, systèmes multi-agents, Web sémantique

Keywords .: Web service, multi-agents system, semantic web

1 Introduction

De nos jours, le Web n'est plus simplement un énorme entrepôt de texte et d'images, son évolution a fait qu'il est aussi un fournisseur de services. La notion de "service Web" désigne essentiellement une application mise à disposition sur Internet par un fournisseur de services, et accessible par les clients à travers des protocoles Internet standard. Par essence, les services Web sont des composants logiciels autonomes et auto-descriptifs et constituent par ce fait un nouveau paradigme pour l'intégration d'applications.

Actuellement, les services Web sont mis en oeuvre au travers de trois technologies standards : WSDL, UDDI et SOAP. Ces technologies facilitent la description, la découverte et la communication entre services. Cependant, cette infrastructure de base ne permet pas encore aux services Web de tenir leur promesse d'une gestion largement automatisée. Cette automatisation est pourtant essentielle pour faire face aux exigences de passage à l'échelle et de la volonté de réduire les coûts de développement et de maintenance des services. Fondamentalement, elle doit s'accommoder d'un moyen pour décrire les services Web d'une manière compréhensible par une machine.

Le Web sémantique [1] est une vision du Web dans laquelle toute information possède une sémantique compréhensible par une machine. Appliqués aux services Web, les principes du Web sémantique doivent permettre de décrire la sémantique de leurs fonctionnalités, et les raisonnements induits constituent par conséquent une proposition d'automatisation des différentes tâches de leur cycle de vie. La combinaison des technologies des services Web et du Web sémantique a mené au concept des services Web sémantiques.

La découverte des services Web représente un axe de recherche émergent. Au début, la découverte est faite au niveau du registre UDDI, elle est basée essentiellement sur la recherche syntaxique des descriptions WSDL des services Web. Mais avec le développement des technologies du Web sémantique, les techniques de découverte sont devenues essentiellement sémantiques. Cette sémantique est apportée grâce aux ontologies une des technologies importantes du Web sémantique. Ainsi, des agents logiciels peuvent êtres développés afin de raisonner sur ces ontologies rendant la découverte des services Web dynamique et automatique.

Dans ce travail, nous proposons une approche de découverte des services Web sémantiques en utilisant la technologie agent et les ontologies.

2 Technologies émergentes

2.1 Web sémantique et ontologie

Le Web sémantique [1] envisage le développement de l'Internet actuel vers un Web où les données disponibles sont enrichies avec leur sémantique. Dans la vision de Tim Berners Lee, Web sémantique est structuré en différentes couches :

- Une couche syntaxique (XML),
- Une couche de méta-données (RDF/RDFS),
- Une couche sémantique (les langages d'ontologie)
- Une couche logique (raisonnement automatique)
- Une couche de validation et de preuve (proof)

Le terme ontologie est initialement emprunté de la philosophie signifiant "explication systématique de l'existence". Une ontologie est similaire à un dictionnaire ou un glossaire mais avec une structure détaillée et grande qui permet aux machines de traiter son contenu. Bertrand [2] définit l'ontologie comme "Il s'agit de représentations formelles d'un domaine de connaissance sous la forme de terminologies dotées de relations sémantiques.".

Dans le Web sémantique, l'ontologie permet à l'utilisateur lors d'une recherche sur le Web d'accéder non seulement aux documents liés aux mots clés de la requête, mais aussi à ceux qui sont liés ontologiquement (sémantiquement) à ces derniers, ce qui rend la recherche encore plus pertinente. Elle a pour but de décrire des concepts et les relations qui les lient entre eux, et avec des règles de déduction les rendre plus compréhensibles et utilisables par les différents agents (humains ou logiciels).

2.2 Le langage OWL-S

Appelé DAML-S dans ses premières versions [3], le langage OWL-S (Ontology Web Language for Service) [4] basé sur DAML+OIL a pour objectif d'ajouter des descriptions sémantiques aux services Web (en plus de leur description syntaxique WSDL). OWL-S a pour objectif de fournir une plus grande expressivité en permettant la description des caractéristiques des services afin de pouvoir raisonner dessus dans le but de découvrir, invoquer, composer et gérer les services Web de façon la plus automatisée possible.

Le langage OWL-S organise la description d'un service en trois zones conceptuelles : le profil (Profile), le modèle de processus (ProcessModel) et les liaisons avec le service (Grounding).

D'une manière générale, la classe ServiceProfile donne les informations nécessaires à un agent pour publier ou découvrir un service. Les profils des services sont généralement organisés sous la forme de taxonomies, qui constituent le premier niveau de discrimination lors de la recherche d'un service Web spécifique [5].

2.3 L'apport des systèmes multi-agents

Les systèmes multi agents (SMA) et les agents autonomes fournissent une nouvelle méthode pour analyser, designer et implémenter des applications sophistiquées car ils font partie du domaine IAD (Intelligence Artificielle Distribuée) [6] en bénéficiant aussi d'autres disciplines comme les sciences cognitives, sociologie, et psychologie sociale.

Aujourd'hui, la plupart des applications nécessitent de distribuer des tâches entre des "entités" autonome (ou semi-autonome) afin d'atteindre leurs objectifs d'une manière optimale. Puisque les approches classiques sont en général monolithiques et leur concept d'intelligence est centralisé, les applications actuelles sont établies à base de système multi-agents.

3 Architecture de Découverte de Services Proposée

L'architecture proposée est une extension de l'architecture orientée services SOA (Service Oriented Architecture). Cette architecture est basée sur des agents pour la découverte de services Web.

Cette architecture (voir figure 1) intègre des composants logiciels et exploitant une ontologie de domaine qui est utilisée lors de la phase de découverte des services Web, elle facilite la découverte automatique de services puisqu'elle permet d'affiner le processus de recherche qui met en correspondance une demande et des offres de services. Le recours à cette ontologie permet l'implémentation de mécanismes de filtrage (comparaison) entre une demande et des offres qui mettent en oeuvre autre chose qu'une simple égalité.

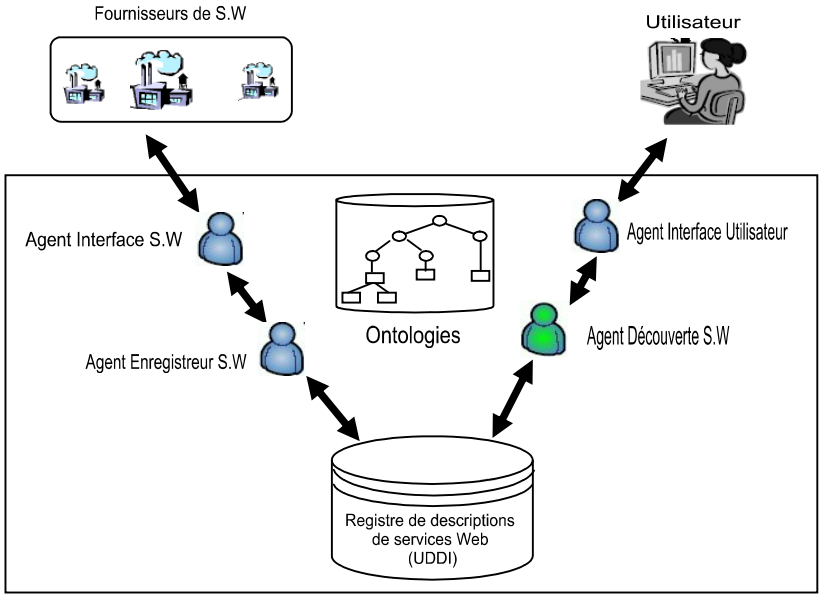

Fig. 1. Architecture multi-agents proposée

3.1 Descriptions des agents composant l'architecture

3.1.1 Agent interface service Web

Cet agent sert d'interface entre le système et le fournisseur du service Web, tel que pour chaque service Web un agent lui est associé. L'agent interface service Web permet l'enregistrement de la description sémantique relative au service Web. De plus, il permet des mises à jour des informations relatives au service Web.
L'architecture interne de l'agent interface service Web est composée de trois modules et un registre de sauvegarde, comme indiquée en figure 2.

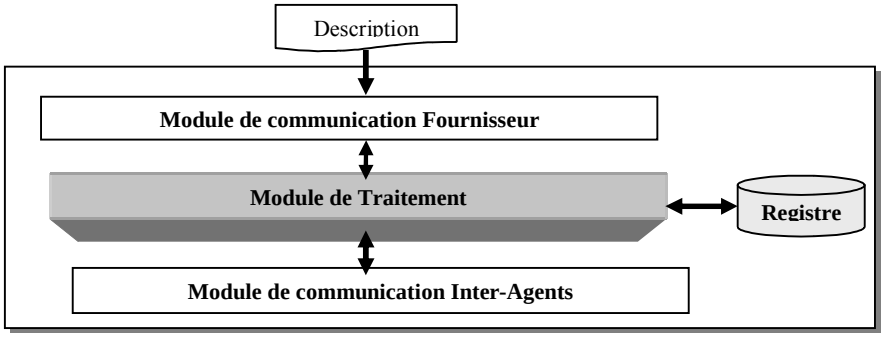

Fig. 2. Architecture de l'agent interface service Web

3.3.2 Agent enregistreur de services Web

Le rôle de cet agent est la sauvegarde des descriptions sémantiques des services Web au niveau du registre UDDI, il contient deux modules et une interface comme indiquée en figure 3.

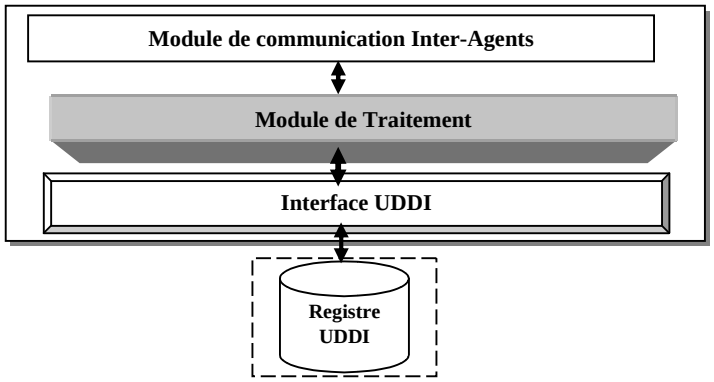

Fig. 3. Architecture de l'agent Enregistreur

3.3.3 Agent interface utilisateur

L'agent interface utilisateur est la porte d'entrée des requêtes externes au système. Il fournit à l'utilisateur le bon formulaire lui permettant de faire une requête.

C'est l'agent qui va initier la découverte, en émettant à l'agent découverte, une requête constituée d'entrées, de sorties, une référence sur l'ontologie de domaine à utiliser (par exemple l'ontologie des voyages touristiques) et présente les résultats adaptés aux préférences des utilisateurs après le traitement.

L'architecture interne de l'agent interface utilisateur est composée de trois modules principaux et d'un registre de sauvegarde comme l'indique la figure 4.

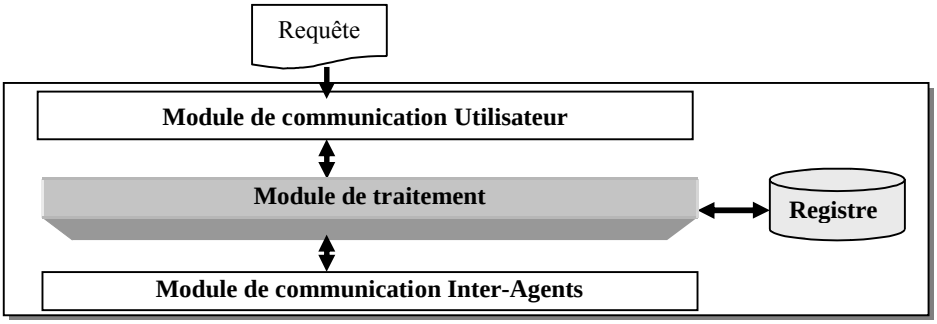

Fig. 4. Architecture de l'agent interface utilisateur

3.3.4 Agent de découverte de services Web

C'est un agent qui permet la découverte des descriptions des services Web satisfaisant la requête envoyée par l'agent interface utilisateur sur le plan sémantique.

L'architecture interne de l'agent découverte est composée de deux modules et une base de services pour stockage les descriptions sémantiques des services rendus par UDDI comme l'indique la figure 5. Ils sont comme suit :

- **Module de communication inter-Agents :** Il reçoit de l'agent interface utilisateur la requête sous forme d'un message et suite à cela, il appelle le module de traitement. Il reçoit également des demandes de transmission de messages de module de traitement. Ces demandes de transmissions constituent des réponses des requêtes reçues.
- **Base des services** : est utilisée pour stocker les descriptions sémantiques des services Web satisfaisant la requête de l'utilisateur.

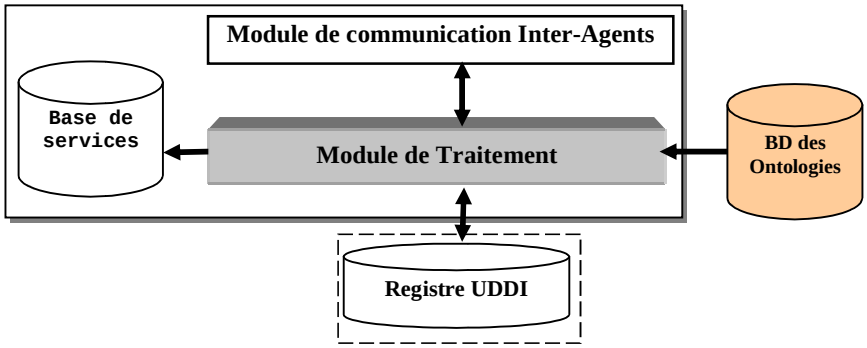

Fig. 5. Architecture de l'agent Découverte

- **Module de traitement** : il a deux tâches :

1) *la tâche d'analyse* : sélectionne l'ontologie de domaine correspondante à la demande (à partir de base d'ontologies qui stocke des ontologies de divers domaines), en extrait les classes et leurs liens et construit l'arborescence correspondante. Dans notre contexte, cette action est possible puisque le vocabulaire défini dans l'ontologie de domaine est décrit sous forme hiérarchique. Chaque sommet de cette arborescence

correspond à une classe de l'ontologie et chaque arc correspond à une relation de sous-classe. Cette arborescence permet de déduire des relations de généralisation (subsumption) entre les concepts, c'est-à-dire le fait qu'un concept soit plus général qu'un autre. Un concept C englobe (subsume) un concept C' si l'extension de C' est incluse dans celle de C. On dira alors que C est plus général que (ou englobe) C'. Ce principe nous permet de réaliser des comparaisons flexibles entre les offres et les demandes, c'est-à-dire d'associer à une demande des offres qui ne correspondent pas exactement aux besoins exprimés mais qui s'en rapprochent.

2) *La tâche de comparaison :* permet de comparer une demande et des offres de services en considérant l'ontologie (voir figure 6) et ce conformément aux quatre principaux modes de comparaison définis dans [7] en utilisant un algorithme de matchmaking : le mode Exact, le mode PlugIn, le mode Subsume et le mode Fail.

1. *Le mode Exact* sélectionne une offre si elle correspond exactement à une demande (demande = offre) c'est-à-dire les entrées et les sorties de la demande sont équivalents aux entrées et sorties de l'offre (matching exact).
2. *le mode Plug-In* retourne une offre si elle englobe une demande (demande<offre) c'est-à-dire les entrées de la demande englobe les entrées de l'offre et les sorties de la demande sont englobées par les sorties de l'offre dans l'ontologie de domaine (matching inclusif).
3. *le mode Subsume* retourne une offre si elle est incluse dans une demande (demande>offre) (l'Inverse de mode Plug-In) (matching partiel)
4. *le mode Fail* retourne faux, si aucune correspondance entre l'offre et la demande (demande # offre) (echec de matching).

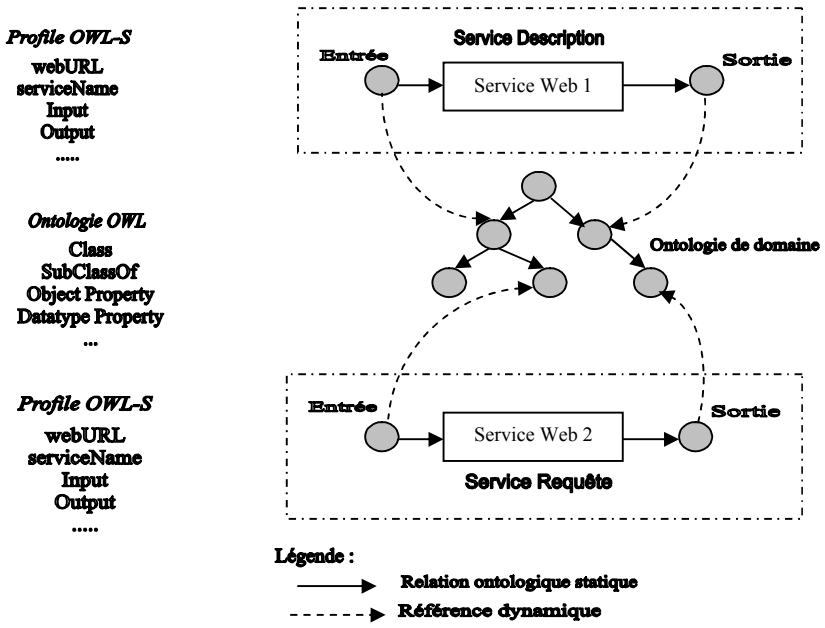

Fig. 6. méthodologie de comparaison

Les modes 2 et 3 de comparaison utilisent l'ontologie de domaine. Plus précisément, les offres et demandes de services étant exprimées en OWL-S, nous comparons, selon les quatre modes précédents, tous les éléments définis dans les clauses «input» et «output» (entrées et sorties) dans la classe ServiceProfile des offres et des demandes.

L'algorithme de comparaison utilisé à la fois en mode Plug-In et en mode Subsume utilise la fonction Englobe [8] (voir figure 7).

```
Fonction Englobe (E1 : chaîne, E2 : chaîne) :
booléen
% Cette fonction retourne vrai si E1 englobe E2 faux
sinon
% E1 est un élément de la clause Input ou Output de
l'Offre
% E2 est un élément de la clause Input ou Output de
Demande
% A représente l'ontologie (sous forme arborescente)
%On utilise les fonctions de haut niveau suivantes:
% Père(E) : retourne le père de E dans A
% Racine(A) : retourne la racine de A
Variables
SommetCourant : UnSommet   % Sommet de A en cours
d'examen
LesAncêtres : EnsembledeSommets % Les ancêtres de
E2
Début
LesAncêtres ← ∅
Si E2 = racine(A) Alors
% E2 n'a pas d'ancêtre et ne peut pas être englobé
LesAncêtres ← ∅
Sinon
SommetCourant ← Père(E2)
LesAncêtres ← Père(E2)
Tant Que (SommetCourant <>Racine(A)) Faire
     SommetCourant ← Père(SommetCourant)
        % «+» désigne l'ajout d'un nouvel élément
        % dans l'ensemble LesAncêtres
     LesAncêtres ← LesAncêtres + SommetCourant
     Fin Tant Que
     Fin Si
     Englobe ← (E1 ∈ LesAncêtres)
 Fin
```

Fig. 7. Fonction Englobe [8]

L'agent applique un test de subsomption sur les sorties (outputs) (voir figure 8) ensuite, on attribue un score pour chaque mode de matching : Exact (score=3), PlugIn (score=2), Subsume (score=1), Fail (score=0) (voir figure 9).

```
Procedure degreeOfMatch(OutD,OutO : chaîne )
% Cette Procédure retourne résultat de comparaison
% OutD, OutO sont la sortie de la demande et de
l'offre respectivement
Début
     Si   OutO = OutD Alors Return Exact
     Si   Englobe(OutO, OutD) Alors   Return PlugIn
     Si   Englobe(OutD, OutO) Alors   Return Subsume
     Autrement Return Fail
     Fin Si
Fin
```

Fig. 8. Procédure de matching des sorties (outputs)

```
Fonction GetScore(rel : chaîne) : Entier
% Cette Fonction retourne le score de matching
Val =0
Début
   Si rel = "Exact"   Alors  val = 3
   Si rel = "PlugIn"  Alors  val=2
   Si rel = "Subsume" Alors  val=1
   Si rel = "Fail"    Alors  val=0
    Fin Si
    GetScore ← val
Fin
```

Fig. 9. Fonction retourne le score de matching

Le niveau de correspondance sémantique entre les paramètres d'entrée (inputs) est assigné de la même manière que pour les paramètres de sortie (outputs).
L'équation (1) généralise la comparaison entre le concept de l'offre de service C_i^O et le concept correspondant de la requête C_i^D :

$$\text{Match}(C_i^D, C_i^O) = \begin{cases} 3 & \text{si } C_i^D = C_i^O \\ 2 & \text{si } C_i^D \sqsubset C_i^O \\ 1 & \text{si } C_i^D \sqsupset C_i^O \\ 0 & \text{sinon} \end{cases} \quad (1)$$

Supposons que l'on a m concepts dans la description de l'offre de service et m concepts correspondants dans la description de la requête, la similarité ou le match global entre la demande (requête) D et l'offre O peut dériver par prendre la somme de score de la paire de concepts (équation (2)) :

$$\text{Similarité}(D, O) = \sum_{i=1}^{m} \text{Match}(C_i^D, C_i^O) \quad (2)$$

Par conséquent, le matching entre la requête et un ensemble d'offres de services Web peut être mesuré de façon quantitative. Le service qui a un haut score de similarité représente le plus précis service pour la requête. Et on peut trouver plus d'un service.

3.2 Exemple

supposons qu'il existe trois services Web de vente S1, S2 et S3 publiés sur le Web. Ses paramètres fonctionnels (entrées, sorties) sont :

- S1 ayant deux entrées "vehicle" et "parts" et une seule sortie "price".
- S2 ayant deux entrées "parts" et "car" et une seule sortie "price".
- S3 ayant deux entrées "unit" et "material" et une seule sortie "price".

Et supposons qu'un client lance une requête de recherche constitue de deux entrées "Car" et "Parts" et une sortie "Price":
Et nous avons le fragment de l'ontologie de "vehicle" suivant :

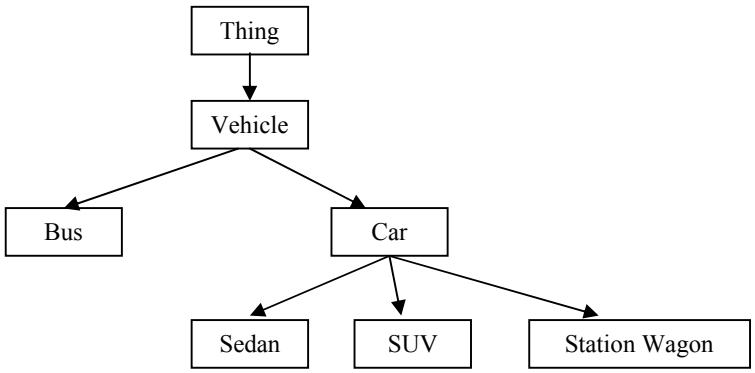

Fig. 10. Un fragment d'ontologie de vehicle

Si nous appliquons l'algorithme de matching, nous allons obtenir les résultats suivants:

- Comparaison des entrées (inputs) :
  S1 :
  Car → vehicle, mode = Plug-in, score = 2, Total =2
  Car → parts, mode= Fail, score = 0, Total = 2
  Parts → vehicle, mode= Fail, score = 0, Total = 2
  Parts → parts, mode = Exact, score = 3, Total =5
  Total du score des entrées = 5
  S2 :
  Car → parts, mode = Fail, score = 0, Total =0
  Car → car, mode= Exact, score = 3, Total = 3
  Parts → parts, mode = Exact, score = 3, Total =6
  Parts → car, mode = Fail, score = 0, Total =6
  Total du score des entrées = 6

S3 :

Car → unit, mode = Fail, score = 0, Total =0
Car → material, mode = Fail, score = 0, Total =0
Parts → unit, mode = Fail, score = 0, Total =0
Parts → material, mode = Fail, score = 0, Total =0
Total du score des entrées = 0

- Comparaison des sorties (outputs) :
  S1 : price → price, mode = Exact, score = 3, Total =3
  Total du score des sorties = 3
  S2 : price → price, mode = Exact, score = 3, Total =3
  Total du score des sorties = 3
  S3 : price → price, mode = Exact, score = 3, Total =3
  Total du score des sorties = 3
- matching global :
  S1 : Total du score (total score des entrées + total score des sorties) = 5 + 3 = 8, Bien
  S2 : Total du score = 6 + 3 = 9, Meilleur
  S3 : Total du score = 0 + 3 = 3, Pas bon

► Donc, le service Web S2 est considéré comme le meilleur qui correspond à la requête.

4 Conclusion

Dans ce papier nous avons présenté un cadre conceptuel et architectural, fondée sur les services Web, pour l'interopérabilité.

La découverte de services Web constitue un axe de recherche émergent. Diverses approches ont été proposées. Ces approches sont passées d'une recherche basée mots-clés (découverte syntaxique) aux méthodes basées sémantiques. Nous avons proposé une approche à base d'agents qui modélise la découverte des services Web sémantiques. Notre architecture à base agents est composée :

- d'un agent interface fournisseurs services Web.
- d'un agent interface utilisateurs.
- d'un agent enregistreur de service Web au sein de registre UDDI.
- d'un agent de découverte de(s) service(s) Web.

L'agent de découverte de services Web applique des inférences pour apparier la requête de l'utilisateur avec les services offerts. L'appariement (matching) repose sur la comparaison des sorties et des entrées de la requête avec les sorties et les entrées du service, et présente différents niveaux de matching : exact, plugin, subsume et fail.

A court terme, nous allons implémenter notre proposition d'architecture. Afin de valider notre travail, nous effectuerons des tests avec des requêtes d'utilisateurs variés et un panel de services Web.

En ce qui concerne les perspectives de notre travail, nous prévoyons les points suivants :

- En ce qui concerne l'algorithme de matching on pourra prévoir d'autres paramètres de recherche tels que les préconditions et les effets, ces derniers augmentent les taux de précision.
- Proposer un matching indirect en cas d'absence de matching direct c'est-à-dire passer à l'étape de composition de services.
- Nous pourrons essayer d'utiliser d'autres types d'agents comme les agents mobiles et évaluer leurs effets sur les performances.

Références

1. Berners-Lee, T., Hendler, J. et Lassila : The Semantic Web, In Scientific American, (2001), 35-43.

2. Bertrand Sajus. : La fonction Thésaurale au coeur des systèmes d'information, ADBS, www.adbs.fr/adbs/prodserv/jetude/html/prog-110402a.html, (2002).

3. Ankolekar, A., Burstein, M., Hobbs, J., Lassila, O., Martin, D., McIlraith, S., Narayanan, S., Paolucci, M., Payne, T., Sycara, K., Zheng, H. : Daml-s semantic markup for web services. In Proceedings of International Semantic Web Conference (ISWC), Sardinia, Italy, (2003), 348–364.

4. Martin, D., Burstein, M., Hobbs, J., Lassila, O., McDermott, D., McIlraith, S., Narayanan, S., Paolucci, M., Parsia, B., Payne, T., Sirin, E., Srinivasan, N., and Sicara, K., OWL-S : Semantic markup for web services, Tech. rep., France Telecom, MINDL Maryland, NIST, Nokia, (2004).

5. Bryson, J., Martin, D., McIlraith, S., and Stein, L. : Agent-based composite services in daml-s : The behavior-oriented design of an intelligent semantic web, Springer Verlag, (2002).

6. Robert H. Guttman, Alexandros G. Moukas, and Pattie Maes : Agent-mediated Electronic Commerce: A Survey, Software Agents Group, MIT Media Laboratory, (1998).

7. Paolucci, M., Kawamura, T., Payne, T., Sycara, K.: Semantic matching of web services capabilities. In: Proceedings of the First International Semantic Web Conference, LNCS 2342, Springer-Verlag, (2002), 333–347.

8. Lotfi Bouzguenda, Rafik Bouaziz, Eric Andonoff : Utilisation d'Ontologies pour la Coordination dans le Workflow Inter-Organisationnel Lâche, (2006), 10-11.