République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

UNIVERSITE MOHAMED KHIDER DE BISKRA

Faculté des sciences exactes et sciences de la nature et de la vie

Département d'informatique

N°d'ordre :…….
N°de Série :…….

Thèse

En vue de l'obtention du diplôme de

Doctorat en Sciences en Informatique

(Option: Intelligence artificielle et systèmes distribués)

Présentée par:

BEN SEGHIER Nadia

THÈME

Un Framework à base d'agent mobile, méta-donnée et profil utilisateur pour service Web

Soutenue devant le jury

Présidente :	Mokhtari Aicha	Professeur	USTHB Alger
Rapporteur :	Kazar Okba	Professeur	Université de Biskra
Examinateurs :	Guergouri Faiez	Professeur	ISIM de Sfax, Tunisie
	Ezziyyani Mostafa	Professeur	FSTT Tanger, Maroc
	Kahloul Laid	MCA	Université de Biskra
	Benharzallah Saber	MCA	Université de Biskra
Invité :	Rezeg Khaled	MCA	Université de Biskra

Remerciements

Avant, toute personne, je tiens à remercier notre Dieu Tout Puissant pour m'avoir éclaircit le chemin de ce travail.

Mes vifs remerciements vont également à Monsieur ***Okba KAZAR*** *professeur à l'université de Biskra d'avoir assuré l'encadrement de cette thèse ainsi que pour ses précieux conseils et la confiance qu'il m'a accordée qui ont fortement contribué à mener à bien ce travail.*

Je tiens à remercier vivement Madame ***Mokhtari Aicha****, Professeur à l'université USTHB d'Alger, pour m'avoir fait l'honneur de présider le jury de ma soutenance.*

Je tiens à exprimer toute ma gratitude à Monsieur ***Guergouri Faiez****, Professeur à l'université ISIM de Sfax - Tunisie, pour avoir bien voulu juger le travail.*

Je tiens aussi à exprimer toute ma gratitude à Monsieur ***Ezziyyani Mostafa****, Professeur à l'université FSTT Tanger, Maroc pour avoir bien voulu juger le travail et faisant partie de jury de soutenance.*

Je tiens à exprimer toute ma gratitude à Monsieur ***Kahloul Laid****, Maître de conférences A à l'université de Biskra, d'avoir accepté d'examiner mon travail.*

Je tiens à remercier vivement Monsieur ***Benharzallah Saber****, Maître de conférences A à l'université de Biskra, pour avoir bien voulu juger le travail et faisant partie de jury de soutenance.*

Je dois aussi remercier vivement Monsieur ***Rezeg Khaled,*** *Maître de conférences A à l'université de Biskra, pour l'accueil et l'aide précieuses et ses encouragements.*

Enfin, Un grand merci s'adresse à ceux qui m'ont aidé de près ou de loin.

Dédicaces

Je dédie ce modeste travail:

À ceux que j'ai de plus cher au monde : mes parents.

À mon époux et mon fils Abd El Salam.

À mes frères et mes sœurs et mes neveux.

À tous mes amis et collègues.

À tous ceux qui m'ont aimé et me souhaitent le bonheur et la réussite.

Résumé

Les services Web sont des technologies émergentes et prometteuses pour le développement, le déploiement et l'intégration d'applications sur l'Internet. Ils constituent la technologie de base pour le développement d'architectures orientées services. Ces architectures sont de plus en plus répandues sur le Web. Un des avantages majeurs des services Web par rapport à ses prédécesseurs (CORBA, DCOM, COM, …) est l'apport de l'interopérabilité sur l'Internet. Le principe essentiel de l'approche service Web est de transformer le Web en un dispositif distribué d'échange et de calcul, où les services Web peuvent interagir d'une manière intelligente.

La découverte de services Web, assurée par les registres UDDI, est relativement primitive. Elle ne tient pas compte de la croissance continue du nombre de services Web offerts sur le Web et des modifications constantes subies par ces derniers. Le registre UDDI permet de découvrir des services Web. Cependant, elle ne permet pas de choisir le meilleur fournisseur d'un service Web parmi plusieurs offrants le même service. Elle n'offre pas de mécanisme qui permette de choisir un service Web en se basant sur sa qualité. Le registre manque aussi d'une sémantique compréhensible et interprétable par des machines capables de sélectionner le meilleur service Web disponible. De plus, un UDDI centrale souffre de problème de seul point centralisé (point d'échec) et le coût élevé de la maintenance. Pour régler ses problèmes, nous présenterons dans le cadre de cette thèse un nouvel Framework à base de profil utilisateur et métadonnées pour la découverte sémantique des services Web, en s'appuyant sur la capacité de la mobilité des agents.

Mots clés

Agent mobile, Architecture distribuée, Découverte sémantique, Métadonnées, Ontologies, Profil utilisateur, Qualité des services Web, Service Web.

Abstract

Web services are emerging and promising technologies for development, deployment and integration of applications on the Internet. They constitute the basic technology for developing service oriented architectures. These architectures are becoming more prevalent on the Web. A major advantage of Web services over its predecessors (CORBA, DCOM, COM …) is the contribution of interoperability on the Internet. The essential principle of the Web service approach is to transform the Web into a distributed system of exchange and calculation, where Web services can interact in an intelligent manner.

Web services discovery provided by the UDDI registries is relatively primitive. It does not take into account the continuous growth in the number of services on the Web. The UDDI standard has been proposed and used for Web service publication and discovery. However, it does not allow users to choose the best provider. It does not offer a mechanism to choose a Web service based on its quality. The register also lacks of sufficient semantic description in the content of Web services, this lack makes it difficult to find and compose suitable Web services during analysis, search, and matching process. In addition, a central UDDI suffers from one centralized point problem and the high cost of maintenance. To get around these problems, we propose a novel Framework based on mobile agent, user profile and metadata catalogue for semantic Web services discovery.

Keywords

Distributed architecture, Metadata, Mobile agent, Ontologies, Quality of service (QoS), Semantic discovery, User profile, Web service.

الملخص

خدمات الانترنت هي تكنولوجيا حديثة و واعدة, تسمح بإنشاء و تطوير تطبيقات متكاملة على شبكة الانترنت. فهي تشكل القاعدة الأساسية لبنية الخدمات التطبيقية (SOA) . من بين الايجابيات الرئيسية لخدمات الويب هو المساهمة في العمل المشترك داخل شبكة الانترنت. فالمبدأ الرئيسي لها هو تحويل الويب إلى جهاز تبادل وحوسبة موزع، حيث يمكن لخدمات الويب أن تتفاعل بذكاء.

طريقة اكتشاف خدمات الويب عن طريق السجلات UDDI هي طريقة بدائية نسبيا.فهي تتجاهل التزايد المستمر في خدمات الويب على شبكة الانترنت والتغيرات المستمرة التي تمر بها. بالإضافة إلى أن المعيار UDDI لا يقدم و لا يختار أفضل خدمة ويب من بين العديد من موردي نفس الخدمة . فهو لا يقدم آلية لتحديد و اختيار خدمة ويب عن طريق النوعية (QoS) .كما أنه يفتقر إلى المفهوم الدلالي وللتفسير من قبل الأجهزة القادرة على اختيار أفضل خدمة ويب متاحة. بالإضافة إلى ذلك، فمركزية السجل UDDI تجعله يعاني من مشكلة النقطة الواحدة المركزية (points échec) وارتفاع تكلفة صيانته. لحل هذه المشاكل، اقترحنا في هذه الأطروحة بنية جديدة لاكتشاف واختيار خدمات الويب بطريقة أوتوماتكية. هذه البنية تعتمد على استخدام خصائص و ايجابيات الويب الدلالي من : انطولوجيا (ontologie), قاعدة بيانات(métadonnées),... بالإضافة إلى استعمال خصائص و مميزات و رغبات المستخدم من أجل تلبية احتياجاته بطريقة أفضل. و استخدمنا أيضا نموذج العميل أو الوكيل المتحرك في نظامنا من أجل نقاط القوة التي لديه مثل: الاستقلالية, المرونة, التنقل عبر الانترنت.

الكلمات المفتاحية

الاكتشاف و الاختيار, الأنظمة الموزعة, الانطولوجيا ,المحتوى الدلالي ,الوكيل المتحرك , خدمات الويب , خصائص المستخدم , نوعية خدمات الويب.

Table des matières

Liste des figures

Liste des tableaux

Chapitre I

Introduction Générale

I.1. Contexte de recherche

De nos jours, le domaine des services Web et le domaine du Web sémantique se croisent de plus en plus. Les services Web ont permis au Web de passer du rôle habituel de réseaux d'informations à un intergiciel (Middleware) d'applications grâce à l'exploitation croissante des technologies XML. En effet, les services Web étant des applications modulaires, faiblement couplées et auto-descriptives parviennent à fournir un modèle facile à comprendre et à utiliser dans la programmation et le déploiement d'applications. En plus, le modèle est basé sur des normes et est apte à s'exécuter à travers le Web. Le processus de standardisation des services Web est actuellement composé de trois principales couches : un protocole de communication (SOAP), une spécification de description des interfaces (WSDL), et une spécification de publication et de localisation (UDDI) (Kadima, 2003).

Au fur et à mesure que les services Web se multiplient, la difficulté de la découverte de service s'accroît. La maitrise de cette phase nécessite la personnalisation de la sélection des services Web désirés par l'utilisateur. La découverte représente l'une des opérations d'interaction entre les éléments de service Web (le fournisseur, le client et l'annuaire UDDI) (Benmokhtar, 2004) (Benna, 2008). Elle est définie comme suit : « *la découverte : les services sont conçus afin d'être sélectionnés via des mécanismes de recherche. La découverte est permise par la description préalable des services et leur publication au sein d'un registre. Elle est effectuée en deux phases* » (Lopez, 2008):

- ***La recherche**: elle peut se faire via les registres dans lesquels les fournisseurs ont publié leurs services Web ;*
- ***La sélection**: le client doit ensuite sélectionner parmi la collection de services Web issus de l'étape de recherche, le service Web qui convient le mieux à ses attentes.*

Il s'avère malheureusement que la découverte des services Web est basée seulement sur la syntaxe. L'aspect sémantique indispensable pour satisfaire l'utilisateur y est absent (Friesen, 2005). Le Web sémantique permet de développer des langages pour exprimer des informations d'une manière traitable par une machine. Il permet de transformer le contenu du Web en un système qui pourrait être compris par un ordinateur, et rendre explicite le contenu sémantique des ressources dans le Web. Donc, les machines peuvent de se concevoir comme des humains. Les ordinateurs et les agents logiciels pourraient donc comprendre les informations contenues dans ces ressources et aider les utilisateurs à exécuter et compléter leurs tâches, leurs requêtes de façon plus automatique et plus efficace. En effet, le traitement automatisé des données requiert une représentation de la sémantique compréhensible et échangeable par les machines (Rezeg, 2010).

Le premier objectif du Web sémantique est de définir et de lier sémantiquement les ressources du Web en général, et les services Web en particulier, afin de simplifier leur utilisation, leur découverte, leur intégration, et leur réutilisation dans le plus grand nombre d'applications. Ces aspects du Web sémantique peuvent faciliter la description, la recherche, la sélection et la composition de services Web dans des applications à base de SOA. Le Web sémantique doit fournir l'accès à ces ressources par l'intermédiaire de descriptions sémantiques exploitables et compréhensibles par des machines. Cette description repose sur les ontologies.

Les services Web sémantiques sont des services Web dont la description est améliorée par des langages empruntés au Web sémantique, tel qu'OWL. Cet emprunt au Web sémantique permet à ces services Web d'être découverts et sélectionnés automatiquement, par des machines ou d'autres services Web distants. Ceci permet aux services Web sélectionnés de répondre au mieux aux différentes requêtes complexes d'un utilisateur. Les motivations pour développer ces services Web sémantiques sont évidemment de faciliter les phases automatiques de découverte, sélection et composition de services Web. En effet, si leur sémantique est connue, alors chercher et composer des services Web pourra être fait

automatiquement en donnant la sémantique cible. Ces dernières années, la recherche dans le domaine des services Web sémantiques a été très active (Châtel, 2010).

Dans la communauté des chercheurs des services Web, des efforts importants sont maintenant consacrés à ce problème, qui consiste a proposé une architecture orienté service pour la découverte de services Web sémantiques.

I.2. Problématique

L'infrastructure de base d'un service Web concernant les standards SOAP, WSDL, et UDDI est suffisante pour déposer des programmes interopérables et intégrables entre eux facilement, mais, insuffisante pour rendre l'automatisation des plusieurs tâches liées au cycle de vie des services Web, par exemple, la découverte des services Web requis. Cette automatisation est pourtant essentielle pour faire face aux exigences du passage à l'échelle, d'une forte réactivité dans un environnement hautement dynamique et de la volonté de réduire les coûts de développement et de maintenance des services.

Dans cette thèse, nous traitons deux problèmes majeurs qui rencontrent la découverte des services Web dans l'architecture des services Web. Le premier est lié au point central de publication et de découverte implémenté généralement par un registre UDDI. Le deuxième problème est lié à la description technique fournie par le contrat publié dans l'annuaire UDDI.

La découverte de services Web, assurée par les registres UDDI, est relativement primitive. Elle ne tient pas compte de la croissance continue du nombre de services Web offerts sur le Web et des modifications constantes subies par ces derniers. Le registre UDDI permet de découvrir des services Web. Cependant, elle ne permet pas de choisir le meilleur fournisseur d'un service Web parmi plusieurs offrants le même service. Elle n'offre pas de mécanisme qui permette de choisir un service Web en se basant sur sa qualité. Le registre manque aussi d'une sémantique compréhensible et interprétable par des machines capables de sélectionner le meilleur service Web disponible. De plus, un UDDI centrale souffre de problème de seul point centralisé (point d'échec) et le coût élevé de la maintenance.

En résumé, la description syntaxique proposée par la norme UDDI est insuffisante pour permettre l'automatisation du processus de découverte, de sélection, de composition et de l'invocation des services Web.

L'objectif de notre travail de recherche porte sur la description et la découverte des services Web sémantiques. Ces services sont accessibles et consultables depuis un annuaire sémantique. Pour cela, nous développons un Framework décentralisé à base de profil utilisateur et métadonnées pour la découverte des services Web, en s'appuyant sur la capacité de la mobilité des agents.

I.3. Objectifs et contributions de la thèse

Avec la croissance exponentielle en nombre et en fonctionnalités des services sur Internet, ainsi que la diversité des technologies utilisés pour l'implémentation et la présentation de ces derniers, il est devenu essentiel de proposer un système de découverte de services qui permet au client de choisir les meilleurs services qu'il désire sans avoir à effectuer n'importe quelle tâche manuellement. Notre contribution afin d'atteindre cet objectif s'articule autour des points suivants :

- La première partie de notre travail consiste à proposer une description sémantique, qui couvre les principaux éléments fonctionnels et non fonctionnels des services Web, afin d'enrichir la spécification d'informations sémantiques qui décrivent ces services. Cette description repose sur les points suivants :
  - ✓ Une description de service au niveau registre UDDI pour décrire l'aspect fonctionnel du service Web ;
  - ✓ Une description de service au niveau catalogue de métadonnées pour décrire l'aspect sémantique du service Web ;
  - ✓ Une description de service concernant la qualité de service Web (QoS) ;

- La deuxième partie de notre travail consiste à proposer une architecture progressive et distribuée à base d'agents mobiles pour la découverte dynamique des services Web sémantiques d'une façon automatique, en partant d'une requête déclarative simple spécifiée par un utilisateur. Afin de bien illustrer cette architecture, nous avons jugé utile de présenter l'ensemble des enchainements pour les différentes étapes devant permettre le processus de découverte automatique des services Web sémantiques :

- ✓ La requête de l'utilisateur doit être exprimée à travers une interface interactive et conviviale du système. Cette étape constitue la première phase de notre démarche ;
- ✓ La deuxième phase de notre processus comporte l'enrichissement de la requête exprimée par l'utilisateur selon son profil;
- ✓ La troisième phase comprend la découverte des services Web sémantiques qui peuvent satisfaire la requête d'utilisateur. Elle se déroule sur cinq étapes:
  1- Appliquer un algorithme de matching fonctionnel au niveau UDDI, en se basant sur l'aspect syntaxique en utilisant l'interrogation des différentes pages (blanches, jaunes et vertes).
  2- Appliquer un algorithme de matching sémantique (Chabeb, 2011) au niveau UDDI, il est utilisé pour trouver la similarité sémantique entre les paramètres d'entrée (inputs) et les paramètres de sortie (outputs) de service Web présentés dans la demande de l'utilisateur et ceux de l'annonce (WSDL).
  3- Appliquer un algorithme de matching sémantique (Chabeb, 2011) au niveau catalogue de métadonnées, il est utilisé pour trouver la correspondance sémantique entre les caractéristiques des services Web présentés dans la demande de l'utilisateur et ceux stockés dans le catalogue de métadonnées des services Web.
  4- Appliquer un algorithme de calcul de score QoS de chaque service Web issu de la phase précédente.
  5- Appliquer un algorithme efficace pour calculer de manière pertinente le degré global de matching pour la découverte.

- La dernière étape de notre travail consiste à concrétiser la solution proposée à travers une implémentation. Pour atteindre ce but, nous avons eu recours à plusieurs outils issus des technologies du Web sémantique, services Web et celles des agents mobiles. Une étude de cas relative a été présentée pour motiver la solution proposée.

I.4. Organisation de la thèse

Cette thèse est scindée en deux parties majeures :

Nous commençons, tout d'abord, par l'introduction générale qui représente les points essentiels de notre travail, concernant le contexte de travail, la problématique, les objectifs, la contribution et l'organisation de ce document.

La première partie est consacrée à un état de l'art relatif aux domaines abordés dans cette thèse. Elle contient trois chapitres :

Dans le deuxième chapitre nous présentons un état de l'art sur les services Web et plus particulièrement sur la technologie des services Web sémantique. Nous introduisons aussi la définition et l'historique de Web sémantique, ainsi que les différents langages de descriptions sémantiques dédiés aux Web services. Ensuite, nous abordons quelques approches concernant ce domaine de recherche.

Dans le troisième chapitre nous proposons un état de l'art sur le profil utilisateur en spécifiant le profil utilisateur, sa définition, ses composantes et ses différents méta-modèles, ainsi qu'une présentation de certains travaux relatifs.

Dans le quatrième chapitre nous présentons un état de l'art sur les systèmes multi-agent et plus particulièrement sur la technologie d'agent mobile: définitions, caractéristiques, motivation en démontrant les avantages de ce paradigme, une synthèse sur les différentes plates-formes d'agents mobiles existantes de nos jours, etc. Ensuite, nous évoquons quelques travaux de découverte des services Web à base d'agent mobile.

La deuxième partie est le coeur de notre travail. Elle est consacrée à notre contribution, elle présente l'architecture de Framework proposé ainsi que la mise en ouvre de cette architecture. Elle est composée de deux chapitres:

Le cinquième chapitre présente notre Framework de découverte sémantique des services Web basé sur les métadonnées, profil utilisateur et les agents mobiles. Dans ce chapitre nous faisons en premier lieu un survol des concepts exploités dans notre travail. En deuxième lieu nous présentons l'architecture de Framework proposé suivie de l'aspect architectural et comportemental de chaque composant.

Le sixième chapitre illustre l'application de notre approche à une étude de cas. Nous décrivons tout d'abord les différents outils techniques utilisés. Nous précisons ensuite l'implémentation des différents composants inclus dans notre architecture et comment les opérations de découverte peuvent être réalisées.

Nous avons conclu notre manuscrit par une conclusion générale qui discute les apports de notre travail, ainsi que les perspectives de recherche envisagées.

Première Partie

ETAT DE L’ART

Chapitre II

La Découverte Sémantique des Services Web

II.1. Introduction

Les services Web (SW) sont une technologie émergeante. Ce sont des composants softwares qui fournissent des fonctionnalités accessibles via des protocoles standardisés du Web. Basés sur le standard XML, ils sont caractérisés par leur indépendance aux plateformes et aux systèmes d'exploitation, ce qui a impliqué leur adoption par différentes organisations commerciales et industrielles offrant leurs services à travers le Web, et par conséquent l'augmentation du nombre des services Web offerts.

La découverte des services Web représente un axe de recherche émergeant. En général, la découverte est faite au niveau du registre UDDI, elle est basée essentiellement sur la recherche syntaxique des descriptions WSDL des services Web. La tendance actuelle de la communauté des chercheurs est d'exploiter les technologies du Web sémantique, par exemple RDF, RDFS, OWL, afin d'enrichir les services Web de descriptions sémantique. La combinaison des technologies des services Web et du Web sémantique a mené au concept des services Web sémantiques.

Les services Web sémantique sont des services Web dotés de descriptions sémantiques. Cette sémantique est apportée grâce aux ontologies une des technologies importante du Web sémantique. Ainsi, des agents logiciels peuvent être développés afin de raisonner sur ces ontologies rendant la découverte des services Web dynamique et automatique.

Ce chapitre est organisé comme suit : la section 2 fait l'objet de l'étude des services Web en indiquant leur architecture, les standards de cette technologie ainsi que les opérations appliquées sur les services Web. Ensuite, nous étudierons le Web sémantique et ses éléments dans la section 3. La section 4 est consacrée à la présentation des services Web sémantique. Par la suite, nous évoquons la notion de qualité de service Web (QoS) dans la section 5. Dans la section 6 nous présentons la définition de la découverte de

services, les différentes approches de découvertes et les catégories des différents mécanismes de découverte et sélection. Ensuite, nous avons présenté dans la section 7 quelques travaux pour la découverte des services Web. Enfin, nous concluons ce chapitre dans la section 8.

II.2. Les services Web

Dans cette section nous présentons les principaux concepts liés aux services Web.

II.2.1. Définition des services Web

Le Web est de plus en plus le support privilégié des applications. Les services Web constituent le développement ultime dans ce domaine. Le terme service Web est souvent utilisé de nos jours, mais pas toujours avec la même signification, car la signification est tributaire de l'application de cette technologie. On peut dire qu'un service Web est souvent vu comme une application accessible à d'autres applications sur le Web, mais il existe plusieurs définitions pour les services Web :

Définition 1 : le consortium W3C définit un service Web comme étant : « *une application, ou un composant logiciel qui vérifie les propriétés suivantes :*

- *Il est identifié par un URI ;*
- *Ses interfaces et ses liens peuvent être décrits en XML;*
- *Sa définition peut être découverte par d'autres services Web ;*
- *Il peut interagir directement avec d'autres services Web à travers le langage XML en utilisant des protocoles Internet standards. »* (Cerami, 2002)

Définition 2 : « *un service Web est une application accessible à partir du Web. Il utilise les protocoles Internet pour communiquer, et utilise un langage standard pour décrire son interface.* » (Melliti, 2004).

Définition 3 : «*un service Web est un système logiciel conçu pour permettre l'interopérabilité de machine-à-machine sur un réseau. Dispose d'une interface décrite dans un format exploitable par machine (en particulier WSDL). D'autres systèmes exerce une interaction avec le service Web d'une manière prescrite par sa description en utilisant des messages SOAP, typiquement transmis avec le protocole HTTP avec une sérialisation XML, en conjonction avec d'autres standards relatifs au Web*» (Booth, 2004).

Définition 4 : « *les services Web sont la nouvelle vague des applications Web. Ce sont des applications modulaires, auto-contenues et auto-descriptives qui peuvent être publiées,*

localisées et invoquées depuis le Web. Les services Web effectuent des actions allant de simples requêtes à des processus métiers complexes. Une fois qu'un service Web est déployé, d'autres applications (y compris des services Web) peuvent le découvrir et l'invoquer. » (Ponge, 2008).

II.2.2. Architecture des services Web

Techniquement, un service Web peut donc être perçu comme étant une interface décrivant une collection d'opérations accessibles via le réseau, à travers des messages XML standardisés. D'un point de vue technique, la description d'un service Web inclut tous les détails nécessaires à l'interaction avec le service constituant, ce qu'on appelle l'architecture des services Web, par exemples, le format des messages, les signatures des opérations, le protocole de transport et la localisation du service Web. (Kreger, 2001)

L'interopérabilité est l'objectif premier des services Web. Pour permettre cet échange d'information entre des applications distantes, les SW sont composés des couches standards. Nous mettons en relief dans cette section deux types d'architecture. La première dite *de référence,* elle contient trois couches principales. La seconde architecture est plus complète, elle utilise les couches standards de la première architecture en ajoutant au-dessus d'autres couches plus spécifiques. Elle est appelé architecture *étendue* ou *en pile*.

II.2.2.1. Architecture de référence

L'architecture de référence s'articule autour des trois rôles suivants:

- **Le fournisseur de service** : correspond au propriétaire du service Web. D'un point de vue technique, il est constitué par la plateforme d'accueil du service Web ;
- **Le client :** correspond au demandeur de service Web. D'un point de vue technique, il est constitué par l'application qui va rechercher et invoquer un service Web;
- **L'annuaire des services :** correspond à un registre de descriptions des services Web offrant des facilités de publication des services Web à l'intention des fournisseurs, ainsi que des facilités de recherche des services Web à l'intention des clients.

Les interactions de base entre ces trois rôles incluent les opérations de publication, de recherche et de liens d'opérations. Avec ce schéma, nous essayons de décrire ces interactions. Nous citons, notamment les standards émergents suivants :

- **SOAP (Simple Object Access Protocol) :** est un protocole léger destiné à l'échange d'informations structurées dans un environnement décentralisé, distribué. Il utilise des

technologies XML pour définir une structure d'échange de messages fournissant une construction de messages pouvant être échangés sur divers protocoles sous-jacents. La structure a été conçue pour être indépendante de tout modèle de programmation et autres sémantiques spécifiques d'implémentation. (SOAP, 2004)

- **WSDL (Web Services Description Langage) :** le langage de description WSDL se comporte donc comme un langage permettant de décrire l'interface visible (ou publiée) du service Web. Il décrit, à l'aide du langage de balises XML, les différents éléments du service, (Rampacek, 2006). Une description WSDL d'un service Web est faite sur deux niveaux, un niveau abstrait et un niveau concret. Au niveau abstrait, la description du service Web consiste à définir les éléments de l'interface du service Web tel que les types de données (*Data Types*), les messages (*Message*), les types de ports (*Port Type*).
- **UDDI (Universal Description, Discovery and Integration):** UDDI est une spécification en langage XML d'un catalogue des services offerts par les entreprises sur leurs sites Web. Le catalogue comprend les adresses et les contacts des entreprises, une classification sectorielle et une description des services proposés. UDDI fournit différentes manières de recherche à travers les pages jaunes, vertes et blanches, il est facile à interroger sur Internet mais n'impose pas de contraintes sur la description de services que doivent introduire les fournisseurs de ces derniers pour publier leurs services. (Chauvet, 2002)

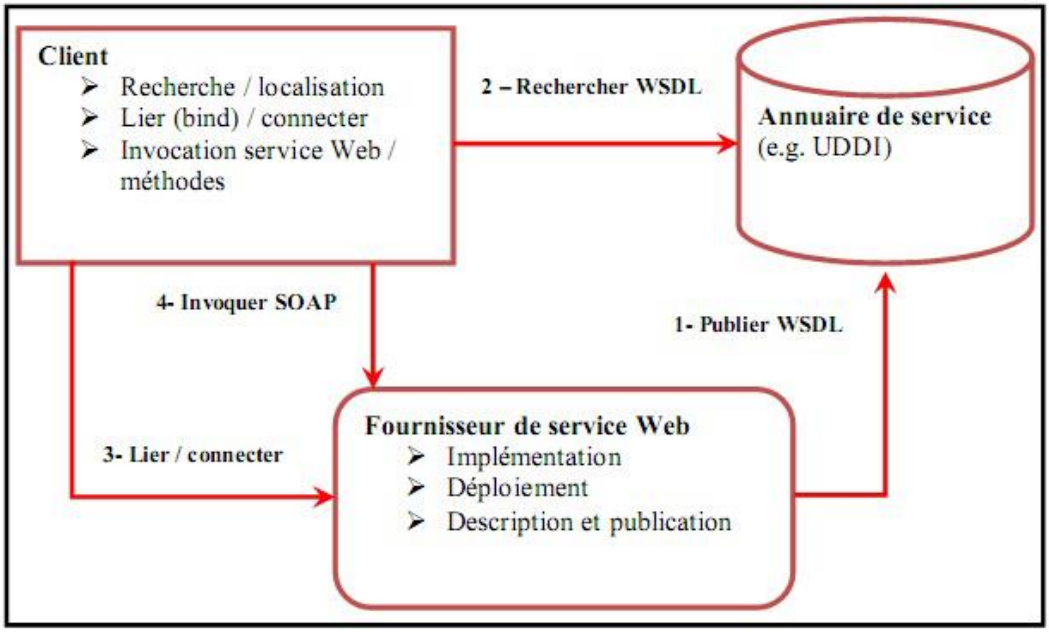

Figure II.1 : Architecture de référence des services Web (Kreger, 2001).

Cependant, cette infrastructure n'est pas suffisante pour permettre une utilisation effective des services Web, dans les domaines dont les exigences vont au-delà de la capacité d'interactions simples via des protocoles standards. Par exemple, dans le domaine du e-business, donc la nécessité d'introduire une nouvelle architecture suffisante et complète.

II.2.2.2. Architecture étendue

Une architecture étendue est constituée de plusieurs couches se superposant les unes sur les autres, d'où le nom de pile des services Web. La pile est constituée de plusieurs couches, chaque couche s'appuyant sur un standard particulier. On retrouve, au-dessus de la couche de transport, les trois couches formant l'infrastructure de base décrite précédemment. Nous apportons une explication de la mise en relief des trois types de couches (voir la figure II.2) :

- **L'infrastructure de base (Discovery, Description, Exchange):** ce sont les fondements techniques établis par l'architecture de référence. Nous distinguons les échanges des messages établis par SOAP, la description de service par WSDL et la recherche de SW que les organisations souhaitent utiliser via le registre UDDI ;
- **Couches transversales (Security, Transactions, Administration, QoS):** ce sont ces couches qui rendent viable l'utilisation effective des SW dans le monde industriel ;
- **La couche Business Processus (BusinessProcess):** cette couche supérieure permet l'intégration de services Web, elle établit la représentation d'un «BusinessProcess» comme un ensemble de SW.

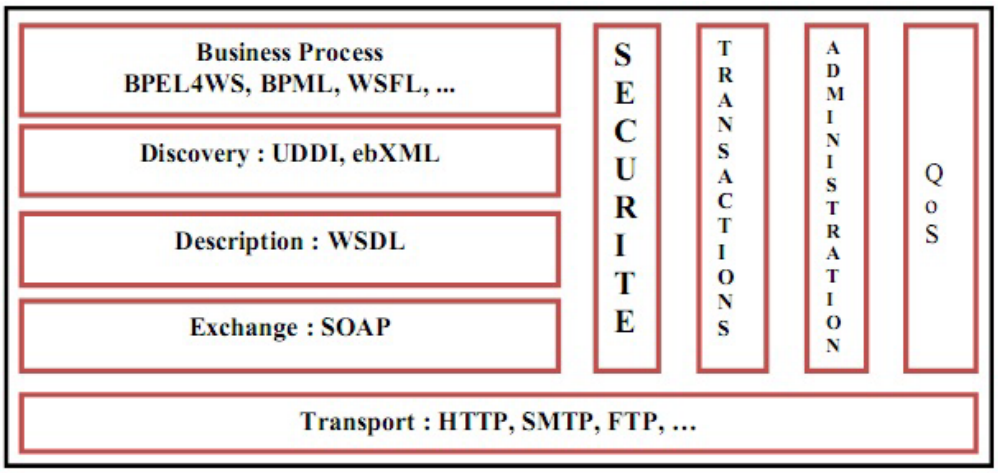

Figure II.2 : Architecture en pile des services Web (Kreger, 2001).

II.3. Le Web sémantique

Dans cette section nous verrons les principales caractéristiques du Web sémantique.

II.3.1. Définition et présentation du Web sémantique

Le Web sémantique, inventé en 1989 par Tim Berners-Lee directeur et fondateur du W3C, fournit un cadre de travail commun qui permet à des données d'être partagées et réutilisées. C'est un effort de collaboration mené par le W3C, avec la participation d'un grand nombre de chercheurs et d'industriels associés.

Selon Tim Berners-Lee (Berners-Lee, 2001) : *« Le Web sémantique n'est pas un Web distinct mais bien un prolongement du Web que l'on connaît, dans lequel, on attribue à l'information une signification clairement définie, ce qui permet aux ordinateurs et aux humains de travailler en plus étroite collaboration »*

Le Web sémantique est construit à partir de Web actuel, c'est une nouvelle génération. Le Web actuel est un trésor immense des connaissances et d'informations. Le Web sémantique hérite de ce trésor, et permet aussi à des machines d'accéder à ce trésor et de l'exploiter. En effet, le Web sémantique est un Web sur lequel les internautes et les chercheurs portent beaucoup d'espoirs. Il aura pour mission de donner une signification aux données et permettre aux machines d'analyser et de comprendre les informations qui circulent dedans. La capacité de prendre en compte la sémantique constitue la différence principale entre le Web de demain et celui d'aujourd'hui (Dardailler, 2002).

Les machines et les agents logiciels pourraient comprendre les informations contenues dans ces ressources et aider les utilisateurs à exécuter et compléter leurs tâches, leurs requêtes de façon plus automatique et plus efficace. Ils pourraient raisonner sur des connaissances déjà existantes dans le Web pour fournir des informations plus précises, des services plus évolués qui s'adaptent mieux aux besoins des utilisateurs. Dans tous les domaines de connaissances, le Web sémantique vise à fournir un moyen d'échange d'information et d'une manipulation de savoir en partageant les ressources entre les différents acteurs d'un domaine (Chiaramonti, 2003).

II.3.2. Architecture du Web sémantique

Le Web sémantique comprend un certain nombre de technologies organisées en couches interdépendantes. Dans ces différentes couches, nous pouvons citer les métadonnées, les ontologies, ainsi que la logique et l'inférence (Guillermo, 2010). La figure suivante illustre ces différentes couches :

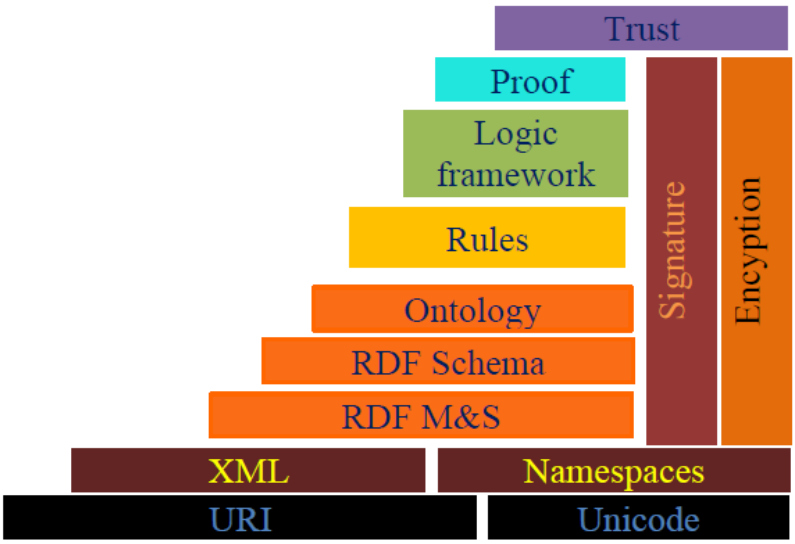

Figure II.3 : Architecture du Web sémantique (Berners-Lee, 2001).

Les différentes couches sont les suivantes (Guillermo, 2010):

- **URI (Uniform Resource Identifier)**: permet d'identifier les ressources sur les réseaux.
- **XML (Extensible Markup Language)** : le langage extensible de balisage est une simplification du langage normalisé de balisage généralisé SGML (Standard Generalized Markup Language). Son objectif initial est l'interopérabilité, de faciliter l'échange automatisé de contenus entre systèmes d'information hétérogènes et particulièrement adapté pour l'envoi de documents sur le Web.
- **XML-Schéma ou XMLS** : Schéma XML est un langage de définition à syntaxe XML, il permet de construire des documents XML suivant un vocabulaire et une structure hiérarchique spécifique. Appliqué pour la description des structures de données des documents XML.
- **RDF (Resource Description Framework) :** est un modèle de données doté d'une syntaxe à balises qui permet de représenter des objets ou des ressources ainsi que des relations entre ces objets sous forme de triplets. Chaque triplet RDF est une association de (sujet, prédicat, objet) ou le sujet représente la ressource à décrire, le prédicat représente un type de propriété applicable à cette ressource et l'objet représente une donnée ou une autre ressource.
- **RDF-Schéma ou RDFS :** est un vocabulaire qui permet de décrire les classes et propriétés pour des ressources RDF. Ce langage extensible de représentation des connaissances appartient à la famille des langages du Web sémantique et fournit des

éléments de base pour la définition d'ontologies ou vocabulaires destinés à structurer des ressources RDF.

- **Ontologies et Logique :** ces couches concernent la définition de langages ontologiques comme OWL (Ontology Web Language) qui s'appuie sur la logique de description. Le langage OWL est préconisé comme standard par le W3C consortium pour modéliser des ontologies. Il permet de modéliser les informations d'un domaine afin de faciliter l'échange d'informations par des processus automatisés.
- **Preuve :** les langages logiques permettent la mise en œuvre d'outils de raisonnement. Ces outils sont disponibles pour des langages comme OWL et permettent par exemple de tester la cohérence des informations, les classiers, …etc. Les modèles à base de règles comportent également des moteurs d'inférence qui permettent d'inférer des informations à partir des informations décrites.
- **Confiance :** la couche confiance est située au haut de la pyramide. Elle concerne l'utilisation de signatures numériques et d'autres types de connaissances afin de garantir la fiabilité et l'origine des informations, par des recommandations d'agents de confiance, de la notation, des organismes de certification et des organismes de consommateurs.

II.3.2.1. Les métadonnées

Le Web sémantique est un moyen d'échange et de partage des ressources grâce à la représentation sémantique du contenu. Ces ressources sont décrites par des informations structurées additionnelles "les métadonnées". Les métadonnées sont des *« données sur les données »* (Cardoso, 2006). L'utilisation des métadonnées permet de décrire le contenu de la ressource afin de rendre son contenu exploitable. Les métadonnées peuvent exister sur différents niveaux distincts. Les informations décrivant la syntaxe, la structure et le contexte sémantique. La figure suivante illustre la représentation faite par Sheth (Sheth, 2003), des différents types de métadonnées :

- Les métadonnées syntaxiques sont la forme la plus simple de métadonnées et décrivent l'information non contextuelle sur le contenu en fournissant des informations très générales, en associant des étiquettes. Certaines métadonnées syntaxiques peuvent être la taille du document, l'emplacement ou la date de création.
- Les métadonnées structurelles fournissent des informations concernant la structure du contenu et décrivent comment les éléments sont assemblés ou arrangés.

- Les métadonnées sémantiques ajoutent les relations, les règles et les contraintes aux métadonnées syntaxiques et structurelles. Elles fournissent un contexte pour l'interprétation des informations basées généralement sur un modèle de métadonnées d'un domaine spécifique ou une ontologie. Ces métadonnées permettent aux applications de comprendre la signification réelle des données.

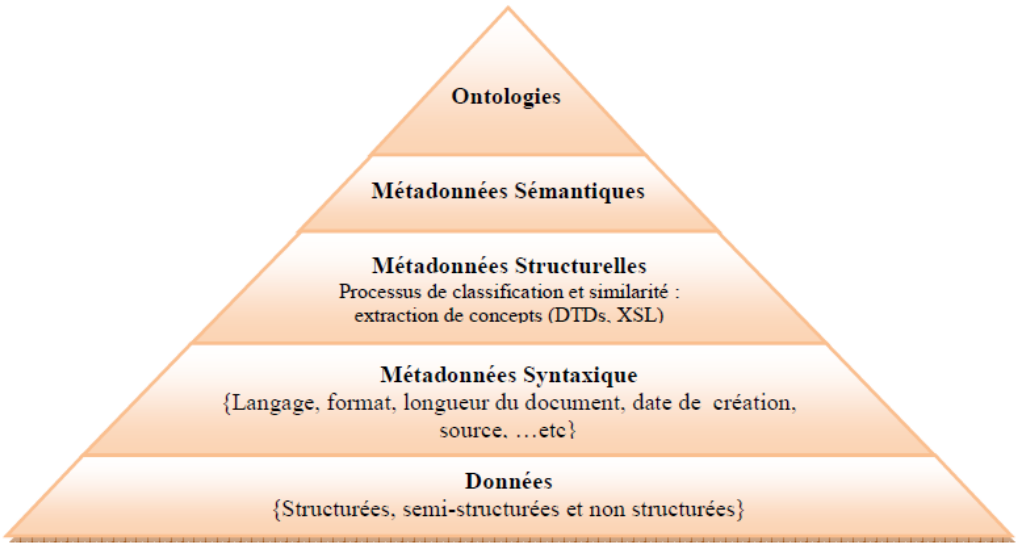

Figure II.4 : Types de métadonnées et des annotations sémantiques (Sheth, 2003)

II.3.2.2. Les ontologies

Il est extrêmement difficile de parler du Web Sémantique sans utiliser le mot ontologie. Plusieurs définitions de l'ontologie ont été proposées. Neches et ses collègues (Neches, 1991) définissent l'ontologie comme suit : *« Une ontologie définit les termes de base (concepts, entités, attributs, processus, etc.) et les vocabulaires d'un domaine et des relations qui permettent de définir des extensions des vocabulaires »*. Cette définition indique qu'une ontologie inclut non seulement les termes et les vocabulaires qui sont définis explicitement mais encore les termes qui peuvent être inférés en utilisant des relations ou des règles.

Thomas Gruber (Gruber, 1993) donne la définition suivante : *« Une ontologie est une spécification formelle explicite d'une conceptualisation partagée »*. Le terme « *conceptualisation* » réfère à un modèle abstrait d'un phénomène dans le monde, en ayant identifié les concepts appropriés à ce phénomène. La notion « *partagée* » réfère au fait qu'une ontologie capture la connaissance consensuelle, c -à-d, non réservée à quelques individus, mais partagée par un groupe.

Une ontologie peut être vue comme un treillis de concepts et de relations entre ces concepts destinés à représenter les objets du monde sous une forme compréhensible aussi bien par les hommes que par les machines. Une ontologie est constituée de concepts et des relations ainsi que des propriétés et des axiomes (Bahloul, 2006).

- ✓ ***Les concepts*** sont des notions (ou objets) permettant la description d'une tâche, d'une fonction, d'une action, d'une stratégie ou d'un processus de raisonnement, etc. Ils peuvent être abstraits ou concrets, élémentaires ou composés, réels ou fictifs.
- ✓ ***Les relations*** sont les liens organisant les concepts de façon à représenter un type d'interaction entre les concepts d'un domaine. Elles sont formellement définies comme un sous ensemble d'un produit de n ensembles. C'est à dire R: C1xC2x … xCn. Des exemples de relations binaires sont : sous-concept-de, connect-à, sorte de, etc.
- ✓ ***Les propriétés*** sont des restrictions des concepts ou des relations.
- ✓ ***Les fonctions*** sont des cas particuliers de relations dans lesquelles le N ième éléments de la relation est unique pour les n-1 précédents. Formellement, les fonctions sont définies ainsi F: C1xC2x … xCn-1,Cn.
- ✓ ***Les axiomes*** de l'ontologie permettent de définir la sémantique des termes (classes, relation), leurs propriétés et toutes contraintes quant à leur interprétation. Ils sont définis à l'aide de formules bien formées de la logique du premier ordre en utilisant les prédicats d'ontologie.
- ✓ ***Les instances*** sont utilisées pour représenter des éléments.

II.4. Les services Web sémantiques

Cette section est réservée à la présentation des objectifs des services Web sémantiques ainsi que les différents langages de descriptions sémantiques dédiés aux Web services.

II.4.1. Présentation et objectifs des services Web sémantiques

Partage, Echange, Réutilisation ; Ces trois mots définissent à eux seuls le Web. Le Web sémantique constitue une prolongation, et une sorte de révolution de fond du Web actuel qui permet une définition non ambiguë de l'information, pour favoriser une meilleure coopération entre humain et machine. Il permet de s'ouvrir à de nouvelles possibilités d'automatisation d'une grande quantité d'information sur le Web. Proclamé technologie du futur, en 2001, par son créateur Tim Berners-Lee, le Web sémantique propose une nouvelle plateforme permettant une gestion plus intelligente du contenu, à travers sa capacité de manipuler les ressources sur la base de leurs sémantiques. En réalité, l'intégration de la

sémantique au Web n'est pas une nouvelle idée mais au contraire, elle est née avec le Web (Falquet, 2001). Dans l'architecture en couche des technologies du Web sémantique, présentée précédemment, on a présenté les technologies Web de base. Puis, on a parlé des services Web comme un moyen qui fourni une plateforme interopérable, pour l'intégration des applications en utilisant des technologies Web (voir la figure II.5).

Le Web sémantique constitue le point de départ pour le développement de services Web intelligents. En effet, non seulement l'humain pourra partager, échanger et réutiliser la connaissance et l'information qui est disponible sur le Web mais en plus, il pourra le faire plus vite et avec l'aide des machines. En effet, la sémantique et la structure des données requirent une représentation de la sémantique compréhensible et échangeable par les machines (Bruijn, 2007).

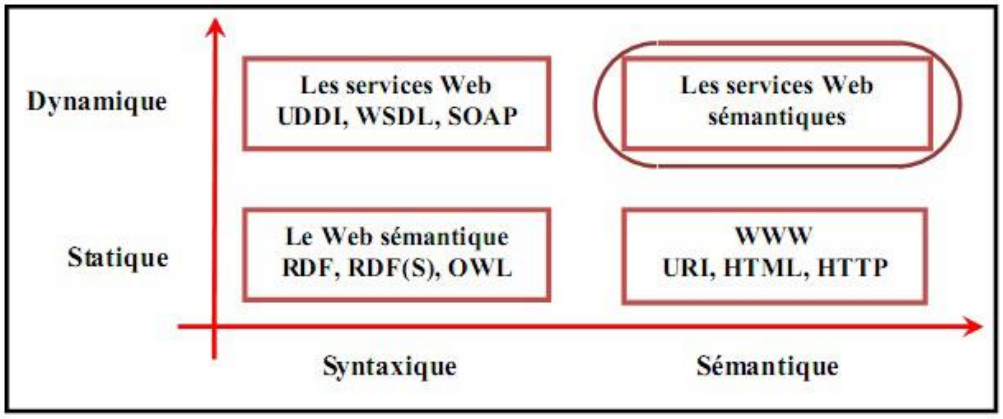

Figure II.5 : L'évolution du Web (Cardoso, 2007).

Le terme services Web sémantique (SWS) est réservé pour l'automatisation des tâches d'utilisation des services Web, tels que : la publication, la découverte, la composition, etc. Ils se trouvent à la convergence de deux domaines de recherche importants concernant les technologies de l'Internet ; le Web sémantique et les services Web (Cardoso, 2007).

Cette tâche de convergence est accomplie en rendant les services Web auto-exploitables par machines, et de réaliser l'interopérabilité entre les applications via le Web en vue de rendre le Web plus dynamique. L'objectif est non pas de permettre aux machines de se comporter comme des êtres humains, mais de développer des langages pour représenter les informations d'une manière traitable, représentable et intelligible par les machines, afin, d'améliorer les rapports des utilisateurs avec le Web. Il semble donc nécessaire de tendre vers des services intelligibles pour des machines, c'est le concept de service Web sémantique (Kadima, 2003).

De manière générale, l'objectif visé par la notion de SWS est de créer un Web sémantique de services dont les propriétés, les capacités, les interfaces et les effets sont décrits de manière non ambiguë et exploitable par des machines, et ce en utilisant les couches techniques sans pour autant en être conceptuellement dépendants (Daconta, 2003).

II.4.2. Approches proposées pour les services Web sémantiques

Le service Web sémantique ne peut pas exister sans le développement d'un ensemble de standards et architectures universels, comme le Web actuel n'aurait pu exister sans le HTTP et le HTML. Pour pouvoir exprimer de la sémantique avec les informations du Web, beaucoup de langages ont été élaborés. Nous allons donc présenter les objectifs et les fonctionnalités des principaux langages consacrés aux services Web sémantiques.

II.4.2.1. WSDL-S

WSDL-S est un langage de description sémantique des services Web. Une description WSDL-S de service Web est une description WSDL augmentée d'annotation sémantique. Les annotations sémantiques peuvent être des références à des concepts définis dans des ontologies externes. WSDL-S définit un modèle sémantique pour capturer les termes et les concepts utilisés pour décrire et représenter la connaissance, cette sémantique est ajoutée en deux étapes (Cerami, 2002) :

- ✓ La première étape consiste à faire référence, dans la partie définition de WSDL, à une ontologie dédiée au service à publier ;
- ✓ La deuxième étape consiste à annoter les opérations de la définition WSDL de sémantique, en ajoutant deux nouvelles balises ; la balise « *Action* » qui permet de représenter l'action de l'opération et la balise « *Contrainte* » qui représente les prés et post conditions d'une opération.

Quatre rôles du modèle sémantique sont distingués :

- **InputSemantics :** le sens des paramètres d'entrées ;
- **OutputSemantics :** le sens des paramètres de sortie ;
- **Precondition :** un ensemble d'états sémantiques qui doivent être vrais afin d'invoquer une opération avec succès ;
- **Effect :** un ensemble d'états sémantiques qui doivent être vrais après qu'une opération accomplisse son exécution.

II.4.2.2. OWL-S

OWL-S est une ontologie et un langage pour les services Web développé dans le cadre du projet DAML. OWL-S se base sur OWL qui permet de décrire des ontologies. Ainsi, OWL-S est une ontologie OWL particulière. OWL-S succède aux travaux antérieurs de DAML-S qui étaient basés sur DAML+OIL. OWL-S décrit un service à l'aide des trois classes suivantes (voir la figure II.6) (Chappell, 2002) (Chatterjee, 2003) :

- **ServiceProfile :** chaque instance de « *Service* » produit zéro ou plusieurs profils de services. Un service Profile exprime « *Que fait un Service* », aux fins des avertissements et sert comme un Template pour les requêtes de services, permettant ainsi la découverte et leurs arrangements. OWL-S fournie cette classe pour décrire un service Web. Cette classe « *ServiceProfile* » spécifie trois informations :
  - ✓ **Le fournisseur du service :** cette information précise les données nécessaires pour identifier et contacter le fournisseur du service Web.
  - ✓ **La description fonctionnelle :** elle spécifie ce que l'on peut attendre du service en termes d'entrées attendues et de résultats produits en sortie. Les transformations d'informations sont représentées par des « *Inputs* » et des «*Outputs*». Le changement d'état du monde réel causé par l'exécution du service est représenté par « *preconditions* » et « *effects* » qui sont les préconditions et les postconditions de son exécution.
  - ✓ **Propriétés additionnelles :** plusieurs propriétés sont utilisées pour qualifier le service Web. La première est la catégorie du service Web. La seconde est la qualité de service Web « *QoS* ». Enfin, le service Web peut fournir une liste de paramètres de façon libre.
- **ServiceModel :** définit le fonctionnement du service Web. Les services Web peuvent être modélisés avec OWL-S en tant que processus. La classe ainsi définie est «*Process*», qui est une sous classe de « *ServiceModel* ». Pour décrire un processus, on spécifie ses entrées et sorties. Les transitions d'un état à un autre sont décrites par les préconditions et les effets de chaque processus. Un model peut être décrit par le lien « *descriptedBy* », un model de service qui expose comment un service fonctionne.
- **ServiceGrounding :** définit les détails techniques permettant d'accéder au service Web publié, tels les protocoles, les URIs, les messages envoyés, … etc. Pour cela, il fournit les détails pour connecter la spécification abstraite et la spécification concrète. Il est effectivement nécessaire d'avoir une combinaison entre le grounding et WSDL. Le

fonctionnement du grounding est assuré si et seulement si, les deux entités (WSDL et Grounding) sont présentes et correspondent.

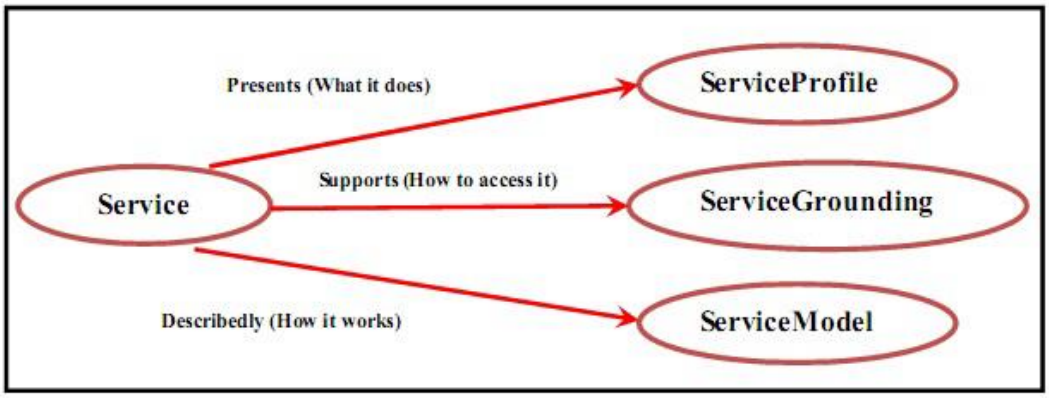

Figure II.6 : Structure générale de l'ontologie OWL-S (Chatterjee, 2003).

II.4.2.3. IRS-II

IRS-II est une architecture pour les services Web sémantiques. IRS-II est basée sur la structure UPML, où diverses ontologies sont définies (Gesnu, 2003) (Lopes, 2005) :

- **Ontologie du domaine (Domain Model) :** permet de décrire le domaine d'une application ;
- **Ontologie de tâche à résoudre (Task Models) :** fournit une description générique de la tâche à résoudre, spécifie les types d'entrées et sortie, le but à atteindre et les préconditions à satisfaire ;
- **Ontologie des méthodes de résolution d'un problème (Problem Solving Methods) :** sépare la description de ce qu'un service fait des paramètres et des contraintes d'une mise en oeuvre particulière ;
- **Liens (bridges) :** permettent la correspondance entre les différents modèles d'une application.

Les principaux composants de l'architecture IRS-II sont : le serveur IRS-II (IRS-II server), éditeur de services (IRS-II Publisher) et la partie client (IRS-II Client). Ces trois composants interagissent entre eux via le protocole SOAP.

Le serveur IRS-II contient les descriptions des services Web sémantiques. Ces descriptions sont faites sur deux niveaux. Au niveau connaissance, une description est sauvegardée selon la structure UPML des tâches, PSM et l'ontologie du domaine. De plus, deux types de mise en correspondances sont utilisés pour lier les descriptions aux niveaux connaissances à un service Web spécifique.

II.4.2.4. WSFM

WSFM comprend quatre éléments principaux (Gardien, 2002) (Dallons, 2004) :

- ✓ Des ontologies qui fournissent la terminologie utilisée par les autres éléments ;
- ✓ Un répertoire d'objectifs qui définit les problèmes qui doivent être résolus par les SW ;
- ✓ Des descriptions des services qui définissent les différents aspects liés aux SW ;
- ✓ Des médiateurs qui sont en charge des problèmes d'interopérabilité.

L'implémentation de WSMF est partagée en deux projets : le projet SWWS; et le projet WSMO. L'objectif de SWWS est de définir une structure de description et de découverte de services Web, ainsi qu'une plateforme de médiation pour services selon une architecture conceptuelle. Le projet WSMO permettra de raffiner WSMF, en plus, du développement d'ontologie formelle de service et un langage pour les SW sémantiques (Casati, 2000).

II.4.2.5. WSMO

Le WSMO est un projet de l'union européenne qui constitue un cadre compréhensible pour SESA et défini un model conceptuel avec un langage de spécification, comme, il fourni une implémentation avec plusieurs outils. L'implémentation WSMX fournie un environnement de développement et d'exécution pour SESA à base de WSMO. L'approche WSMO définie les ontologies, les services Web, les buts et les médiateurs comme ses éléments de haut niveau avec un model conceptuel qui prend en charge ces derniers. Ce model conceptuel a pour but la structuration des annotations sémantique des services.

WSMO permet la description des services, en considérant les aspects suivants (Newcomere, 2004) :

- ✓ Les propriétés non fonctionnelles incluent des propriétés telles que la performance, la fiabilité, la sécurité, la robustesse et la scalabilité du service Web ;
- ✓ La fonctionnalité du service est décrite en termes de préconditions, postconditions, hypothèses et effets ;
- ✓ Une interface de description d'un service Web est constituée d'informations telles que les erreurs gérées par le service, une description d'orchestration si le service fait appel à d'autres services, une description de la conversation du service appelée échange de message, et des stratégies de compensations utilisées dans le cas ou certaines opérations exécutées par le service doivent être annulées ;

- ✓ Le grounding spécifie les informations concrètes pour l'accès au service Web. Il est clair que de nombreux aspects définis dans cette ontologie sont également couverts par OWL-S. Par exemple, les préconditions, postconditions, hypothèses et effets dans WSMO. D'autre part, les deux ontologies se basent sur WSDL pour la liaison. Un autre point commun est la description de la conversation d'un service dans les interfaces WSMO qui est également décrite dans le process Model de OWL-S. Ceci dit qu'il existe quelques différences entre les deux approches.

Contrairement à OWL-S qui n'exige aucun style architectural, l'approche proposée par WSMO est basée sur l'architecture WSMF. Cette architecture définit notamment la notion de médiateurs. Les médiateurs sont des composants logiciels utilisés par les SW pour répondre à des problèmes d'interopérabilité tels que l'incompatibilité des types (Burstein, 2004).

II.5. Qualité de services Web

Avec la prolifération des services Web, la notion de qualité de service (QoS) émerge aujourd'hui et prend de plus en plus une grande importance pour les fournisseurs de service aussi bien que pour les clients de service. Il n'existe pas de consensus sur la définition de la qualité de service.

Définition 1 : la recommandation ITU-X.9028[1] définit la QoS comme *« un ensemble d'exigences dans le comportement collectif d'un ou plusieurs objets ».*

Définition 2 : dans le contexte des technologies de l'information et multimédia, la QoS a été définie par Vogel et al. comme *« l'ensemble des caractéristiques quantitatives et qualitatives d'un système multimédia, nécessaires pour atteindre la fonctionnalité requise par l'application »*. (Vogel, 1995)

Définition 3 : selon le standard ISO 8402 (ISO, 2000), la qualité de service se définit comme: *« l'ensemble des propriétés et caractéristiques d'un produit ou d'un service qui lui confère l'aptitude à satisfaire des besoins exprimés ou implicites »* (Troispoux, 2007).

Définition 4 : « *la qualité de service représente un concept abstrait et diffus, à multiples facettes »* (Bressoles, 2003).

Dans l'objectif d'assurer une qualité de service adaptée à l'utilisateur de nombreuses propriétés non fonctionnelles doivent être exploitées durant le déploiement ou l'exécution des SW. Ces propriétés de QoS révèlent des caractéristiques diverses et s'expriment sous

[1] The International Telecommunication Union (ITU) standard X.902, Information technology Open distributed processing - Reference Model.

différentes formes, car elles peuvent être relatives aux services mêmes, notamment en terme (i) de performance (fiabilité, temps d'exécution attendu, etc.), (ii) au contexte d'exécution (temps de latence, débit, etc.) ou (iii) aux besoins de l'utilisateur (qualité du résultat, coût économique du calcul, etc.) (Acher, 2008).

Puisque les services Web sont fournis par des tiers et être appelés dynamiquement au-dessus de l'Internet, leur QoS peut varier considérablement. Par conséquent il est important d'avoir un cadre pour capturer le QoS donné par le fournisseur et le QoS exigé par le client, et finalement la correspondance entre les deux en découvrant le meilleur service Web selon la QoS exigée (Shuping, 2003). L'ISO 8402 (ISO, 2000) décrit la qualité comme *« totalité de dispositifs et de caractéristiques d'un produit ou d'un service qui concernent sa capacité de satisfaire aux besoins indiqués ou implicites»*. Nous rejoignons la définition de la qualité de service présenté par Shuping dans (Shuping, 2003), elle est définie comme ensemble d'attributs non fonctionnels qui peuvent effectuer la qualité du service offert par un service Web. Dans le modèle de Shuping (Shuping, 2003), Il y a beaucoup d'aspects de QoS importants pour des services Web. Nous commençons à les organiser en catégories de QoS. Chaque catégorie doit avoir un ensemble de paramètres ou de mesures quantifiables. Pour faciliter la description, les catégories sont groupées dans différents types.

II.5.1. Qualité de service liée au temps d'exécution

- **Évolutivité:** la capacité d'augmenter la capacité informatique du système informatique du fournisseur de services et de la capacité du système de traiter plus d'opérations ou de transactions en période donnée.
- **Capacité :** limite des demandes concourantes d'exécution garantie.
- **Performance :** une mesure de la vitesse en remplissant une demande de service. Elle est mesurée par ((Gunther, 1998) cité dans (Shuping, 2003)):
  - **Temps de réponse :** le maximum (moyen ou minimum) de temps nécessaire pour remplir une demande de service (liées à la capacité).
  - **Latence:** le temps pris entre l'arrivée de la demande de service et la satisfaction de la demande.
  - **Sortie :** le nombre de demandes de service réalisées sur une période de temps. La sortie est liée à la latence et à la capacité.
- **Exécution :** une mesure de la vitesse en accomplissant une demande de service.
- **Fiabilité :** la capacité d'un service de remplir ses fonctions requises dans des conditions indiquées pendant une période spécifique.

- **Flexibilité / Robustesse:** c'est la mesure dans laquelle un service peut fonctionner correctement dans la présence d'entrées invalides, incomplets ou contradictoires.
- **Manipulation d'exception:** puisqu'il n'est pas possible que le concepteur de service spécifie tous les résultats et solutions de rechange possibles (particulièrement avec de divers cas spéciaux et possibilités imprévues), des exceptions peuvent être prévues. La manipulation d'exception est comment le service manipule ces exceptions.
- **Exactitude:** définit le taux d'erreur produit par le service.

II.5.2. Qualité de service liée aux transactions

- **Intégrité:** des transactions peuvent être groupées dans une unité afin de garantir l'intégrité des données opérées par ces transactions. L'unité peut être réussie où toutes les transactions en unité «commettent» ou tous « roulez en arrière » à leur état original en cas d'échec de transaction. Ceci est décrit par les propriétés ACIDES : Atomicité (s'exécute entièrement ou pas du tout), uniformité (maintient l'intégrité des données), isolement (individu les transactions fonctionnent comme si aucune autre transaction n'est présente) et longévité (les résultats sont persistants)).

II.5.3. Qualité de service liée a la gestion de la configuration et coût

- **Réglementation:** il s'agit d'une mesure de la façon dont le service est conforme à la réglementation.
- **Coût:** c'est une mesure du coût entraîné en demandant le service.
- **Norme soutenue :** une mesure de si le service est conforme aux normes (par exemple normes propres à l'industrie).
- **Cycle de stabilité/changement:** une mesure de la fréquence du changement s'est rapportée au service en termes de son interface et/ou exécution.
- **Perfection:** une mesure de la différence entre l'ensemble de dispositifs spécifiques et l'ensemble de dispositifs mis en application.

II.5.4. Qualité de service liée à la sécurité

- **Authentification:** comment le service authentifie-t-il les directeurs (des utilisateurs ou d'autres services) qui peuvent accéder au service et aux données ?
- **Autorisation:** comment le service autorise-t-il les directeurs de sorte que seuls eux peuvent accéder aux services protégés?

- **Confidentialité:** comment le service traite-t-il les données, de sorte que seulement les principaux autorisés puissent accéder ou modifier aux données ?
- **Responsabilité** : le fournisseur peut-il être prise responsable de leurs services ?
- **Traçabilité et contrôlabilité:** est il possible de tracer l'histoire d'un service quand une demande a été entretenue.
- **Chiffrage de données:** comment le service chiffre-t-il des données ?
- **Non-répudiation:** le principal ne peut pas refuser la demande d'un service ou des données après le fait.

II.6. Découverte des services Web

La découverte des services Web est l'opération qui permet de *« localiser une description de service Web exploitable et inconnue par les machines sur un réseau, et qui répond à certains critères fonctionnels ou/et non fonctionnels »*[2]

II.6.1. Problématique de la découverte des services Web

L'objectif principal de la technologie des services Web est de fournir l'intégration et l'interaction dynamique des systèmes hétérogènes, et de ce fait, faciliter la coopération rapide et efficace entre les différentes entités dans un environnement collaboratif (Emekci, 2004), (Sahin, 2005), (Essafi, 2005).

Pour atteindre ces objectifs, les services Web doivent agir automatiquement avec le minimum d'intervention humaine; ils doivent être capables de découvrir d'autres services qui ont des capacités particulières et réalisent des tâches précises (Essafi, 2005). A cause de ces contraintes, la découverte des services Web est conçue comme un problème critique depuis la naissance de ce paradigme. La première génération des travaux de recherche effectuée sur les architectures des services Web, proposent des solutions basées sur les méthodes de découverte centralisées (comme UDDI) où les services Web sont décrits par les interfaces (signatures) de leurs fonctions. Dans une telle architecture, les fournisseurs des services Web publient les capacités et les fonctionnalités des ces services dans un enregistrement central. Cependant, ces méthodes souffrent d'un nombre de problèmes classiques connus dans les systèmes centralisés comme le seul point centralisé (point d'échec) et le coût élevé de la maintenance. De plus, ces méthodes ne garantissent pas

[2] http://www.w3.org/TR/2004/NOTE-ws-gloss-20040211/

l'évolutivité (scalabilité) requise pour soutenir un environnement flexible et dynamique (Mandreoli, 2007), (Hu, 2005).

Les paragraphes suivants présentent trois types de solutions centralisées ainsi que les inconvénients et les problèmes rencontrés au niveau de chaque solution.

II.6.1.1. Méthodes centralisées de découverte

Les méthodes de découverte centralisées présentent la première génération des travaux proposés dans ce domaine. La première initiative industrielle et académique issue pour la découverte des services Web est la recommandation UDDI. Ce dernier a été utilisé de deux façons différentes, tant qu'un annuaire universel ou privé.

II.6.1.1.1. Annuaire universel

La première version de la spécification UDDI a été publiée en septembre 2000 par Ariba, IBM et Microsoft (Chauvet, 2002). La philosophie de UDDI dans cette version est de centraliser les points d'entrées des SW, c'est à dire la création d'un annuaire universel pour les SW fournis par les différents propriétaires (entreprises ou individus). Cet annuaire universel recueillant les inscriptions des fournisseurs de services Web permet à un chacun d'effectuer des recherches pour trouver les services particuliers dont il a besoin (Chauvet, 2002). Un exemple de ce type d'annuaire est celui qui regroupe les annuaires UDDI des opérateurs Microsoft, IBM et XMethods (il été accessible à partir de l'adresse suivante : http://soapclient.com/index.html). Ce site fournit aux utilisateurs la possibilité de chercher des services Web en utilisant des différents critères : Business Names (Business Entity), Service Names (Business Service), Service Types (tModel) et autres. Cette proposition fut l'objet d'un certain nombre de et pose deux grands problèmes (Modi, 2002) :

- ✓ L'absence de sécurité et de fiabilité pour les échanges B2B particulièrement dans le cas de paiement électronique.
- ✓ Le manque de modération dans les enregistrements UDDI existants : les modérateurs sont responsables de régler les teneurs des enregistrements qu'ils modèrent (des compagnies posant comme fournisseurs des services qu'ils ne présentent pas).

A cause de ces problèmes, cet annuaire universel a été supprimé par ses opérateurs en 2006. D'autres solutions ont été développées pour le remplacer. Dans ce qui suit, nous présentons les annuaires privés et les moteurs de recherche.

II.6.1.1.2. Annuaires privés

Comme UDDI a obtenu le soutien des entreprises et la plupart des fournisseurs des services Web, son utilisation n'était pas limitée aux échanges B2B, mais aussi dans le domaine du commerce électronique (B2C : Business to Customers). C'est dans ce contexte que la deuxième version d'UDDI a été publiée en juin 2001 ayant pour but de corriger quelques problèmes apparaissant dans la première version. Cette version est moins lourde à mettre en oeuvre, elle est en plus adaptée aux échanges B2B privés (sur un Extranet), au commerce électronique (dans un contexte B2C) et même à l'intégration des applications à l'intérieur de l'entreprise. Dans le contexte B2C, une entité commerciale implémente un ensemble de registres UDDI privés ou semi-privés. Généralement, l'entité commerciale a certaines règles commerciales ou des intérêts à suivre pour personnaliser les résultats de la découverte des services Web. Par exemple, une entité commerciale, qui possède un UDDI privé, veut toujours contrôler l'affichage des résultats correspondants aux requêtes des consommateurs (Tan, 2006).

Néanmoins pour les autres initiatives qui ont vu le jour, comme WSIL (Web Services Inspection Language) (WSIL) et ebXML (e-business eXtensible Markup Language) (ebXML), UDDI reste la spécification la plus utilisée par les industriels, et le standard le plus cité dans les travaux de recherche. Cependant, même dans sa deuxième version, UDDI est incapable de supporter les environnements dynamiques et flexibles tout particulièrement dans le cas de la composition dynamique des services Web. En plus, UDDI offre seulement un contenu technique à cause de la publication des services Web à l'aide de la description WSDL.

II.6.1.1.3. Moteurs de recherches

A cause de la description technique des services Web, il n'est pas évident de retrouver ces derniers en utilisant un moteur de recherche publique comme Google. Un service Web est un composant logiciel encapsulant des fonctionnalités métiers. Il est considéré comme une boite noire (on ne connaît pas en détail son implémentation) et n'expose que des données techniques décrites par son interface WSDL. Dans la littérature, il n'existe qu'une seule initiative qui offre un moteur de recherche spécifique dédié à la découverte des services Web, c'est « *service finder* » (Finder). Les initiateurs de service finder visent à développer une plate-forme de découverte des services dans laquelle les services Web sont intégrés dans un environnement Web 2.0. Cependant, dans sa première version béta lancée

en 2008 (actuellement disponible à http://demo.service-finder.eu/), ce moteur de recherche pose presque les même problèmes déjà reconnus dans l'UDDI universel. Pour le moment il ne supporte pas la découverte automatique (en temps d'exécution) effectuée par des machines. Néanmoins, service finder a connu des améliorations sur le plan sémantique par rapport à l'UDDI universel en utilisant quelques techniques du Web 2.0.

II.6.1.2. Discussions

Les méthodes centralisées présentées dans la section précédente souffrent de trois problèmes principaux : la prise en compte de la sémantique pour la découverte des services Web, le non considération de la Qualité de Service (QoS) et la composition dynamique des services Web (Tamrout, 2006).

II.6.1.2.1. Problème de la sémantique

WSDL et UDDI ne permettent pas à des services Web d'interagir de manière intelligente. Ils ne décrivent pas suffisamment les connaissances de manière à les rendre exploitables par des machines. Dans ce contexte, la sémantique est considérée comme un élément primordial qui permet de représenter et de manipuler des connaissances de natures différentes : structurelles, fonctionnelles et même opérationnelles. De ce fait, en plus de la recherche par mots-clés, la découverte de services se fera selon les différents concepts qui leurs sont associés et le contexte dans lequel ils doivent être compris. Dans une telle situation, la découverte des services Web est caractérisée par des capacités de raisonnement qui s'appuient sur une description formalisée et sur la mise en relation des différentes sources d'information (Tamrout, 2006).

II.6.1.2.2. Problème de qualité de service (QoS)

Les services Web proposés sur Internet par différents fournisseurs sont nombreux. Dans une telle situation, il n'est pas évident de découvrir un service Web qui réponde à un besoin d'un demandeur. Pour effectuer le bon choix entre les différents services Web candidats, diverses propriétés de qualité de service telles que la disponibilité, la performance et la fiabilité sont nécessaires. Généralement, les fournisseurs publient des descriptions concernant les propriétés fonctionnelles des services qu'ils proposent. Par contre, des informations, comme le temps de réponse et la disponibilité des services, ne sont jamais fournies par ces fournisseurs. Par exemple, dans un environnement des

échanges entre partenaires (B2B), il est important d'en trouver le meilleur. Il devient donc nécessaire de pouvoir extraire et exploiter les services Web pertinents parmi ceux qui ont été collectés (Tamrout, 2006).

II.6.1.2.3. Problème de composition dynamique des services Web

Il existe deux types de méthodes pour la composition des services Web: orchestration et chorographie. Dans ces deux catégories, nous pouvons distinguer deux techniques permettant la mise en place d'une composition : statique et dynamique.

Dans les méthodes centralisées, ce problème est plus raisonnable parce qu'UDDI ne propose pas de solutions pour les compositions dynamiques de services. Il n'a pas la capacité d'assembler et d'orchestrer des services Web à cause de la description de ceux proposés par UDDI ; cette dernière n'intègre que très peu d'aspects sémantiques.

II.6.2. Découverte distribuée des services Web

D'autres méthodes de découverte et de composition des services Web ont été proposées pour résoudre les limites des méthodes centralisées. Ce type de méthodes est caractérisé par la distribution du mécanisme de publication et de découverte. Elles ont pour objectif principal la composition dynamique des services Web et il existe deux techniques de bases qui sont utilisées pour implémenter les méthodes décentralisées : les Systèmes Multi Agents et les réseaux P2P.

II.6.2.1. Utilisation des SMA (Systèmes Multi Agents)

Le couplage des services Web et des systèmes multi agents est bidirectionnel. D'une part les agents peuvent utiliser les services Web pour exposer leurs capacités au monde extérieur. Dans ce cas, les SMA hétérogènes utilisent les services Web pour raison d'interopérabilité (Seghrouchni, 2004). D'autre part, les SMA sont utilisés généralement pour implémenter la composition chorographique des services Web.

Dans ce type d'architecture, chaque agent est responsable d'un ou de plusieurs services Web. Sont but est la négociation et la communication d'une manière intelligente avec les autres agents, afin d'accomplir un objectif commun à travers la composition de plusieurs services (Vadivelou, 2011), (Bourdon, 2007), ces derniers pouvant appartenir à différents fournisseurs. Cependant, les SMA sont généralement utilisés pour composer les services Web dans un environnement coopératif fermé (pour échanges B2B par exemple) (Brahimi,

2009). Habituellement, ce type de solutions ne propose pas un mécanisme de découverte des services Web. L'objectif principal des ces solutions est la proposition d'une architecture SMA ainsi que des protocoles et des algorithmes pour composer dynamiquement les services Web.

II.6.2.2. Utilisation des réseaux P2P

Le deuxième type des méthodes décentralisées dédiées à la découverte et la composition des services Web permet d'implémenter les systèmes P2P. Ces derniers ont été impliqués dans ce contexte parce qu'ils fournissent une alternative évolutive aux systèmes centralisés et ce en distribuant les services Web sur tous les pairs du réseau. De plus, la convergence entre les technologies du Web sémantique, des services Web et du P2P, présente des nombreux avantages, notamment pour le partage des connaissances dans les réseaux à grande échelle.

II.7. Travaux relatifs

En plus des travaux mentionnés ci-dessus (section I.6), nous allons présenter dans ce qui suit et d'une façon détaillée autres travaux de recherche proposés pour la découverte des services Web :

- WSPDS (Web Services Peer-to-peer Discovery Service) est une architecture de découverte des services Web dans un réseau P2P pur (non structuré) (Kashani, 2004). Elle est composée essentiellement de deux modules : un moteur de communication et un moteur de traitement des requêtes. De plus, elle offre une interface pour les utilisateurs externes (simples internautes ou autres applications). Le fonctionnement des WSPDS est simple : les pairs, qui implémentent cette architecture, forment un espace collaboratif pour la découverte des services Web. Lorsqu'il reçoit une requête, un pair participant utilise le moteur des requêtes pour rechercher un service Web qui répond à cette requête, il envoi la réponse (si elle existe) ou il transmet la requête aux pairs voisins et renvoi les réponses au pair demandeur. La réception et la propagation des requêtes s'effectuent en utilisant le moteur de communication.
- Dans le travail proposé à (Verma, 2005), Verma et al. présentent l'architecture METEOR-S WSDI (METEOR-S Web Service Discovery Infrastructure ou MWSDI) qui est un système distribué basé sur la typologie hybride. MWSDI est constitué d'un ensemble de Super-pairs contenant ses propres registres, et collaborant entre eux pour

répondre aux requêtes des demandeurs. Une ontologie, d'un (ou plusieurs) domaine (s), est associée à chaque registre. Ce dernier est implémenté par un annuaire UDDI qui supporte la découverte et la publication sémantiques des services Web (en utilisant une annotation sémantique).

- Une autre architecture, qui implémente la typologie hybride, est celle de GloServ (Global Service discovery architecture) (Arabshian, 2006). Cette solution est néanmoins critiquée car les pairs reliés à un Super-pair sont organisés selon le protocole P2P structuré CAN (Content Addressable Network) (Ratnasamy, 2001) et les pairs sont catégorisés selon les concepts ontologiques décrivant les services fournis. Pour gérer l'infrastructure sémantique, GloServ utilise le langage OWL-DL où chaque concept est décrit par une classe OWL avec un ensemble propriétés. Cependant, GloServ ne décrit pas suffisamment comment un pair peut joindre le réseau surtout dans le cas où un pair peut fournir des services de différents domaines avec différents concepts. De plus, GloServ ne définit pas un mécanisme de composition des SW.
- Dans (Scioscia, 2014), les auteurs présentent Mini-ME (Mini Matchmaking Engine), un matchmaker compact et raisonneur pour l'ALN (Attributive Langue avec restrictions Numériques non qualifiés) et DL (Description de Logic). Il est destiné au matchmaking sémantique pour la découverte des ressources / services dans des contextes mobiles et ubiquitaires, donc c'est un moteur d'inférence sémantique à des fins générales. L'expressivité réduite du langage logique est compensée par un niveau de mobilité accrue et fourni la qualité de la découverte de ressources. Mini-ME est appropriée pour une classe très répandu d'applications où un grand nombre de ressources à faible complexité peuvent être agrégées pour construire des services composées avec la croissante de la complexité sémantique.
- Dans le travail (Dietze, 2011), les auteurs affirment que la médiation au niveau sémantique « semantic-level-mediation » est nécessaire pour identifier les similarités sémantiques entre les représentations de SW distinctes. Les auteurs formulent et implémentent une approche de médiation fondée sur des espaces de médiation « Mediation-Spaces » (MS), qui permet la représentation implicite des similitudes sémantiques entre les descriptions SW différentes. En conséquence, compte tenu d'une approche spécifique pour les SWS et la proposition du MS, un algorithme général est implémenté pour habiliter la sélection des SW avec le calcul automatique des similarités sémantiques entre une demande de SW et un ensemble des offres de SW.

II.8. Conclusion

La découverte de services Web présente un axe de recherche émergeant. Diverses approches ont été proposées dans la littérature. L'objectif commun de ces approches était de découvrir des descriptions, non connues auparavant, de services Web décrivant certains critères fonctionnels. Au début, les descriptions des services Web étaient essentiellement syntaxiques, cependant, avec la fusion des deux technologies services Web et Web sémantique une nouvelle génération de services Web est née connue sous le nom de services Web sémantiques. Les services Web sémantiques sont des services Web dont les descriptions ont une sémantique. Avec le développement des services Web sémantiques de nouvelles approches de découverte ont été proposées. La plupart des approches sémantiques proposées étaient basées sur le calcul du niveau de correspondance sémantique entre les critères fonctionnels définis dans la description sémantique des services et ceux définis dans la requête de recherche. Cependant, vu la diversité des utilisateurs et des conditions dans lesquelles ils accèdent aux services Web, d'autres paramètres doivent êtres considérés lors de la découverte, tel que le type du dispositif utilisé (PDA, ordinateur portable, etc.), préférences de l'utilisateur, localisation de l'utilisateur, etc. Tous ces paramètres forment un contexte d'utilisation particulier.

Il s'avère nécessaire d'exploiter des connaissances sémantiques pour satisfaire l'utilisateur. Cette satisfaction peut être assurée à travers : la personnalisation (profil utilisateur). Ce dernier fait l'objet de notre prochain chapitre.

Chapitre III

Etat de l'Art sur la Personnalisation

III.1. Introduction

Au fur et à mesure que les services Web se multiplient, la difficulté de la découverte de service s'accroît. La maitrise de cette phase nécessite la personnalisation de la sélection des services Web désirés par l'utilisateur. La personnalisation est une réponse efficace au problème de la surcharge d'informations, d'une part, en injectant dans les requêtes des critères de filtrage plus restrictifs, et, d'autre part, en reformulant éventuellement les requêtes pour mieux tenir compte des centres d'intérêt et des préférences de l'utilisateur. Une sélection plus restrictive des données permet de limiter la surcharge d'informations et une reformulation de certaines parties de la requête permet de mieux cibler les résultats.

L'ensemble des données décrivant l'utilisateur et ses préférences sont souvent regroupés et stockées dans un profil utilisateur. Dans un tel contexte, la requête utilisateur exprime une demande d'information particulière tandis que le profil représente la partie relativement stable des besoins de l'utilisateur qui doit être prise en compte lors de l'évaluation de la requête (Bouzeghoub, 2007) (Kostadinov, 2007).

Ce chapitre résume les grandes lignes de l'état de l'art sur la personnalisation. Les principaux points abordés sont les systèmes de recherche d'informations personnalisés, les modèles de définition et de représentation des profils utilisateurs, les techniques de construction et d'exploitations des profils, ainsi qu'une présentation de certains travaux relatifs.

III.2. Les systèmes de recherche d'information personnalisés

Nous présentons dans ce qui suit les principaux composants d'un système de recherche d'informations (SRI) personnalisé.

III.2.1. Définition

Un système de recherche d'informations personnalisé (SRIP) est un SRI qui intègre totalement l'utilisateur tout au long du processus de recherche. Il répond ainsi de manière personnelle aux besoins en informations de chaque utilisateur.

La recherche d'informations (RI) personnalisée est une activité faisant intervenir deux entités : les caractéristiques de l'utilisateur communément appelés «profil de l'utilisateur » et le caractéristiques des documents appelés les « métadonnées des documents ».

Un SRI personnalisé inclut (Zemirli, 2008):

- ✓ Des modèles et algorithmes pour capturer et modéliser le but, les préférences et les centres d'intérêts de l'utilisateur ou un groupe d'utilisateur. Un modèle de profil est alors décrit et instancié.
- ✓ Une procédure de mise à jour du profil qui traduit son évolution dans le temps.
- ✓ Des mécanismes pour extraire les caractéristiques descriptives des documents à l'aide de métadonnées.

Le processus de personnalisation s'effectue en incluant le profil utilisateur et le profil document dans l'ensemble de étapes du processus en U de la RI (voir figure III.1). Cette personnalisation peut avoir lieu à différents niveaux du processus (Zemirli, 2008):

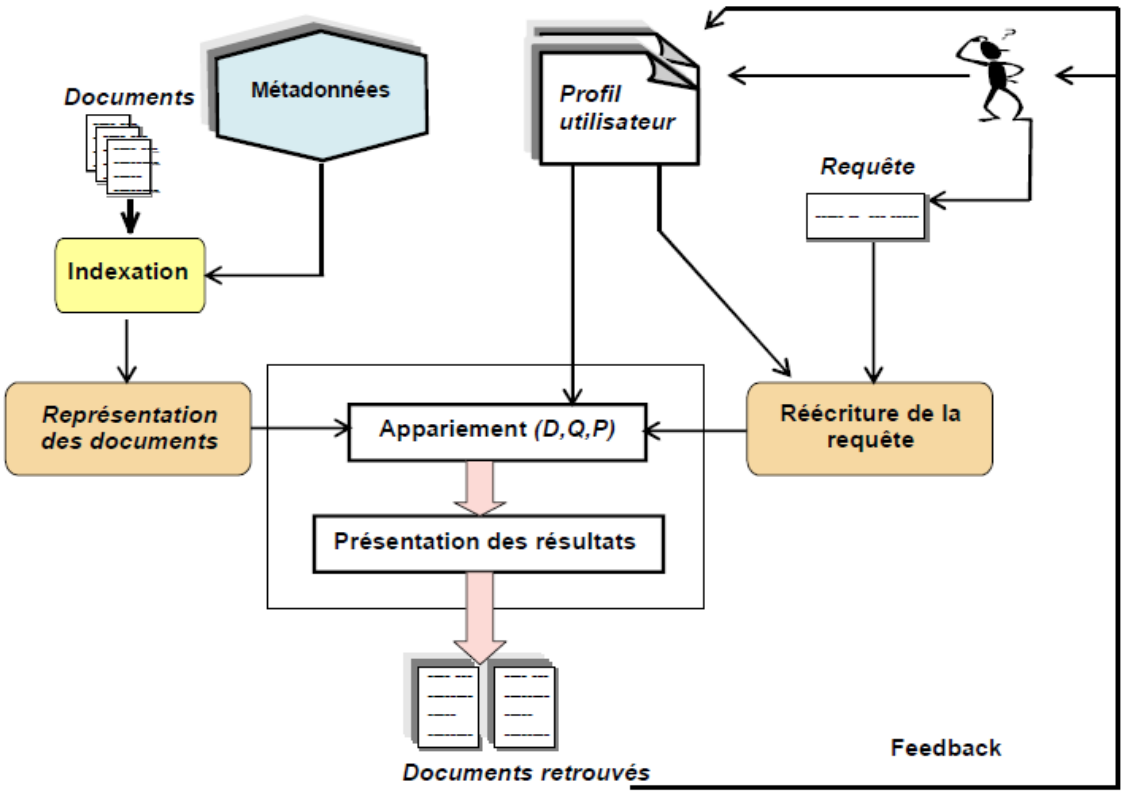

Figure III.1 : Architecture générale d'un SRIP (Zemirli, 2008).

- ✓ Lors de ***l'analyse de la requête***. Le SRIP intègre le profil utilisateur pour mieux cibler le besoin informationnel effectif de l'utilisateur.
- ✓ Lors de ***l'indexation du corpus documentaire***. Le SRIP utilise des métadonnées des documents pour une meilleure représentation de la sémantique des documents.
- ✓ Lors de ***l'appariement requête/document***. Le SRIP inclut également le profil utilisateur pour calculer la pertinence d'un document.
- ✓ Lors de ***l'affichage des résultats***. Le SRIP restitue les documents selon la directive et les préférences incluses dans le profil utilisateur.

III.2.2. Notion de profil utilisateur

La personnalisation de l'information se définit, entre autres, par un ensemble de préférences individuelles représentées par des couples (attribut, valeur), par des ordonnancements de critères ou par des règles sémantiques spécifiques à chaque utilisateur ou communauté d'utilisateurs. Ces préférences servent à décrire le centre d'intérêt de l'utilisateur, le niveau de qualité de données qu'il désire ou les modalités de présentation de ces données. L'ensemble de ces informations est représenté dans un modèle d'utilisateur appelé souvent profil.

Un profil regroupe l'ensemble de connaissances nécessaires à une évaluation efficace des requêtes et à une production d'une information pertinente adaptée à chaque utilisateur. Le contenu du profil utilisateur varie selon les approches et les spécifications. Dans le domaine de l'IHM, le profil contient, par exemple, des informations permettant l'affichage des résultats selon les préférences de l'utilisateur. La notion de profil est souvent liée à celle de préférences et de contexte. En effet, les préférences de l'utilisateur font partie intégrale de son profil. En plus la description de l'utilisateur et ses préférences peut changer en fonction du contexte dans lequel il évolue (Bouzeghoub, 2005).

Un profil utilisateur représente l'ensemble des informations décrivant l'utilisateur. Un contexte représente les données décrivant l'environnement d'interaction entre un utilisateur et un système. Une préférence est une expression permettant de hiérarchiser l'importance des informations dans un profil ou un contexte.

III.2.3. Représentation du profil utilisateur

Comme le contenu du profil dépend fortement de l'application qui l'exploite, on trouve dans la littérature trois principales approches pour représenter et structurer les profils des utilisateurs : représentation vectorielle, hiérarchique et multidimensionnelle.

III.2.3.1. Représentation vectorielle

Ce type de représentation s'appuie généralement sur le modèle vectoriel (Salton, 1971). Le contenu du profil est constitué d'un ou de plusieurs vecteurs définis dans un espace de termes. Ces termes sont obtenus à partir de plusieurs sources d'informations concernant l'utilisateur.

Les coordonnées des vecteurs correspondent aux poids associés aux termes retenus dans le profil. On peut citer comme exemple des systèmes : Alipes, WebMate et Surfagent (Somlo, 2003). L'utilisation de plusieurs vecteurs correspond à deux préoccupations : pouvoir prendre en compte des centres d'intérêt multiples et gérer leur évolution dans le temps.

Cette représentation apporte l'avantage de la simplicité de mise en œuvre. Néanmoins, même si ces systèmes prennent en considération des centres d'intérêts multiples en utilisant plusieurs vecteurs, cette représentation manque de structuration. Cette représentation ne facilite ni l'interprétation ni la prise en compte des différents niveaux de généralités caractérisant l'utilisateur (Bottraud, 2004).

Il reste aussi à résoudre le problème de l'ordonnancement des préférences et des centres d'intérêts de l'utilisateur. En effet, ces derniers sont très variés et n'ont pas le même degré d'importance pour chaque utilisateur. Il faut donc modéliser le profil utilisateur de façon à prendre en considération l'ensemble des paramètres représentant l'utilisateur.

III.2.3.2. Représentation hiérarchique

De nombreuses approches ont été suivies pour améliorer d'avantage les performances des systèmes de recherches d'information personnalisés (SRIP) en structurant mieux les profils utilisateurs. La construction d'une hiérarchie de concepts ou d'une ontologie personnelle, plutôt qu'un ensemble de domaines indépendants (suivant une direction proposée dans un contexte plus général par Huhn (Huhn, 1999)) offre une alternative intéressante à l'approche précédente.

Dans cette approche, la modélisation de l'utilisateur est fondée sur l'élaboration d'une ontologie personnelle. L'ensemble des caractéristiques de l'utilisateur est organisé dans une structure hiérarchique de concepts (catégories) où chaque catégorie représente la connaissance d'un domaine d'intérêt de l'utilisateur.

Le SRIP s'appuie sur la sélection dans une ontologie générale de nœuds estimés correspondre aux intérêts de l'utilisateur. Ainsi, le rapport de généralisation /spécification existant naturellement dans ce genre de structure permet d'avoir une représentation plus réaliste du profil utilisateur.

Le premier à avoir utilisé une telle structure fut Pretschner (Pretschner, 1999) dans le système OBIWAN. Il a proposé un modèle innovant pour la construction du profil utilisateur. Il s'appuie sur l'ontologie publique de Magellan[3] qui est composée d'approximativement 4.400 nœuds de concepts. Semblable à ce travail, on peut citer le système SmartPush (Kurki, 1999) (Gauch, 2003).

Bien que la représentation de ce profil d'utilisateur soit innovatrice, ces travaux ne se servent pas des caractéristiques de la structure hiérarchique (par exemple pour dédoubler ou fusionner des nœuds dans le profil d'utilisateur) pour capturer la dynamique des changements. De plus, la sémantique générale de cette hiérarchie n'est pas formellement indiquée; dans la plupart des cas, ils correspondent à une relation de généralisation/spécialisation.

III.2.3.3. Représentation multidimensionnelle

La représentation multidimensionnelle du profil s'inscrit dans une réflexion globale sur la personnalisation de l'information. En effet, le profil utilisateur est un élément clé dans le processus de recherche, la modélisation de l'utilisateur doit pouvoir capturer toutes les dimensions qui représentent l'utilisateur.

Différents travaux ont abordé cet aspect sans le couvrir dans son ensemble. Ainsi, les propositions de standards P3P (W3C, 2004) pour la sécurisation des profils ont défini des classes distinguant les attributs démographiques des utilisateurs (identité, données personnelles), les attributs professionnels (employeur, adresse, type) et les attributs de comportement (trace de navigation).

Une autre proposition faite par Amato (Amato, 1999) consiste à représenter le contenu du profil utilisateur par un modèle structuré de dimensions (ou catégories) prédéfinis. C'est

[3] http://www. magellan.excite.com

la première approche où les informations sont structurées et qui offre un modèle général. Le modèle de profil contient cinq catégories: données personnelles, données de la source, données de livraison, données de comportement, données de sécurité.

Poursuivant la classification de (Amato, 1999), Kostadinov (Kostadinov, 2007) propose un ensemble de dimensions ouvertes, capables d'accueillir la plupart des informations caractérisant un profil. Il distingue principalement huit dimensions: les données personnelles, le centre d'intérêt, l'ontologie du domaine, la qualité attendue des résultats délivrés, la customisation, la sécurité et la confidentialité, le retour de préférences (feedback), les informations diverses.

III.3. Meta modèles pour un système de personnalisation

Le profil d'un utilisateur ne contient pas uniquement des informations factuelles le décrivant. La notion de profil est souvent liée à celle de préférences et de contexte. En effet, les préférences de l'utilisateur font partie intégrale de son profil. En plus la description de l'utilisateur et ses préférences peuvent changer en fonction du contexte dans lequel il évolue. Cependant il y a une ambiguïté autour des trois concepts : profil, contexte et préférences. Le sens qu'on leur donne change d'une approche à l'autre et il arrive souvent que l'un d'entre eux soit utilisé à la place des deux autres ou des trois à la fois. Cette ambiguïté de la terminologie rend difficile l'étude et la compréhension de la problématique liée à la personnalisation.

Pour décrire l'utilisateur, le contexte et les préférences, Kostadinov propose dans (Kostadinov, 2007) des méta modèles génériques. Le terme « *générique* » ne veut pas dire un ensemble exhaustif d'attributs, mais plutôt une liste de concepts de haut niveau pouvant être spécialisés, affinés et instanciés dans chaque environnement de personnalisation. Il y a toutefois une différence importante entre les méta modèles de profil et de contexte, d'une part, qui caractérisent les individus et les situations dans lesquelles ils interagissent avec un système d'information, et le méta modèle de préférences, d'autre part, qui décrivent une typologie de préférences utilisées dans les deux premiers. Les trois méta modèles doivent satisfaire les exigences suivantes :

- ✓ Ils doivent être capables d'acquérir les principales catégories des connaissances utilisées dans les systèmes de personnalisation actuels,
- ✓ Ils doivent être indépendants de tout système de gestion des données et de toute technologie,

- ✓ La spécialisation, la généralisation et l'instanciation de ces modèles doit être facile,
- ✓ Ils doivent être ouverts, facilement extensibles à d'autres types de connaissances et d'autres types de préférences.

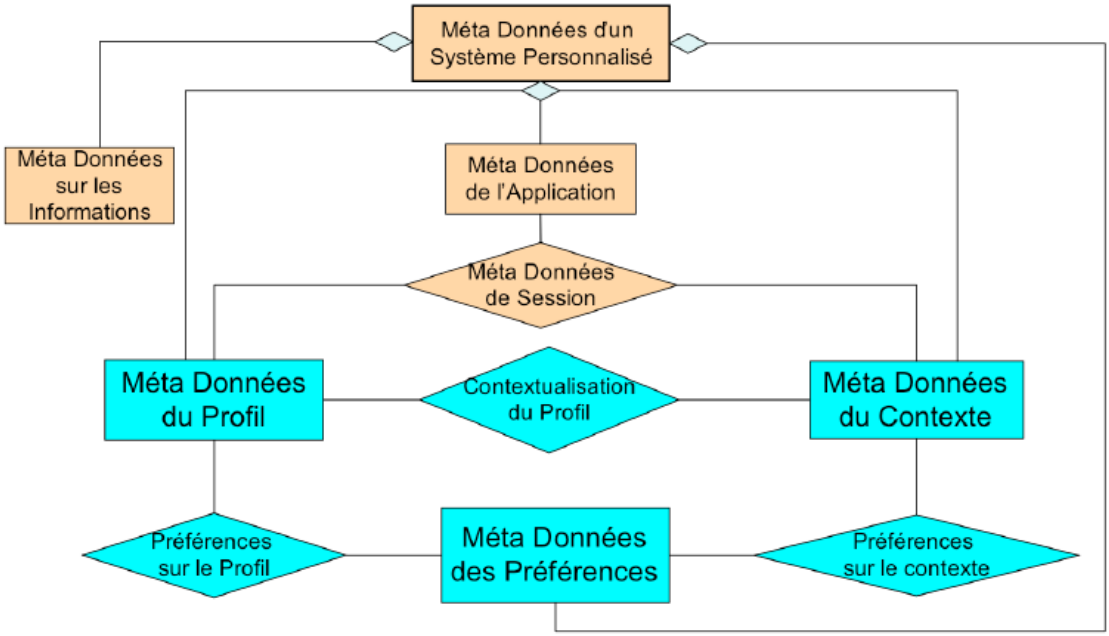

Figure III.2 : Méta modèle global d'un système de personnalisation (Kostadinov, 2007).

III.3.1. Méta modèle de profil

Le profil utilisateur est composé de plusieurs dimensions. Chaque dimension est constituée d'un ensemble d'attributs éventuellement organisés en entités. Les attributs peuvent être simples ou composés. Les attributs composés sont appelés des sous dimensions. Une sous dimension regroupe un ensemble d'attributs simples qui sont liés sémantiquement (par exemple l'adresse est composée du numéro de la rue, du nom de la rue, du code postal etc.). Chaque attribut simple est caractérisé par son nom, le type de ses valeurs et la structure des valeurs. Le type d'un attribut peut être un des types couramment utilisés: entier, réel, chaîne de caractères, etc. La seconde caractéristique (la structure des valeurs) décrit le modèle de représentation des valeurs. Ce modèle de représentation peut être une valeur unique, un ensemble, un intervalle, un vecteur etc. À chaque attribut simple peuvent être associées une ou plusieurs valeurs.

Le profil utilisateur présenté entre le cadre de l'approche ***multidimensionnelle***, est identifié à travers 5 principales dimensions : domaine d'intérêt, données personnelles,

données de qualité, données de livraison et données de sécurité. Les sections suivantes présentent le méta modèle de chacune de ces dimensions.

III.3.1.1. Domaine d'intérêt

Le domaine d'intérêt est la dimension centrale du profil utilisateur. Cette dimension regroupe tous les attributs qui concernent les objets de contenu (informations ciblées). Elle peut définir aussi bien le domaine d'expertise et le niveau de qualification de l'utilisateur dans un domaine particulier que le contenu auquel s'intéresse l'utilisateur.

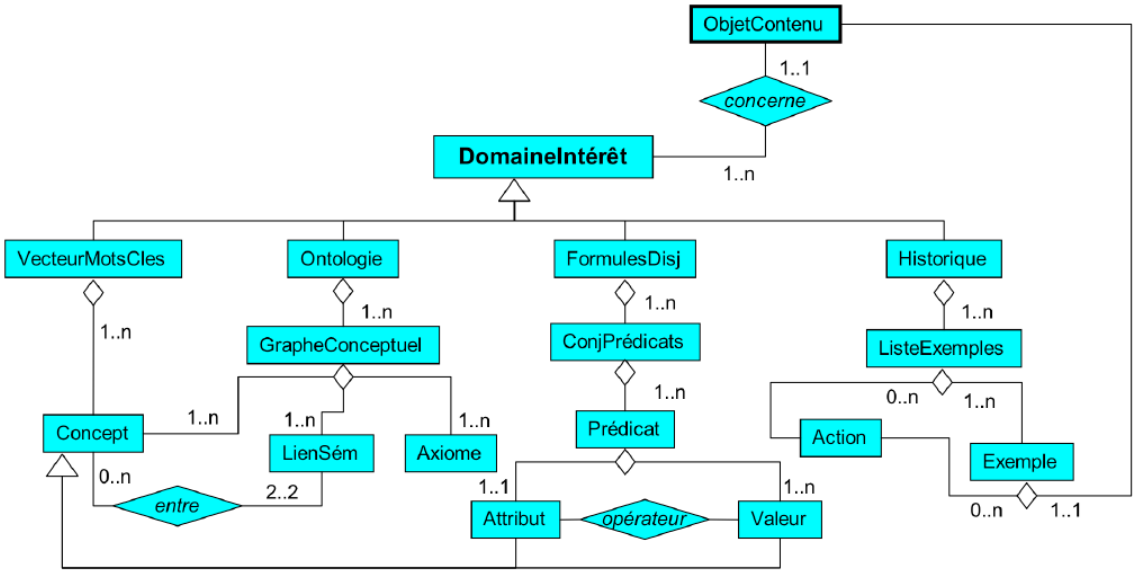

Figure III.3 : Méta modèle de la dimension domaine d'intérêt (Kostadinov, 2007)

Le domaine d'intérêt peut être représenté de différentes manières (voir figure III.3) :

- ***Par un vecteur de mots clés***, comme c'est généralement fait dans le domaine de la RI. Ces mots clés, éventuellement pondérés, traduisent les concepts auxquels l'utilisateur s'intéresse et que le système doit prendre en compte pour délivrer les documents qui les contiennent. Ces mots clés peuvent être organisés en graphes conceptuels ou ontologies explicitant les relations sémantiques entre les concepts.
- ***Par un ensemble de formules logiques***, comme c'est généralement fait dans le domaine des bases de données. Le centre d'intérêt est alors représenté par un ensemble de formules disjonctives. Une formule disjonctive contient un ensemble de conjonctions de prédicats où chaque prédicat est défini sur un attribut d'une entité ou d'une table.

- ***Par un historique des interactions entre l'utilisateur et le système***. Cet historique est représenté par des listes d'exemples annotés par les actions que l'utilisateur a effectuées sur un objet ou sur l'ensemble des objets (ex. lecture, envoie par mail).

III.3.1.2. Données personnelles

Les données personnelles sont la partie statique du profil. Elles contiennent des informations qui décrivent l'utilisateur et ne dépendent pas du système à interroger. Nous distinguons trois catégories de données personnelles : l'identité de l'utilisateur, les données démographiques et les contacts (voir figure III.4). La première catégorie (identité) est composée d'un ensemble d'attributs d'identification de l'utilisateur. Des exemples de tels attributs sont le nom et le prénom de l'utilisateur, son adresse, son numéro de téléphone ou de fax, son adresse email etc. La seconde catégorie de données personnelles contient un ensemble d'attributs démographiques comme par exemple la date de naissance de l'utilisateur, son genre, son revenu, son état civil, le nombre d'enfants etc. Finalement, les contacts de l'utilisateur représentent son carnet d'adresse.

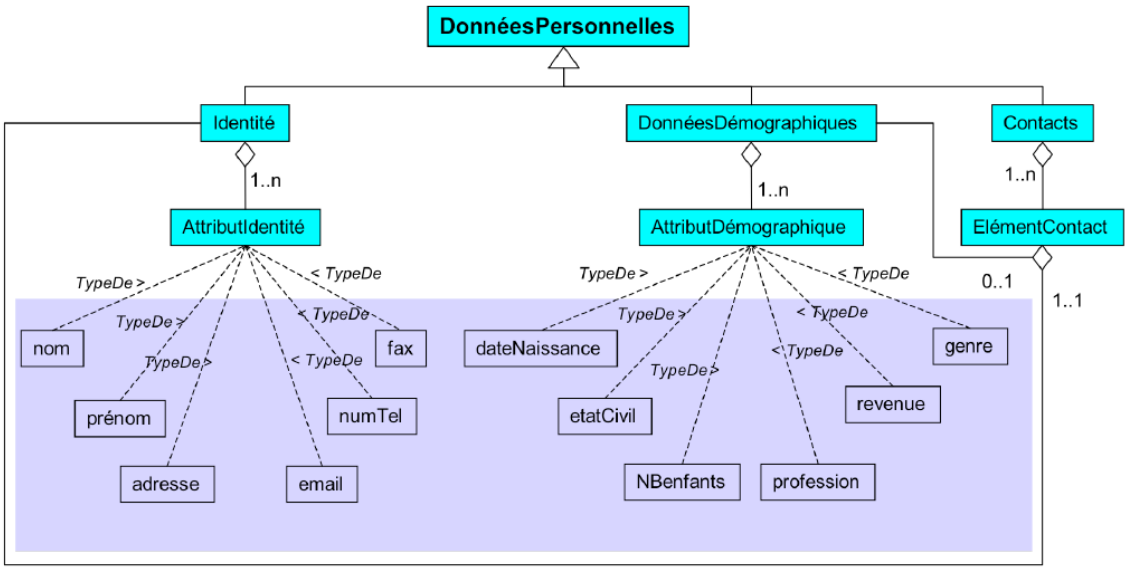

Figure III.4 : Méta modèle de la dimension données personnelles (Kostadinov, 2007)

III.3.1.3. Qualité

La dimension « *Qualité* » joue un rôle très important dans le domaine de la personnalisation. Les données de cette dimension décrivent la qualité attendue ou espérée par l'utilisateur qui sera confrontée à la qualité effective produite par le système de

recherche d'informations afin de restreindre l'espace de recherche. Les informations dans cette dimension sont représentées par des facteurs de qualité (voir figure III.5). Nous distinguons trois catégories de facteurs de qualité selon le type d'objets auxquels ils font référence : (i) facteurs sur le contenu des données, (ii) facteurs sur le contenant des données et (iii) facteurs sur les processus. La première catégorie concerne la qualité désirée des objets de contenu. Des exemples de facteurs de cette catégorie sont la fraîcheur et l'exactitude des données. La seconde catégorie regroupe les facteurs de qualité des conteneurs des données (sources). La plupart des facteurs de cette catégorie résultent de l'expérience des autres utilisateurs (ex. la fiabilité). Ces facteurs sont utilisés pour filtrer les sources de données lorsque leur nombre est trop important. Finalement, la troisième catégorie de facteurs de qualité concernent les processus qui produisent, délivrent ou notifient l'arrivée des résultats. Les facteurs de cette catégorie mesurent aussi bien les performances des processus (ex. temps de réponse) que leur capacité de produire tous les résultats pertinents (rappel) et uniquement les résultats pertinents (précision).

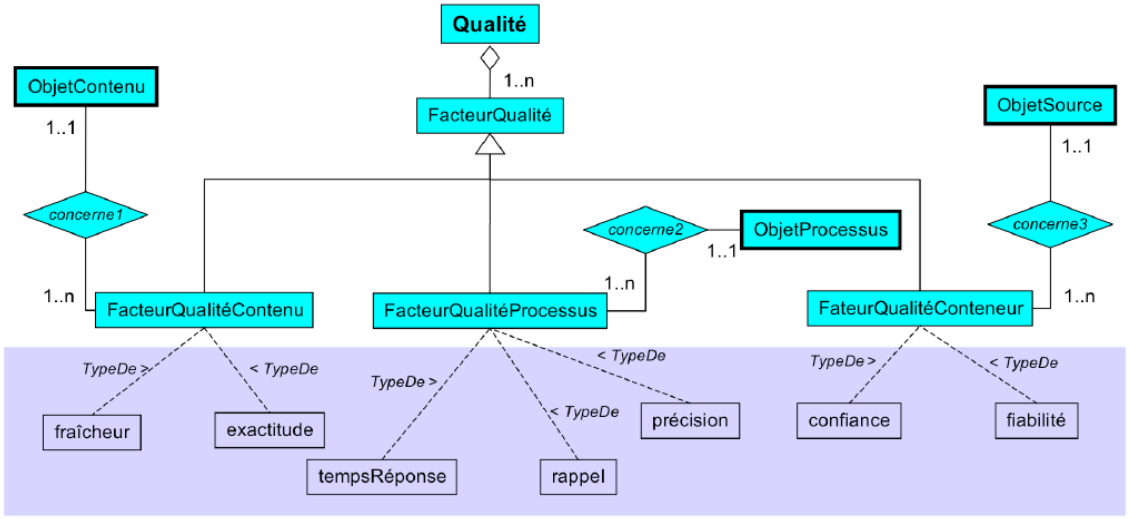

Figure III.5 : Méta modèle de la dimension Qualité (Kostadinov, 2007)

III.3.1.4. Données de livraison

Les données de livraison concernent d'abord tout ce qui est lié aux modalités de présentation des résultats en fonction de la plateforme de l'utilisateur, de la nature et du volume des informations délivrées, des préférences esthétiques ou visuelles de l'utilisateur. À ces modalités de présentation, on peut ajouter les modalités de livraison et de

notification, décrivant le moment et le moyen de livraison des résultats et la manière de notifier cette livraison (différé, immédiat par exemple).

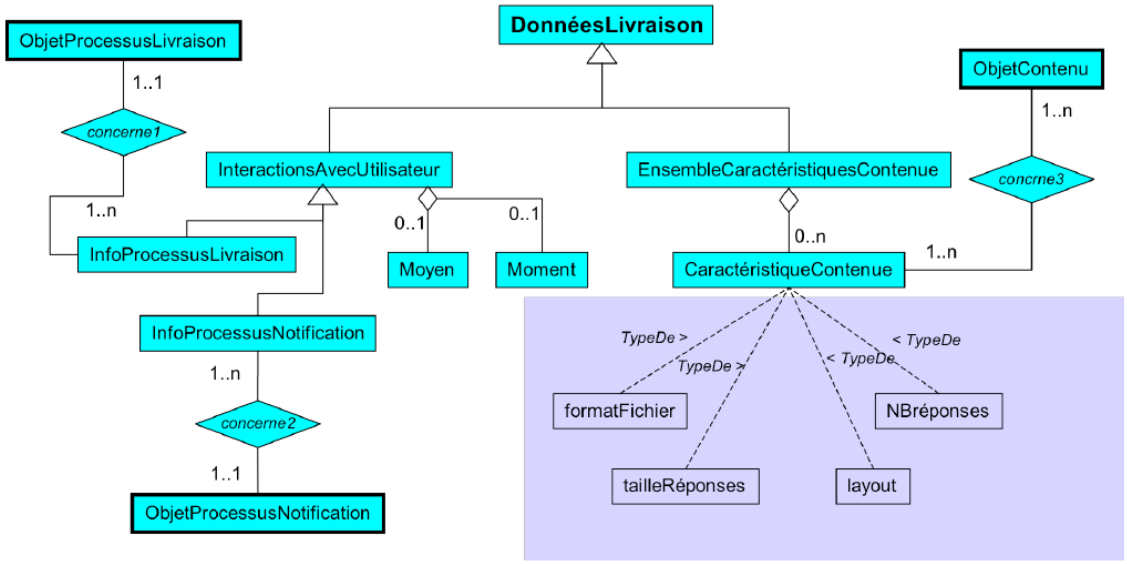

Figure III.6 : Méta modèle de la dimension données de livraison (Kostadinov, 2007)

Les données de livraison sont divisées en deux classes : (i) des caractéristiques des objets de contenu délivrés et (ii) informations sur les interactions entre le système et l'utilisateur (voir figure III.6). La première classe regroupe l'ensemble de caractéristiques qui concernent les objets de contenu. Ces caractéristiques comprennent par exemple le format des documents, leur nombre et leur taille ou encore la mise en page des données. La seconde classe fournit des informations sur le moment et le moyen des interactions entre l'utilisateur et le système. Ces interactions peuvent être la livraison des résultats ou la notification de leur arrivée. Dans le premier cas, les données de la dimension concernent le processus de livraison, alors que dans le second cas, elles caractérisent le processus de notification.

III.3.1.5. Données de sécurité

La sécurité est une dimension fondamentale du profil. Elle peut concerner les données que l'on interroge ou modifie, les informations que l'on calcule, les requêtes utilisateurs ou les autres dimensions du profil.

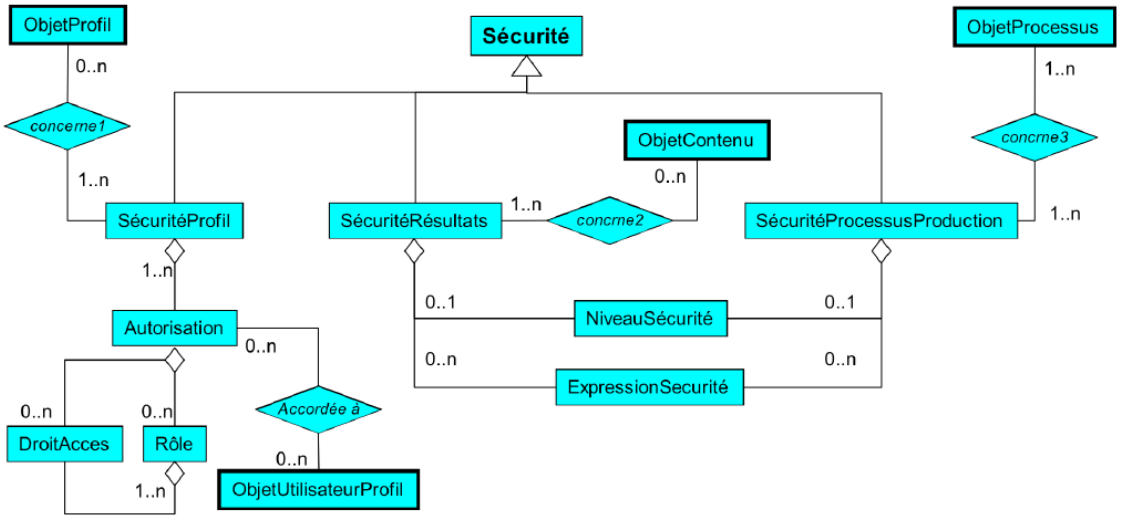

Figure III.7 : Méta modèle de la dimension sécurité (Kostadinov, 2007)

Nous distinguons trois types de sécurité en fonction des objets concernés : (i) la sécurité du profil utilisateur, (ii) la sécurité des résultats et (iii) la sécurité du processus de production des résultats (voir figure III.7).

Le premier type de sécurité (la sécurité des éléments du profil utilisateur) se fait à travers des autorisations d'accès aux informations du profil. Les utilisateurs d'un profil, qui peuvent être le système d'information ou les autres utilisateurs particuliers, se voient attribuer des droits d'accès ou des paquets de droits d'accès (rôle) afin de pouvoir interroger les données de ce profil.

La sécurité du second type (sécurité des résultats) concerne les objets de contenu qui sont délivrés à l'utilisateur. Elle est exprimée en indiquant le niveau de sécurisation des résultats (sur une échelle prédéfinie) ou à travers des expressions dans un formalisme existant.

La sécurité du troisième type (sécurité du processus) s'exprime de la même manière que celle de la sécurité des résultats. La sécurité de ce dernier type exprime la volonté de l'utilisateur de cacher un traitement sensible qu'il effectue. Par exemple, l'utilisateur d'un système d'information proposant des informations sur la bourse peut vouloir cacher le fait qu'il fait des requêtes pour connaître le prix des actions des compagnies pétrolières.

III.3.2. Méta modèle de contexte

Le contexte regroupe toutes les informations relatives à l'environnement dans lequel se fait l'interaction entre l'utilisateur et le système d'information. Le contexte est composé

d'un ensemble de dimensions où chaque dimension contient des attributs simples ou composés (sous dimensions). Les principales dimensions du contexte sont : la dimension spatiale, la dimension temporelle et l'équipement utilisé par l'interaction (voir figure III.8). Rappelons que le méta modèle de contexte, comme celui du profil, est ouvert et d'autres dimensions peuvent y être ajoutées si les besoins de l'application l'exigent.

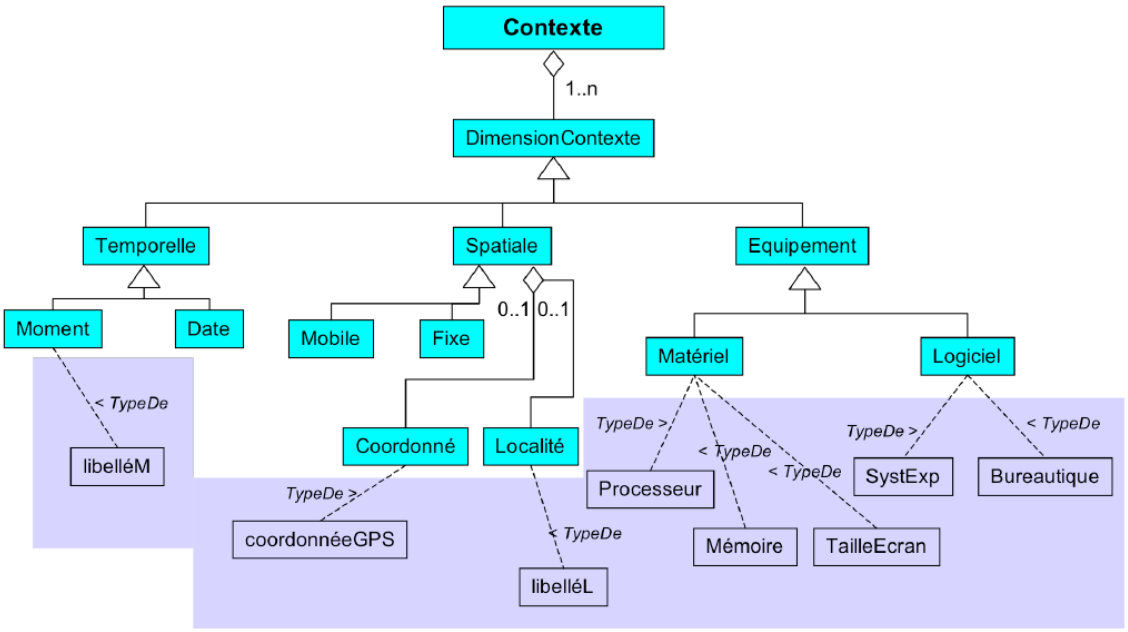

Figure III.8 : Méta modèle du contexte (Kostadinov, 2007)

Le contexte permet de situer l'utilisateur dans le temps et dans l'espace. Ceci est fait à travers les dimensions temporelle et spatiale. La dimension temporelle indique le moment de l'interaction. Ce moment peut être représenté dans un format de date standard ou en utilisant un libellé qui indique la période du temps (matin, soir, ...).

La dimension spatiale désigne l'endroit dans lequel se trouve l'utilisateur. Cet endroit peut être statique (à la maison) ou dynamique (en voiture). La dimension spatiale peut contenir les coordonnées précises de l'utilisateur ou le libellé général de la localité dans laquelle il se trouve (travail, maison, voiture, ...).

En plus de l'aspect spatio-temporel, le contexte contient des informations sur l'équipement de l'utilisateur. Cet équipement peut être représenté par des caractéristiques du matériel et/ou du logiciel dont se sert l'utilisateur. Des exemples de caractéristiques du matériel sont la taille de l'écran, la capacité de la mémoire, la rapidité du processeur etc.

Il est intéressant à remarquer que le contenu du contexte est très proche de celui de la dimension « *Données de livraison* » du profil utilisateur. Même s'il existe une forte

ressemblance entre les deux, le contexte et les données de livraison ne sont pas redondants, mais complémentaires. Le contexte détermine les valeurs du profil utilisateur qui peuvent être prises en compte, mais si plusieurs options sont possibles, le choix de celle qui sera prise est fait en fonction des préférences qui se trouvent dans le profil.

III.3.3. Méta modèle de préférences

Le méta modèle de préférence décrit les types de préférence qu'on peut trouver dans les profils et dans les contextes. Ce modèle n'est pas instanciable séparément des deux autres. Il fournit une palette de préférences qui peuvent être utilisées pour hiérarchiser l'importance des connaissances définies dans les profils et les contextes.

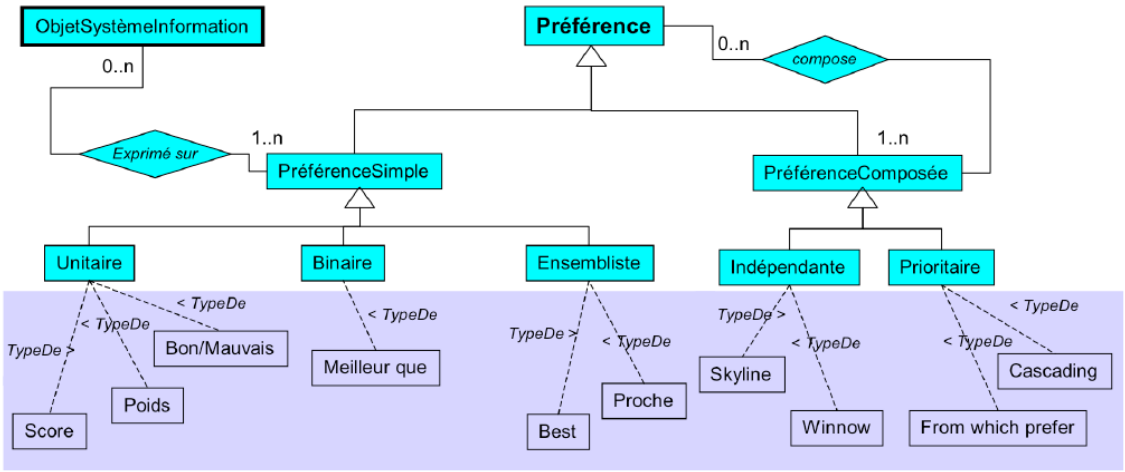

Figure III.9 : Méta modèle de préférences (Kostadinov, 2007)

Il y a deux types de préférences : des préférences simples et des préférences composées (voir figure III.9). Une préférence simple est définie par un seul concept caractérisant un ou plusieurs objets. Ce concept peut être un poids, un score, une fonction d'utilité, un prédicat ou tout mot clé introduisant une appréciation sur un objet ou un ordre d'intérêt entre deux ou plusieurs objets. Selon le nombre d'objets caractérisés, les préférences simples peuvent être unitaires, binaires ou ensemblistes. Les préférences unitaires s'appliquent sur un seul objet. Souvent elles sont représentées par une annotation de l'objet qui peut être un score, un poids ou une note de pertinence. Les préférences binaires, comme leur nom l'indique, montrent un favoritisme entre deux éléments. Elles sont souvent exprimées par des fonctions binaires de préférence permettant d'établir une relation d'ordre entre les éléments. Finalement, les préférences ensemblistes s'appliquent à un ensemble d'objets. Elles permettent de distinguer le sous ensemble pertinent

d'éléments. Des exemples de telles préférences sont la recherche des meilleurs objets (Best) ou de ceux qui s'approchent le plus d'un ensemble d'exemples pertinents (Proche).

Une préférence composée est une expression de deux ou plusieurs préférences intermédiaires. Une préférence intermédiaire peut être une préférence simple ou une préférence composée. La combinaison de préférences peut se faire de façon indépendante ou prioritaire.

III.3.4. Relations entre le profil, le contexte et les préférences

Dans les sections précédentes, la description de l'utilisateur, la représentation du contexte et les expressions des préférences ont été séparées afin de mieux les étudier. Cependant, une représentation complète de l'utilisateur et du contexte, contient non seulement des éléments de description, mais également des préférences. En plus, la définition d'un profil utilisateur peut dépendre du contexte dans lequel il est exploité. Par conséquent, l'obtention d'un modèle complet décrivant l'utilisateur et/ou le contexte passe obligatoirement par l'identification des relations qui existent entre les trois composants profil, contexte et préférences.

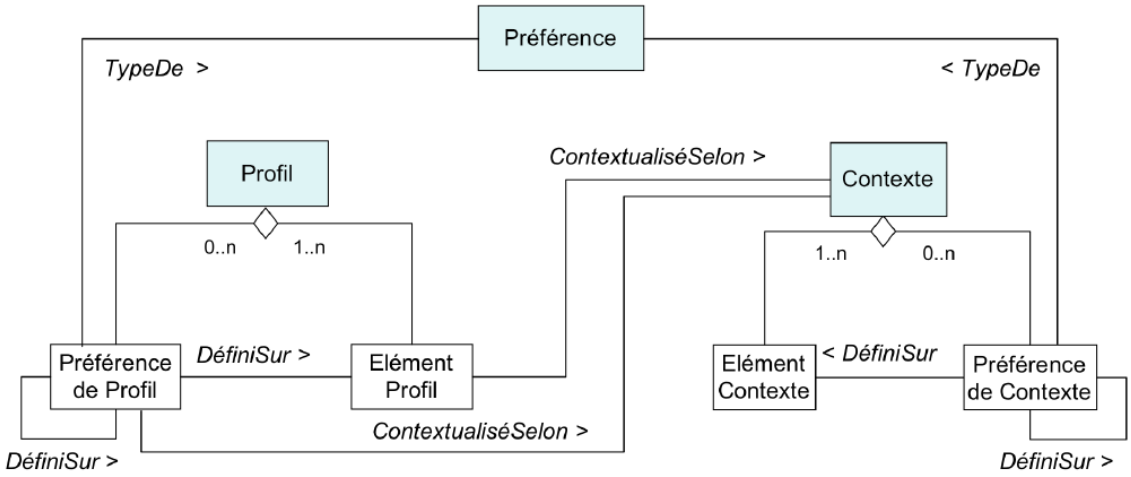

Figure III.10 : Associations entre profil, contexte et préférences (Kostadinov, 2007)

La figure III.10 illustre les principaux liens qui existent entre le profil, le contexte et les préférences. En plus de l'agrégation, trois catégories d'associations sont utilisées pour construire le modèle de cette figure : « type de », « défini sur » et « contextualisé selon ». Les liens de la première catégorie permettent de différencier les préférences du profil de celles du contexte. La seconde catégorie de liens sert à relier les préférences aux éléments

sur lesquels elles sont exprimées. Finalement, les associations « contextualisé selon » assurent la relation entre le contexte et le contenu du profil.

Un profil utilisateur est composé d'un ensemble d'éléments de profil et de préférences. Les éléments du profil peuvent être des dimensions, des sous dimensions, des attributs et des valeurs. Des préférences peuvent être définies sur l'ensemble de ces éléments à condition de combiner des éléments du même type. Par exemple il est possible d'exprimer une préférence entre deux valeurs du même attribut, mais il est difficile d'envisager une préférence entre une dimension du profil et la valeur d'un attribut.

Le contexte est défini de la même manière que le profil. Il est composé d'un ensemble d'éléments de contexte et de préférences. L'utilisation de préférences au niveau du contexte paraît moins naturelle que celle au niveau du profil. Les préférences du contexte permettent de définir un choix par défaut sur les caractéristiques du contexte. Par exemple, il est possible d'exprimer le fait que l'ordinateur à partir duquel l'utilisateur se connecte est plus important que son emplacement physique pour décider s'il est dans un contexte de travail ou de loisir.

Finalement, le contenu du profil peut dépendre du contexte. Cette dépendance est exprimée par les associations « contextualisé selon ». Un élément du profil ou une préférence peut être contextualisé ou non. Lorsqu'un composant (préférence ou élément du profil) est non contextualisé, il est utilisé systématiquement dans tous les contextes. Un exemple de tel composant est le nom de l'utilisateur qui ne change jamais.

III.4. Gestion de profils

La gestion de profils implique la mise en place d'un ensemble d'outils qui facilitent la définition et l'évolution des profils. Les principales opérations de gestion de profils sont (Kostadinov, 2007) :

- ***Instanciation :*** cette opération permet de créer des profils particuliers.
- ***Importation :*** cette opération permet de rendre un profil utilisable sur un système qui n'est pas celui qui a servi à le créer. Elle implique un test de validité du profil par rapport au modèle de profil reconnu par le système (validité sémantique).
- ***Validation :*** la validation d'un profil permet de déterminer quelle partie de ce profil peut être satisfaite sur un système particulier. Elle implique un test par rapport à l'environnement technique dans lequel le profil sera utilisé (validité opérationnelle).

- ***Intégration :*** l'intégration permet de combiner le contenu de deux ou plusieurs profils dans un seul. Cette opération peut être appliquée sur les différents profils du même utilisateur ou sur les profils de différents utilisateurs. Dans le premier cas, l'objectif est d'uniformiser la représentation de l'utilisateur, alors que dans le second cas il s'agit de créer le profil d'une communauté d'utilisateurs.
- ***Adaptation :*** l'adaptation d'un profil se fait toujours par rapport à un contexte. Elle permet d'obtenir le contenu du profil par rapport au contexte dans lequel se trouve l'utilisateur.
- ***Appariement :*** l'appariement de profils comprend un ensemble d'opérations de base qui sont le matching, la différence et l'équivalence de profils. Ce sont des opérations qui peuvent être utilisées comme briques de base pour obtenir les autres opérations.

III.5. Le profil utilisateur dans le Web sémantique

Le Web d'aujourd'hui ce concentre de plus en plus sur l'utilisateur et plusieurs projets et standards viennent prendre part au Web 2.0. des ontologies RDF (comme FOAF, vCard, etc.), des standards XML (comme APML, etc.), des extensions de HTML (les microformats, etc.). Tous ces standards sont là pour avoir un Web plus ouvert et plus structuré, pour pouvoir partager le plus grand nombre d'information sur l'utilisateur, dans l'objectif d'améliorer la qualité des informations transmises à celui-ci.

III.5.1. Formats

- **VCard**

VCard[4] (VisitCard) est un format ouvert pour l'échange des données personnelles, il est utilisé par les logiciels de carnet d'adresses (Outlook, Thunderbird, …) ainsi que par les appareils mobiles et les logiciels de messagerie instantanée (Jabber). Il est possible de décrire les données vCard dans un format simple utilisé par les outils cités ci-dessus, ou bien au format RDF ou XML [http://www.w3.org/TR/vcard-rdf] pour utilisation dans un système plus global. Avec vCard, il est possible de représenter des informations personnelles (nom, prénom, surnom, date de naissance, titre, rôle, organisation, adresse, e-mail, téléphone, …), des données binaires (photo, son, logo, …), ainsi que des liens vers d'autres entrées vCard via la propriété (agent).

[4] ftp://ftp.rfc-editor.org/in-notes/rfc2426.txt

```
<rdf:RDF xmlns:rdf=".../22-rdf-syntax-ns#" xmlns:vCard=".../vcard-rdf/3.0#">
<rdf:Description rdf:about = "http://qqqfoo.com/staff/corky" >
<vCard:FN>Corky Crystal</vCard:FN>
<vCard:N rdf:parseType="Resource">
<vCard:Family>Crystal</vCard:Family>
<vCard:Prefix>Dr</vCard:Prefix>
</vCard:N>
<vCard:BDAY>1980-01-01</vCard:BDAY>
<vCard:TITLE>Computer Officer Class 3</vCard:TITLE>
<vCard:ROLE>Programmer</vCard:ROLE>
<vCard:TEL rdf:parseType="Resource">
<rdf:value> +61 7 555 5555 </rdf:value>
<rdf:type rdf:resource="http://www.w3.org/2001/vcard-rdf/3.0#work"/>
<rdf:type rdf:resource="http://www.w3.org/2001/vcard-rdf/30#voice"/>
</vCard:TEL>
</rdf:Description>
</rdf:RDF>
```

Figure III.11 : Exemple de description vCard

- **FOAF**

FOAF[5] (Friend Of A Friend) est une ontologie permettant de décrire les personnes, les liens entre eux, et les choses qu'ils créent et font. Elle est composée de plusieurs modules, des attributs basics (nom, prénom, e-mail, ...), des informations personnelles (connaissances, blogs, intérêts, ...), comptes (MSN, ICQ, Yahoo, ...), Appartenance à des groups (groups, organisations, projets), documents (documents, images, ...) (Mika, 2005).

Une des extensions de FOAF est FOAF/Trust qui est un module permettant de définir la confiance d'une relation (Goldbeck, 2005) (Dolog, 2003). La confiance dans une relation permet de définir les utilisateurs pouvant influencer dans le cadre d'un système de recommandation. Une personne peut donc être identifiée par son profil FOAF, celui-ci étant unique sur le Web. On peut donc apercevoir la possibilité d'avoir une authentification unique avec OpenID et un profil unique avec FOAF dans un futur proche.

- **SIOC**

L'ontologie SIOC[6] (Semantically Interlinked Online Communities) a pour objectif de rationaliser les communautés en ligne, Utilisateurs, Groupes, Forums, Posts, Mailing-Lists

[5] http://www.foaf-project.org/index.html
[6] http://sioc-project.org/

et thèmes associé. Cette ontologie s'intègre donc naturellement avec FOAF, ou une entité foaf:Person peut avoir plusieurs comptes sioc:User. SKOS peut aussi être utilisée pour définir les thèmes des forums.

- **APML**

APML[7] (Attention Profiling Markup Language) est un standard émergeant permettant de modéliser les préférences d'un utilisateur. L'idée est de regrouper toute forme de données concernant les préférences (attention portée à des thèmes) dans un format portable contenant ces préférences et leurs poids (APML, 2007).

Le processus global de capture de préférences est composé des étapes suivantes :

1. Analyse du comportement de l'utilisateur sur un ensemble de ressources, nommées « Attention Data »
2. Inférence des intérêts de l'utilisateur « Interest Cloud »
3. Structuration du profil sous forme XML « APML profile »

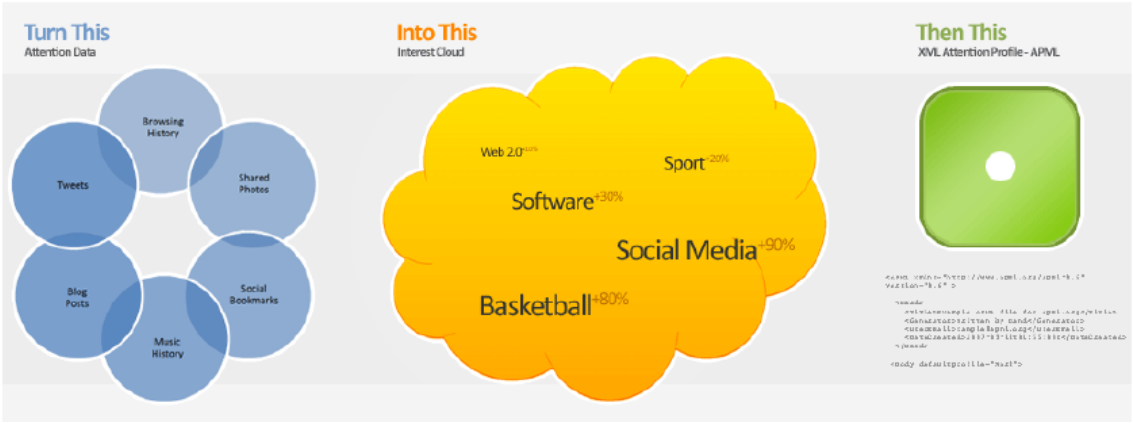

Figure III.12 : Profiling des intérêts de l'utilisateur - Attention Profiling[8]

Les avantages du standard APML résident dans les points suivants :

- ✓ Il permet/facilite le filtrage de l'information. En pondérant l'attention portée à des thèmes, des auteurs, il est possible de donner des priorités ou de filtrer les données reçues par un utilisateur.
- ✓ Imputabilité des données. Certaines informations inférées automatiquement par le système peuvent être erronées, l'utilisateur peut valider l'exactitude de ces données.
- ✓ Respect de la vie privée, par la définition du : Qui, Quand, Quoi. Qui peut voir quelles informations, à quel moment ?

[7] http://www.apml.org/
[8] http://www.faradaymedia.com/

- ✓ Données partagées. Le partage de données et l'agrégation de plusieurs profils APML peut donner lieu à des scenarii du genre : J'écoute de la musique sur last.fm. Mon profil APML est transféré vers Amazon. Lorsque je me connecte sur Amazon, le système me propose d'acheter la musique que j'aime.
- ✓ Control du profil. APML donne la possibilité à l'utilisateur de contrôler son profil, pour qu'il soit le plus proche de la réalité et pour contrôler l'image qu'ont les autres personnes de vous.

- **Microformats**

Les 'microformats'[9] (hCard, hCalendar, hReview, rel-tag, XFN, …) sont des normes d'écriture de fichiers XHtml, pour pouvoir en tirer une signification, en utilisant des valeurs spéciales dans les attributs (class et rel) de XHtml.

La vision des microformats vient se positionner entre le Web classique et le Web sémantique. En fait le Web classique se focalise sur l'utilisateur comme consommateur de l'information, et le Web sémantique sur la machine et l'adoption du Web sémantique fût un peu difficile, les microformats viennent améliorer le Web classique sans le changer carrément. En introduisant des conventions sur l'utilisation de propriétés XHtml telles que (class et rel). Donc les microformats sont conçus pour l'homme d'abord, pour ne pas chambouler tout ce qui a été construit pour le Web 1.0, puis pour les machines qui peuvent traiter ces données.

III.5.2. Portabilité des données (data portability)

L'initiative Data Portabilité[10], DP, et les standards ouverts permettent de partager les données d'une manière plus libre entre différents sites pour pouvoir réutiliser les informations personnelles, les listes de contacts, les préférences, …

Plusieurs standards composent aujourd'hui l'initiative : OpenID, OAuth, RSS, APML, Yadis, OPML, hCard, XFN, etc. Une telle initiative permet à l'utilisateur de reprendre contrôle sur ces propres données, lui permettant de réutiliser ces données et d'éviter de créer un nouveau compte pour chaque site, retenir les informations d'authentification, créer son profil, créer sa liste de contacts, …

[9] http://microformats.org/
[10] http://www.dataportability.org/

III.6. Requête utilisateur

D'après l'encyclopédie wikipedia, en informatique, une requête (en anglais request) est une demande de traitement. Le terme est notamment employé dans le contexte:

- ✓ des bases de données, une requête correspond à l'interrogation d'une base pour en récupérer une certaine partie des données ;
- ✓ des protocoles client-serveurs, une requête est le message initial envoyé par le client vers le serveur. Le message du serveur étant la réponse

Dans notre contexte la requête représente une demande pour trouver un service Web satisfaisant l'utilisateur. Dans cette section nous présentons la classification des requêtes proposée par Yann dans sa thèse (Yann, 2006). Cette classification propose les catégories de requêtes suivantes :

III.6.1. Requêtes de recherche de traitements

Un utilisateur cherche un ou plusieurs traitements, en vue de satisfaire un besoin précis et bien défini, ou au contraire vague. Voici des exemples de requêtes.

- ✓ Ou sont disponibles les programmes d'appariement ?
- ✓ Quels sont les algorithmes de calcul de flux sur un réseau ?
- ✓ Quels sont les traitements utilisables sur les objets "bâtiments" ?

Certaines requêtes sont plus complexes que d'autres. Plutôt qu'une simple sélection dans la base de métadonnées, ces requêtes requièrent la mise en œuvre des raisonnements.

III.6.2. Requêtes pour connaître les traitements

L'analyse des requêtes typiques exprimées par les utilisateurs montre que pour un traitement donné les besoins d'information portent sur cinq thèmes principaux : les métadonnées qui l'identifient (nom, date, auteur, etc.), "ce qu'il fait", "comment il fonctionne", "comment l'utiliser" et "quelle évaluation en est faite".

Voici un exemple de requêtes pour chacun des cinq thèmes en question.

- ✓ Quels sont les requêtes topologiques permises par Arcview 8 ?
- ✓ Sur quelle théorie mathématique repose l'algorithme Accordéon ?
- ✓ Le programme de détramage « planche mère » est-il rapide ?

III.6.3. Requêtes de l'utilisation des traitements

Une des questions les plus couramment posées est probablement "comment utiliser ce traitement ?". C'est également une des questions auxquelles il est le plus difficile de répondre. D'abord parce qu'elle mobilise de nombreuses connaissances, souvent tacites et non liées directement au traitement, ensuite parce que la réponse dépend du contexte d'utilisation.

III.7. Travaux relatifs

Une tendance actuelle consiste à exploiter l'aspect de la personnalisation dans la découverte des services Web. Dans ce cadre plusieurs travaux sont effectués, entre autres nous citons :

- L'approche proposée par Sacramento dans (Sacramento, 2009), qui utilise les annotations sémantiques pour la découverte automatique et la composition des services Web géographiques. La thèse traite d'abord de la façon de combiner la sémantique fournies par les catalogues de métadonnées avec la sémantique fourni par des ontologies pour produire des services catalogue capable de faciliter la recherche des services appropriés du Web. Puis, il aborde la façon de composer des services Web à partir des services Web de base OGC de manière automatique.
- Dans (Dou, 2012), une méthode multicritères de décision, nommé AHP (Analytic Hierarchy Process), est introduit pour transformer les préférences personnelles qualitatives et les priorités des utilisateurs en poids numériques. En outre, une méthode d'évaluation de QoS est présentée pour une sélection des services partagée. Enfin, une étude de cas est présentée pour démontrer la faisabilité de la méthode.
- Dans (Elfirdoussi, 2014), les auteurs proposent une approche flexible qui utilise un algorithme efficace basé sur le Score de Popularité des Services Web (WSPS). Ce score est un classement des services Web simple, composite et sémantique, avec l'utilisation des indicateurs tels que l'utilisabilité, l'âge, la pertinence, etc. De plus, les auteurs optent pour affecter un poids à chaque métrique comme un filtre pour sélectionner le service le plus adéquate, puisque une métrique peut avoir une grande importance par rapport à autre métrique selon les préférences de l'utilisateur.
- Dans (Tang, 2014), les auteurs présentent un moteur de recherche de confiance par l'intégration des fonctionnalités de service, QoS (qualité de service) et la confiance de service. Le moteur de recherche proposé contient principalement quatre composantes:

matching des services à base mots clés, évaluation de la qualité des services, évaluation de la réputation de service et une méthode de classement hybride qui combine les résultats obtenus par les trois composantes précédentes pour produire les services finaux adéquates. Pour évaluer la performance du moteur de recherche des services, des expériences complètes sont effectuées à l'aide d'un service Web réel « jeu de données ».

- Dans (ElBouhissi, 2014), les auteurs abordent la question de la découverte de service Web donnée par une description sémantique de service non explicite, qui correspond à une demande de service spécifique. Leur approche est basée sur un objectif de l'utilisateur capturée à partir d'un formulaire HTML et la traçabilité. Elle donne comme résultat une catégorisation sémantique de service, découvertes sémantique et sélection du meilleur service Web. En outre, la proposition des auteurs utilise des algorithmes de matching par ontologie pour correspondre une requête spécifique à un service Web existant.
- Dans (Guidara, 2015), les auteurs proposent une approche de sélection de service basé sur QoS et des techniques d'élagage temporelles afin de réduire le nombre des candidats des services. L'approche proposée permet l'élagage des services basé sur un ensemble de seuils locaux. Ces derniers sont mesurés par des modèles d'optimisation qui traitent des structures générales de flux, y compris : séquentielle, parallèle, le choix, les boucles et de différents types de QoS et de contraintes temporelles.

III.8. Conclusion

Pour compléter notre étude concernant les connaissances sémantiques nécessaires pour la satisfaction de l'utilisateur, nous avons présenté dans ce chapitre un survol sur le concept profil utilisateur. Ce dernier est présenté selon Kostadinov dans sa thèse (Kostadinov, 2007) selon 05 dimensions : domaine d'intérêt, données personnelles, données de qualité, données de livraison et données de sécurité. Nous avons présente aussi les méta modèles de profil, de contexte et de préférences ainsi que les relations qui existent entre eux.

Dans l'objectif d'optimiser la découverte des services Web en réduisant l'espace de recherche et augmentant le nombre de services pertinents. Nous avons opté à proposer comme première contribution un Framework à base de profil utilisateur et métadonnées pour la découverte sémantique des services Web en utilisant la technologie d'agents mobiles. Ce dernier « les agents mobile » fait l'objet du prochain chapitre.

Chapitre IV

Les Agents Mobiles et Leurs Spécificités

IV.1. Introduction

Les systèmes multi-agents sont devenus un paradigme dominant dans le domaine de développement des systèmes distribués complexe. Leurs avantages consistent notamment en leur capacité d'aborder les problèmes complexes d'une manière distribuée et de proposer, en outre, des solutions réactives robustes. Les concepts qui y sont liés font les systèmes multi-agents une approche prometteuse pour la conception des systèmes complexes.

D'autre part le paradigme agent mobile est devenu aussi un formalisme très puissant pour le développement des applications réparties. Un agent mobile (Arcangeli, 2002) est une entité logicielle qui se déplace d'un site à un autre en cours d'exécution pour accéder à des données ou à des ressources distantes. Il se déplace avec son code son état d'exécution et ses données propres. Le but du déplacement est d'accéder localement à des données ou à des ressources initialement distantes, d'effectuer le traitement en local et de ne déplacer que les données utiles.

L'objectif de ce chapitre est d'initier le lecteur novice au domaine de système multi-agent qui constitue un des piliers de notre travail. Nous nous intéressons tout d'abord aux entités qui composent cette catégorie de système : les agents. Pour une telle raison, nous allons présenter, en premier lieu, des considérations générales sur les notions d'agent et de systèmes multi-agents. Par la suite, nous procédons par un état de l'art sur les agents mobiles: définitions, caractéristiques, motivation en démontrant les avantages de ce paradigme, une synthèse sur les différentes plates-formes d'agents mobiles existantes de nos jours, etc. Enfin nous terminons ce chapitre par les principaux travaux de découverte des services Web en utilisant les systèmes multi-agents et la technologie d'agent mobile.

IV.2. Concept d'agent

Dans cette section nous présentons les définitions et les différentes méthodes de classification des agents.

IV.2.1. Définition

Le concept d'agent a été essentiellement inspiré des domaines de philosophie et de la psychologie. Plusieurs définitions ont été données à travers les années, où chaque définition traite un ou plusieurs aspects de ce paradigme :

Définition 1 : *« un agent est un système capable d'action autonome et réfléchie dans un environnement réel. »* (Brustoloni, 1991)

Définition 2 : *«un agent est un logiciel persistant qui a un but bien précis, l'agent peut être distingué d'un logiciel classique par sa taille car plus petite et par ces objectifs et agendas sur les quelles il se base pour accomplir ses taches. »* (Smith, 1994)

Définition 3 : *«un agent est un système informatique qui se trouve dans un environnement complexe et dynamique, et qui aperçoit puis réagit de façon autonome, afin de réaliser les buts pour lesquels il a été créé.»* (Maes, 1994)

Définition 4 : *«un agent intelligent est un logiciel qui effectue un ensemble de tâches prédéterminées, avec un certain degré d'indépendance et d'autonomie, en se basant sur un ensemble de connaissances et une représentation d'objectifs prédéterminés.»*[11]

Un agent peut être également, distingué d'autres entités logicielles par son (Mlungisi, 2008):

- ***Autonomie:*** un agent peut agir sans revenir à son possesseur afin de prendre des décisions.
- ***Comportement social:*** un agent a la faculté de communiquer et d'interagir avec les autres éléments de son environnement.
- ***Réactivité:*** un agent peut apercevoir et ensuite réagir aux différents évènements sur son environnement.
- ***Pro-activité:*** un agent a la faculté de prendre des initiatives afin de s'adapter au changement de son environnement.

[11] http://www.networking.ibm.com/wbi/wbisoft.htm

- ***Persistance:*** un agent doit suivre son but sans interruption jusqu'à l'achèvement de ce dernier.
- ***Raisonnement et rationalité:*** un agent a la faculté de raisonner rationnellement afin de choisir les meilleures actions à entreprendre pour optimiser sa productivité.
- ***Mobilité:*** un cas particulier d'agent logiciel est l'agent mobile qui a la faculté de se déplacer d'une machine à une autre, afin d'exécuter des tâches de natures distribuées.

IV.2.2. Typologie des agents

Les agents peuvent être classés selon différentes propriétés (Zgaya, 2007), par exemple: la granularité (degré d'intelligence), le rôle, la mobilité, la capacité de coopérer, etc…

IV.2.2.1. Classification des agents selon la granularité

- **Les agents réactifs :** ce sont des agents passifs. Ils ont un comportement du type "stimulus/réponse", le stimulus étant un élément de l'environnement (action, message) Ce type d'agent ne dispose pas de module de raisonnement interne. Généralement un système réactif comprend un grand nombre d'agents de faible granularité.
- **Les agents proactifs :** ce sont des agents dynamiques qui entreprennent car ils possèdent, en plus de leurs attributs et méthodes, des processus internes qui leur permettent de prendre des initiatives pour réaliser leurs buts. Un agent proactif est donc un agent dirigé buts.
- **Les agents cognitifs :** un agent cognitif raisonne avant d'agir, il est souvent associé au trio bouclant : perception-raisonnement-action. Il possède, en plus de ses buts, des notions psychologiques qui peuvent être exprimés par le biais des attitudes mentales comme les croyances, les intentions et les désirs. Le courant cognitif prédispose les agents à être dotés d'un outil d'abstraction pour les décrire comme des systèmes intentionnels.

IV.2.2.2. Classification des agents selon la fonction ou le rôle

- **Les agents d'information (agents Internet) :** la fonction principale de ces agents est de gérer un grand volume d'information à travers le Web via les moteurs de recherche. Ils peuvent être stationnaires ou mobiles; stationnaires comme les agents «gestionnaires

de courrier » ou « secrétaires virtuelles » ou mobiles pour pouvoir naviguer sur la toile pour chercher les informations et les amener vers leur destination.

- **Les agents de bases de données :** ces agents jouent un rôle important dans la répartition des données sur plusieurs serveurs. Ils sont souvent utilisés dans l'exploration des données « datamining », la collecte des données, la recherche de l'information, le traitement parallèle des requêtes, etc…
- **Les agents de détection d'intrusions :** ce sont des agents conçus pour la sécurité informatique afin de repérer les tentatives de nuisance au matériel ou au logiciel via des systèmes de détection d'intrusions (IDS). La détection d'intrusions consiste à scruter le trafic réseau, collecter tous les événements, les analyser et générer des alarmes en cas d'identification de tentatives malveillantes.
- **Les agents de commerce :** ces agents facilitent les opérations commerciales effectuées en ligne telles que les visites des galeries ou magasins virtuels et la recherche des offres commerciales. Deux types d'agents de commerce sont distinguées : les agents contrôlés par les clients et ceux contrôlés par les producteurs.

IV.2.2.3. Classification des agents selon la mobilité

- **Les agents stationnaires :** cet agent agit localement, pendant tout son cycle de vie, dans la machine là où il a été implanté initialement.
- **Les agents mobiles :** contrairement à un agent stationnaire, un agent mobile est capable de se déplacer à travers un réseau, d'un nœud à un autre pour agir à son propre compte ou à la demande d'un autre agent. C'est un paradigme de plus en plus utilisé dans le contexte des réseaux largement distribués. La technologie d'agents mobiles est évoquée plus en détail dans la section IV.4.

En fait, l'agent en tant qu'entité individuelle peut s'avérer limité dans des bien des cas, surtout au vu de ce qu'on voit aujourd'hui comme applications distribuées et pour lesquelles un ensemble d'agents mettant en commun compétences et connaissances parait plus que nécessaire. Ces types d'agents forment ce qu'on appelle des systèmes multi-agents que nous détaillons dans la section suivante.

IV.3. Système Multi-Agent

Un système multi-agent (SMA) est un système constitué d'un ensemble d'agents coopérants pour atteindre un but global ou un ensemble de buts distincts. En se basant sur

des techniques de coordination et de négociation, les agents peuvent orchestrer leurs actions et arriver à des consensus. Ceci permet d'avoir un comportement cohérent qui optimise la productivité des solutions émergentes. Les SMA sont généralement utilisés pour représenter des systèmes où: les composants opèrent avec des informations incomplètes sur l'environnement (chaque composant peut être représenté par un agent), le contrôle des évènements et l'accès aux données sont décentralisés, et la communication est asynchrone (Sycara, 1998a).

Les SMA son célèbres pour leurs faculté à représenter et à s'adapter aux problèmes réels et de nature complexe. Ils le sont également grâce aux performances et caractéristiques que l'on y trouve (Sycara, 1998b):

- ***Fiabilité:*** la distribution des tâches sur des agents spécifiques, facilite la résolution des défaillances d'un système rapidement et efficacement.
- ***Extensibilité:*** l'indépendance des agents les uns vis à vis des autres facilite la modification de leurs comportements.
- ***Robustesse:*** la coopération entre agents permet au système de faire face à des situations incertaines et d'augmenter la tolérance aux fautes, vu que l'échec d'un agent n'a pas d'influence sur le fonctionnement des autres.
- ***Maintenabilité:*** l'inter-indépendance entre agents permet également de maintenir chaque agent séparément des autres sans affecter le fonctionnement global du système.
- ***Evolutivité et Flexibilité:*** la faculté d'auto-adaptabilité permet aux développeurs d'ajouter ou de supprimer de nouvelles contraintes (ou même de nouveaux agents) au SMA sans pour autant altérer le mécanisme global du système.
- ***Efficacité:*** la capacité de communication entre agents permet de développer des systèmes de calcul distribués très puissants et très efficaces.
- ***Réduction des coûts:*** un système centralisé est toujours gourmand en temps de développement et de maintenance, avec les SMA on peut décentraliser la gestion et l'exécution des tâches, en créant des sous-systèmes représentés par des agents spécifiques, ce qui réduit les coûts d'extension et de maintenance.

IV.4. Les agents mobiles

Dans cette section nous présentons les principaux concepts liés à la technologie d'agent mobile.

IV.4.1. Définition

Un agent mobile est un logiciel autonome et auto-adaptable qui a la faculté de migrer d'une machine à une autre et de suspendre ou changer son comportement selon les évènements et les circonstances qu'il rencontre.

La définition de Gray et al. nous semble la plus distinctive (Gray, 1996): *« A mobile agent is an executing program that can migrate, at times of its own choosing, from machine to machine in a heterogeneous network. On each machine, the agent interacts with stationary service agents and other resources to accomplish its task ».*

Kalchuk et Karmouch ont étendu cette définition en ajoutant: «*The agent decides when and where it will migrate, and may interrupt its own execution and continue elsewhere on the network*» (Nguyen, 1998).

En général un agent mobile est composé : d'un code (statique), d'un état et de données. Lors de la migration d'un agent mobile d'une plate-forme à une autre, ses données et son état peuvent changer. On distingue deux types de mobilités: une forte mobilité, qui consiste à transférer le code, les données ainsi que l'état de l'agent, et une faible mobilité qui consiste à transférer le code et les données seulement (Jeffrey, 2006).

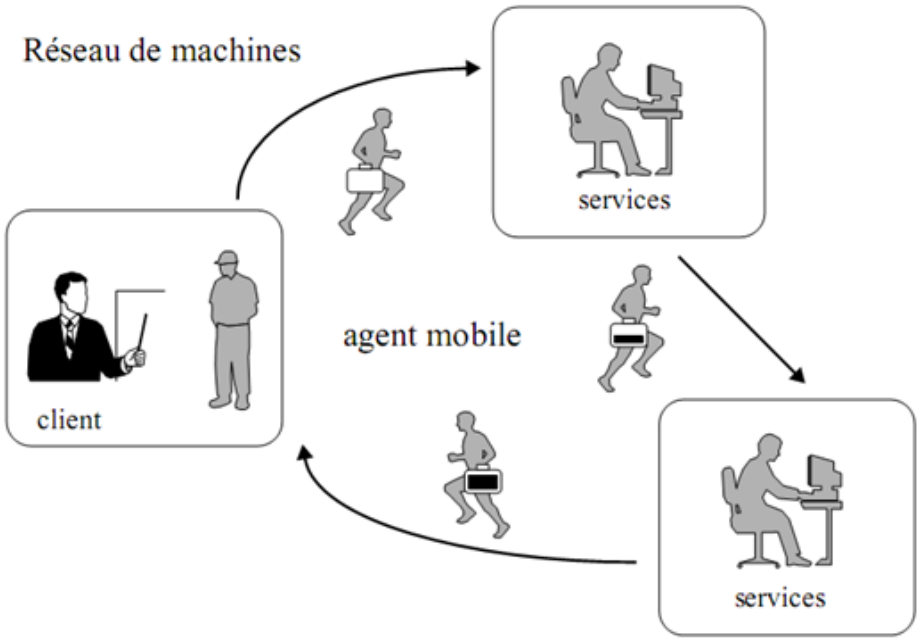

Figure IV.1: Le paradigme d'agent mobile (Perret, 1997).

IV.4.2. Structure d'un agent mobile

Un agent mobile est une entité qui possède cinq attributs : son état, son implémentation, son interface, son identifient et son autorité. Quand un agent se déplace à travers le réseau, il transporte ses attributs (ELFalou, 2006):

- **L'état:** l'état d'un agent peut être considéré comme une photo instantanée de son exécution. Quand un agent voyage, il transporte avec lui son état, ceci lui permet de reprendre son exécution quand il à arrive à destination.
- **L'implémentation:** comme n'importe quel autre programme, l'agent mobile a besoin d'un code pour pouvoir s'exécuter. Quand il se déplace à travers le réseau, l'agent peut soit emporter son code soit aller à destination, voir quel code est disponible sur la machine distante et récupérer le code manquant à partir du réseau (c'est la technique du « *code on demande* »).
- **L'interface:** un agent fournit une interface qui permet aux autres agents et autres systèmes d'interagir avec lui. Cette interface peut être un ensemble de méthode qui permet aux autres agents et applications d'accéder aux méthodes de l'agent par un système de messagerie.
- **L'identifiant:** chaque agent possède un identifiant unique durant son cycle de vie, qui lui permet d'être identifié et localisé. Puisque l'identifiant est unique, il peut être utilisé comme clé dans les opérations qui exigent un moyen pour référencer une instance particulière d'agents.
- **L'autorité:** une autorité est une entité dont l'identité peut être authentifie par n'importe quel système auquel elle essaye d'accéder. Une autorité peut être soit une personne privée, soit une organisation. L'identité est constituée d'un nom et d'autres attributs.

IV.4.3. Migration d'un agent

La migration est le processus par lequel les agents mobiles se transportent vers d'autres ordinateurs pour y continuer leur exécution (Bernier, 2000). Pour migrer, un agent mobile utilise une instruction spéciale qui indique à son environnement d'exécution qu'il désire être transporté vers un autre ordinateur. L'agent peut utiliser cette instruction à n'importe quel moment de son exécution. Lorsqu'il le fait, son exécution est interrompue et son état sauvegardé pour ensuite être transporté vers un autre ordinateur. Lorsqu'un serveur d'agent reçoit un agent, il recrée l'agent localement et lui redonne son état sauvegardé. Ensuite, le

serveur redémarre l'agent nouvellement créé ce qui complète le processus de migration. Il existe deux types de migration (Bourdon, 2006) : forte et faible.

IV.4.3.1. La migration forte

La migration forte permet à un agent de se déplacer quelque soit l'état d'exécution dans lequel il se trouve. Dans ce type de migration l'agent se déplace avec son code, son contexte d'exécution et ses données. Dans ce cas, l'agent reprend son exécution après la migration exactement là où elle était avant son déplacement. La migration forte nécessite un mécanisme de capture instantanée de l'état d'exécution de l'agent. Elle peut être proactive ou réactive. Dans la migration *proactive*, la destination de l'agent est déterminée par l'agent lui-même. Dans la migration *réactive*, la migration de l'agent est dictée par une partie ayant une relation avec l'agent mobile.

IV.4.3.2. La migration faible

La migration faible ne fait que transférer avec l'agent son code et ses données. Sur le site de destination, l'agent redémarre son exécution depuis le début en appelant la méthode qui représente le point d'entrée de l'exécution de l'agent, et le contexte d'exécution de l'agent est réinitialisé. Pour que l'agent se branche sur une instruction particulière de son code après sa migration, le programmeur doit inclure dans l'état de l'agent des moments privilégiés (explicites) dans le code de l'agent (point d'arrêt) pour pouvoir le relancer.

La migration forte, bien que beaucoup plus exigeante à implanter que la migration faible, n'en est pas moins indispensable pour toutes les applications pour lesquelles les notions de fiabilité et de tolérance aux pannes sont primordiales.

IV.4.4. Avantages et inconvénients des agents mobiles

La technologie des agents mobiles a fait naitre un intérêt particulier domaines de la recherche relative aux applications réparties. Cependant, cette technologie nouvelle a été évaluée afin de constater et estimer ce qu'elle pouvait apporter comme caractéristiques propres et ce qu'elle permettait d'améliorer par rapport aux méthodes de programmation classiques (Bernard, 1999) (Chess, 1994). Dans cette section, nous allons mettre en relief les différents apports ainsi que les difficultés soulevées par les agents mobiles (Cros, 2005).

IV.4.4.1. La performance

En premier lieu, l'amélioration obtenue par les agents mobiles porte sur le gain de performances dues à une meilleure utilisation des ressources physiques mises à disposition. L'amélioration interviendra à différents niveaux tout en permettant d'optimiser la tâche globale des agents.

IV.4.4.1.1. Diminution de l'utilisation du réseau

Le déplacement des agents mobiles permet la réduction significative des communications distantes entre clients et serveurs. En privilégiant les interactions locales, l'utilisation du réseau se limitera essentiellement au transfert des agents. Cette situation présente trois avantages à savoir:

- **Brèves périodes de communication:** on réduit considérablement le temps de connexion entre deux sites. Ceci par l'écourtement des communications distantes aux seuls transferts d'agents mobiles. Cette diminution des communications réseaux dégage une moindre inquiétude aux ruptures de liens physiques qui souvent peuvent surgir au sein des environnements sans fil (Cros, 2005).
- **La diminution de la consommation de bande passante:** l'installation des agents mobiles permet l'obtention d'une réduction significative de la charge réseau en termes de nombre total de données transférées. Cette réduction est enregistrée dans différents types d'applications nécessitant d'intenses échanges d'informations entre client et serveur (la collecte d'informations dans des bases de données réparties, l'exploration d'un internet ou encore la gestion de réseaux) (Gray, 2002) (Sahai, 1998).
- **La diminution des temps de latence:** aux réseaux à large échelle, la mise en place d'applications reparties qui nécessitent des interactions fréquentes entre client et serveur, se trouve en opposition avec un temps de latence propres aux communications réseaux. Il arrive souvent de constater que le temps d'attente de la réponse d'une requête est plus long que le temps nécessaire au traitement de la réalisation du service. Il se trouve que si on rapproche client et serveur au un même site, on les place dans un environnement où les temps de réponse des interactions sont limités, ce qui nous permet de réduire le temps de latence (Johansen, 1998) (Jun, 2002).

IV.4.4.1.2. Des calculs indépendants

Ainsi au modèle Client - Serveur, le client et le serveur restent connectés durant le temps d'exécution. En plus, on relève des problèmes liés aux fluctuations des performances réseaux. On remarque que certains services qui nécessitent de longues périodes de traitement, supportent difficilement une rupture de connexion avec le client. Afin de résoudre ce problème, ils (client-serveur) doivent souvent redémarrer entièrement leurs calculs. Mais, le maintien du lien de communication s'avère difficile aux réseaux sans fil. Avec les agents mobiles, un client peut déléguer les interactions avec le service sans entretenir une connexion de bout en bout. Avec cette possibilité de calcul indépendant, le client peut demander un service, se déplacer puis revenir pour récupérer finalement les résultats (Gray, 2001).

IV.4.4.1.3. Optimisation du traitement

L'optimisation des phases de traitement se produit à deux niveaux (Ismael, 1999).

- ✓ En localisant les ressources et le savoir-faire sur un même site, on se débarrasse des phases de dialogue entre client et serveur qui sont perturbés par des temps de latence dus aux communications réseaux.
- ✓ Le déplacement du savoir-faire nous permet de déléguer les calculs sur des machines serveurs, qui sont souvent plus puissantes qu'une machine cliente.

IV.4.4.1.4. Tolérance aux fautes physiques

Les agents mobiles se déplacent avec leur code et leurs données propres peuvent s'adapter aisément aux erreurs systèmes. Ces erreurs peuvent être d'ordre purement physique (disparition d'un nœud) ou d'ordre plus fonctionnel (arrêt d'un service). Si on prend le cas d'un site qui perd une partie de ses fonctionnalités, un service tombant en panne, l'agent peut choisir le déplacement vers un autre site contenant la fonctionnalité voulue. Ceci permet une meilleure tolérance aux fautes (Rouvrais, 2002).

IV.4.4.2. La conception

Les agents mobiles présentent à la fois une méthode qui permet de mieux définir certaines applications. Cependant, ils apportent aussi une complexité accrue par rapport au modèle classique client-serveur (bien mieux maîtrisé) (Cros, 2005).

- Souvent, les concepteurs favorisent l'obtention d'une méthode permettant une description facile d'un comportement réel (Milojicic, 1999). Cependant, avec la méthode classique, il est très contraignant de décrire des algorithmes d'exploration (de réseau) ou bien encore de caractériser les déplacements des utilisateurs nomades. A l'aide des agents mobiles, les concepteurs disposent d'une méthode qui permet de décrire naturellement ce genre de comportement. Ainsi, on peut aisément mettre en place un déploiement et/ou une maintenance d'application sur un réseau et de suivre les utilisateurs dans leurs déplacements (Picco, 1998) (Satoh, 2002).
- Les agents sont dotés d'une capacité de traitement spécifique leur permettant de s'adapter à leur environnement. Fréquemment, au modèle client-serveur et préalablement le service est caractérisé par une interface stricte, avec un protocole d'utilisation bien défini. Ainsi, si le client ignore la description du service, il ne pourra pas l'utiliser. Par contre, les agents pourront eux s'adapter aux caractéristiques des services afin d'exprimer leur requête. On cite comme exemple, si un service demande une communication sécurisée, l'agent pourra récupérer un module de sécurité, mettre à jour sa pile de communication et dialoguer avec le service (Inagaki, 1999). On peut aussi imaginer que l'agent pourra s'adapter aux conditions du réseau en se séparant d'une partie de ses fonctionnalités pour s'adapter aux terminaux pauvres en ressources (Libsie, 2002).
- La capacité de raisonnement va permettre la conception d'agents qui seront autonomes; ils adaptent leurs déplacements en fonction de l'environnement et pouvant moduler leurs fonctionnalités en cours d'exécution. Cependant ces propriétés s'accompagnent d'une conception bien plus difficile que celle du modèle classique client-serveur (Vigna, 2004). En effet, dans la majorité des applications distribuées, on essaie de cacher le plus possible la répartition en s'appuyant sur un ensemble de services système qui permettent de concevoir une application comme si tous ses éléments étaient locaux (CORBA) (Geib, 1999). Afin de tirer pleinement partie des agents mobiles, la distribution doit être explicite et gérée par les agents eux-mêmes, ce qui complique la tâche des concepteurs. Cependant, si on se réfère à la conception objet où une application est définie comme un ensemble d'éléments et de relations, il est difficile de bien définir la tâche de chaque agent (élément) et surtout de savoir comment et où les interactions (relations) vont s'opérer.

- Enfin, la complexité même des services rendus par les applications impose souvent aux concepteurs l'utilisation de méthode parfaitement maîtrisée plutôt que de recourir à un mécanisme méconnu (Cros, 2005).

IV.4.4.3. Le développement

Lors des phases de développement, le domaine des agents mobiles rencontre de fortes contraintes à savoir:

- En première lieu, il existe actuellement un nombre élevé de plateforme pour agents mobiles qui possèdent chacun d'eux ses propres défauts et qualités (Johansen, 2004). Ce qui soulève la difficulté de parler d'une standardisation. Sachant que les agents doivent s'adapter aux conditions de l'environnement. Les développeurs sont confrontés à un éventail de possibilités. Par manque de standardisation le défaut se trouve aussi dans les interactions entre agents, ainsi il n'existe pas de langage partagé par toutes les plateformes. Ceci représente un sérieux handicap. Les agents ont besoin d'exprimer les caractéristiques des services qu'ils recherchent; afin d'obtenir des réponses précises sur leur localisation et la manière de les utiliser. Par la même, les développeurs sont en face à large éventail de possibilités du fait qu'il existe plusieurs langages. Pour résoudre ce problème, il faudra se tourner vers le domaine des systèmes multi-agents qui cherche à mettre en place des langages précis en s'appuyant sur un ensemble d'ontologies et de protocoles caractérisant les interactions inter-agents (Ametller, 2003).
- Le deuxième handicape, lors du développement c'est la complexité de la mise au point des programmes formant la partie critique du processus de développement. La plupart des environnements de programmation possèdent des outils permettant de suivre les différents éléments d'une application durant son exécution afin de dénicher les erreurs qui pouvaient surgir. Cependant, ce genre d'outil se manipule aisément avec des éléments statiques mais, se maitrise avec difficulté lors du déplacement des éléments. Ceci est particulièrement vrai dans les architectures hybrides, à grande échelle où il est quasiment impossible de contrôler la totalité d'un système donné. Ainsi, il s'avère impossible de récupérer l'état d'erreur d'un élément se localisant sur un site déconnecté. En l'absence outil de suppression des bogues, il est difficile de différencier les erreurs de conception et de développement. Il est à signaler que des efforts sont

fournis afin de proposer un premier outil de suppression des bogues est conçu par la plate-forme JADE (Bellifemine, 1999).

- Le troisième et dernier hic, en rattachement avec le débogage, on rencontre un handicape à mettre en place un contrôle des applications à base d'agents mobiles. La technique la plus usitée en ce moment, malgré qu'elle n'est pas parfaitement fiable, est l'opération test. A cette méthode, on met à l'épreuve une application en lui injectant des données en ses différents points d'entrée dans le but de simuler le comportement utilisateur et de recouvrir une plage de situations la plus large possible. Si le test s'applique plutôt naturellement dans les programmes classiques, i.e. non répartis, il représente un domaine de recherche à part entière dès qu'il s'agit d'applications réparties (Ulrich, 1999) (Benattou, 2002). Dans cette optique, il faut maitriser la coordination des différentes injections sur les points d'entrée répartis. Ceci afin de garantir que le comportement simulé est bien celui qui était visé. Cette méthode est complexe dans le cas d'une application répartie statique. Si cette dernière est difficilement contrôlable que peut-on dire si les points d'entrées (les agents) deviennent mobiles. Pour vérifier les applications construites sur des agents mobiles, on applique une méthode permettant la garantir du comportement et les interactions des agents avant leur déploiement. L'analyse statique de programme permet de déterminer statiquement, sans exécuter un programme, des propriétés qui seront satisfaites lors de l'exécution (Colaço, 1997). Cette méthode, nous permet de garantir un certain niveau de correction du programme analysé ; elle semble être préconisée afin de se passer des phases de test et, ce, même dans le cas des applications réparties (Kobayashi, 1995) (Pierce, 1993). Il est à souligner que l'analyse statique a fait ses preuves dans le cadre de la programmation classique. Toutefois elle constitue un domaine de recherche pour les applications réparties.

IV.4.4.4. La sécurité

Dans ce domaine, la sécurité constitue le talon d'Achille des agents mobiles. Du point de vue logiciel, la sécurité consiste à inhiber les accès ou les modifications, non-autorisés aux éléments d'un système informatique. Dans cette vision, nous somme amené à faire la distinction entre les demandeurs et les éléments demandés. Si on veut avoir un système sûr et sécurisé, il est nécessaire de mettre en place une politique en ce sens, garantissant la confidentialité (les données des éléments ne sont pas divulguées aux demandeurs non-

autorisés) et l'intégrité (les éléments ne sont pas modifiés par des demandeurs non-autorisés). Nous omettons volontairement la disponibilité qui doit garantir, aux demandeurs autorisés, l'accessibilité aux éléments et qui est plus particulièrement du ressort de la tolérance aux fautes (Cros, 2005).

La mise en place d'une politique de sécurité, demande généralement la mise en place de mécanismes à savoir:

- **Authentification (mot de passe, certificat):** l'authentification a pour but d'identifier avec précision le demandeur,
- **Cryptographie:** la cryptographie doit assurer la confidentialité des données échangées,
- **Contrôle d'accès (droit utilisateur, pare-feu):** le contrôle d'accès vérifie l'adéquation entre demandeur et éléments demandés.

Au schéma classique des applications réparties, on regroupe les éléments critiques sur des machines sécurisées et on les focalise sur les canaux de communication externes tout en utilisant les mécanismes d'authentification et de cryptographie. La mise en place de la mobilité de code, les politiques de sécurité recoure plutôt à la relation étroite entre le site stockant le programme et celui qui l'exécute. Implicitement, le possesseur du code est le même que celui de l'environnement qui l'exécute.

Pour ce qui est des agents mobiles et du point de vue sécurité, nous sommes devant un problème non encore résolu intégralement. C'est d'ailleurs le principal argument avancé pour expliquer la faible utilisation de ce paradigme. En effet, les agents mobiles représentent un nouveau champ d'investigation pour la recherche en matière de sécurité. On protège les sites vis-à-vis des agents malveillants, ceci d'une part, d'autre part on sécurise les agents vis-à-vis des sites malveillants. Nous intervenons sur deux points essentiels à savoir: protection des sites et protection des agents.

IV.4.4.4.1. Protection des sites

La sécurisation des sites à l'encontre des attaques menées par les malveillants pose un problème, actuellement maîtrisé. En effet, plusieurs solutions permettent de déjouer d'éventuelles attaques. Les méthodes les plus connues sont (Cros, 2005):

- **Bac à sable:** la technique du bac à sable consiste à exécuter un agent à l'intérieur d'un environnement restreint, en interdisant l'accès au système de fichiers par exemple.
- **Signature de code:** la signature du code intervient lors de la création d'un agent, son créateur le signe numériquement afin qu'il puisse s'identifier durant ses déplacements.

- **Contrôle d'accès:** l'amélioration des deux techniques précédentes, demande la mise obligatoire en place d'une politique de contrôle d'accès plus complexe. On peut la voir comme un raffinement d'une politique de bac à sable générale vers une politique spécifique à chaque application ou classe d'agents.
- **Vérification du code:** la vérification de code permet d'acquérir une garantie sur la sémantique d'un code à travers l'analyse de sa structure, ou de son comportement pour un agent, en fonction d'une politique de sécurité donnée. Les bacs à sable réalisent des vérifications rudimentaires en cours d'exécution, ceci pour garantir avant tout le typage des opérandes des instructions, mais il s'avère très coûteux. On utilise une deuxième approche qui se base sur la vérification automatiquement de code avant son lancement, s'appuyant sur une preuve de conformité. Pour cela on peut utiliser l'approche PCC (Proof Carrying Code).

IV.4.4.4.2. Protection des agents

À l'inverse de la protection des sites, la protection des agents contre des sites malveillants ne dispose pas de solution éprouvée et demeure encore un champ ouvert à la recherche. Pour imaginer ce que risque un agent lors de son exécution sur un site malveillant, nous pouvons prendre comme référence les éléments transportés pouvant être prises comme cible d'attaque (Cros, 2005) (Loureiro, 2001):

- **Le code:** ensemble d'instructions composant la tâche de l'agent.
- **Les données statiques:** données ne variant pas durant les déplacements (la signature par exemple)
- **Les données collectées:** ensemble de résultats obtenus au cours des déplacements réalisés par l'agent depuis son lancement.
- **L'état courant:** ensemble de données servant à l'exécution courante de l'agent.

La sécurité des agents mobiles consiste à offrir la garantie des critères de confidentialité et d'intégrité de l'ensemble de ces éléments. Du points de vue données, il est évident qu'un agent ne désire pas divulguer les informations critiques à n'importe quel site. Par exemple, un site malveillant pourrait récupérer la signature d'un code et l'utiliser pour créer un nouvel agent afin de s'introduire dans des environnements auxquels ils lui sont interdits normalement. Pour le code, un agent transporte un savoir-faire propre à son concepteur qui pourrait tomber aux mains des intrus. Dans (Cros, 2005), l'auteur classe en trois importantes catégories d'attaques que les sites sont capables d'accomplir:

- **L'inspection:** elle consiste à l'examen du contenu de l'agent, ou le flot d'exécution afin de récupérer les informations critiques transportées par l'agent.
- **La modification:** elle s'accomplie par le remplacement de certains éléments de l'agent dans le but d'une éventuelle attaque. En remplaçant le code, l'agent effectuera des opérations malveillantes sur les futurs sites à visiter.
- **Le rejeu:** s'obtient en clonant l'agent puis en exécutant le clone dans plusieurs configurations pour retrouver le savoir de l'agent.

Plusieurs techniques sont en cours d'étude pour garantir aux agents qui peuvent accéder en toute confiance à des nœuds ou pour détecter les agents intrus. Cependant, aucune d'elle ne peut fournir une sécurité suffisant comme celle proposée à la protection des sites. Il est aussi important de signaler qu'en matière de problèmes de sécurité liés à l'utilisation des agents mobiles, celle-ci est encore un champ d'investigation complet. Bien que la protection des sites est à présent maîtrisée, la protection des agents est insatisfaisante à l'heure actuelle.

IV.4.4.5. Bilan des avantages et inconvénients

À travers les différentes caractéristiques des agents mobiles, nous pouvons dire que les agents mobiles ne représentent pas le paradigme idéal pour tous les types d'applications réparties, mais simplement une nouvelle possibilité à la construction des applications. Selon les besoins de performance, de conception, de développement et de sécurité les agents mobiles pourront apporter une alternative intéressante au modèle classique client-serveur dans certains types de configuration (Cros, 2005).

Le critère de performance des agents sera très utile dès que les applications vont comporter des communications intensives. Par leurs interactions locales, les agents permettent de ne pas subir les ralentissements dus aux bandes passantes limitées et aux temps de latence des réseaux (surtout sur Internet). En utilisant les machines serveurs plus performantes que les clients, les phases de traitement seront plus rapides. Du point de vue conception, les agents mobiles permettent de décrire les comportements difficiles à caractériser avec le modèle client-serveur. Ainsi, l'exploration d'un réseau, la représentation d'un utilisateur, le support des ruptures de communication ou encore l'adaptation à l'environnement sont naturellement exprimables grâce aux agents mobiles.

D'autre part, ces qualités font ressortir d'importantes difficultés de développement. Ces dernières sont dues essentiellement au manque de standardisation des plateformes qui

n'offrent pas l'obtention d'un langage homogène partagé par tous les agents. Ajoutons à cela, les difficultés de mise au point dues aux déplacements de l'unité d'exécution qui sont difficiles à suivre et aux problèmes de test qui nécessitent d'importants efforts de coordination. Ces difficultés de développement poussent les concepteurs à s'orienter vers des paradigmes bien mieux maîtrisables en mettant en place des mécanismes de mobilité plus limités. D'un point de vue plus informel, la programmation classique dispose d'outils graphiques d'aide au développement qui intègrent complètement les phases de programmation, de déboguage et de test. Quelques outils du même type apparaissent pour les agents mobiles mais ne sont pas totalement intégrés en restant pour la plupart focalisés sur la phase de programmation et seuls une minorité s'intéresse à la phase de déboguage.

Finalement, nous pouvons conclure que le problème majeur empêchant l'adoption actuelle des agents mobiles est la sécurité. En effet, même si la protection des sites est quasiment assurée, celle des agents reste un réel problème non résolu définitive. Différentes études sont actuellement menées pour permettre l'obtention d'un niveau de sécurité satisfaisant.

IV.4.5. Plates formes et standardisation

Dans cette section, nous présentons deux normes développées pour les agents mobiles (FIPA, MASIF), ainsi que certaines plates-formes existantes (voir tableau IV.1).

IV.4.5.1. Les normes

L'émergence des nombreuses plates-formes expérimentales a rendu nécessaire la proposition d'une harmonisation grâce à la standardisation des différents concepts communs pouvant être identifiés. Cette normalisation devrait permettre à terme de rendre compatibles les différents systèmes. La technologie agent faisant l'objet de nombreux travaux, une intégration des concepts clés est apparue essentielle avec comme finalité la production de spécifications destinée à l'industrie. C'est l'objectif que s'est fixé la (Foundation for Intelligent Physical Agents, FIPA) que de promouvoir le développement et la spécification de la technologie agent (FIPA , 2004). De même (Object Management Group, OMG) a étendu le champ de ses travaux initialement fondé sur le paradigme objet pour intégrer la technologie agent mobile en vue de produire des spécifications, Depuis peu, les deux organismes font des efforts conjugués pour fusionner les deux approches.

On peut trouver à l'heure actuelle deux normes principales (Cros, 2005) :

- **FIPA (Fondation for Intelligent Physical Agents)**

En revanche, la communauté d'origine de FIPA étant celle des systèmes multi-agents, plus proche de l'intelligence artificielle, FIPA s'intéresse plus particulièrement à l'interopérabilité des agents intelligents (les efforts sont placés au niveau du langage, des protocoles et des infrastructures de communication), elle va se situer à un niveau plus élevé c'est à dire le niveau applicatif, en décrivant les éléments nécessaires à la réalisation d'une application et principalement en détaillant la communication entre les agents. Le but est de décrire un ACL (Agents Communication Language) et des protocoles de négociation permettant ainsi de définir parfaitement les interactions entre les agents.

- **MASIF (Mobile Agent System Interoperability Facility)**

La norme MASIF a été spécifiée par l'Object Management Group (OMG) qui se préoccupe généralement de l'hétérogénéité entre les systèmes, l'OMG s'intéresse à l'interopérabilité des agents mobiles à travers sa spécification appelée MAF. Dans cette optique, le but, dans la norme MASIF, est de décrire les notions élémentaires permettant l'échange des agents entre différentes plates-formes. Pour ce faire, elle standardise la manière de gérer le code des agents, leur identification, la migration et l'adressage local (ELFalou, 2006).

IV.4.5.2. Implémentations existantes

A l'heure actuelle, il existe beaucoup d'implémentation, incompatibles entre elles. Certaines de ces plates-formes existantes à nos jours sont présentées ci-dessous :

- **LIME:** (Linda In Mobile Environnement), est un intergiciel basé sur Java qui propose une couche de coordination pour les agents, en réutilisant le modèle de synchronisation Linda. Ce n'est pas à proprement parler une plate-forme car elle ne prend pas en charge les différents éléments décrits par les normes.
- **AGLETS:** la technologie Aglets et son environnement de développement ASDK (Aglets Software Development Kit) sont actuellement maintenus au travers d'un site communautaire, selon un modèle libre et open source. L'ASDK permet de programmer rapidement et simplement des agents logiciels mobiles en Java. Le modèle d'agent est simple, associant à un objet java un thread et des méthodes d'activation spécifiques.

L'exécution est supportée par un serveur d'Aglets nommé Tahiti qui permet de gérer les communications, la sécurité et la mobilité des agents.

- **TACOMA:** le projet TACOMA (Troms And COrnell Moving Agents), de Norway & Cornell University, a été développé dans le but d'offrir différentes abstractions de haut niveau pour les agents. L'accent est mis sur un découplage, entre le langage et le niveau agent, semblable à celui introduit dans les normes à travers la notion de place.
- **PLANGENT:** une plate-forme basée sur le langage Java, s'intéressant à l'adaptation des agents durant leurs déplacements au sein des environnements dynamiques. Il s'agit de l'intergiciel le plus proche de l'intelligence artificielle. Son but est de proposer aux agents des mécanismes pour modifier les objectifs intermédiaires nécessaires à la réalisation de leur tâche finale.
- **JADE:** (Java Agent Development Framework), est une plateforme construit sur Java afin de permettre une programmation multi-agents simplifiée. C'est une plateforme très abouti qui offre un environnement graphique pour l'aide au développement et à l'administration. Le container est construit comme une agence contenant une unique place exécutant une machine virtuelle Java. Une région est alors définie par un ensemble de containers distribués correspondant concrètement à la plate-forme JADE.

Le tableau récapitulatif IV.1 regroupe les principales caractéristiques des plates-formes existantes.

Plateforme	**Agents**		**Communication**		**Langage**	**Référence**
	Réac/Pro	**Migration**	**sync/async**	**locale/distante**		
Aglets	Proactif	Faible	les deux	les deux	Java	(Jones, 2002)
D'Agents	Proactif	Forte	les deux	les deux	multiple	(Gray, 2002)
JADE	les deux	Faible	asynchrone	les deux	Java	(Bellifemine, 1999)
JavAct	Proactif	Faible	les deux	les deux	Java	(Arcangeli, 2004)
LIME	Supportées		synchrone	les deux	Java	(Picco, 99)
PLANGENT	Proactif	Faible	synchrone	locale	Java	(Ohsuga, 1997)
TACOMA	Proactif	Faible	les deux	locale	multiple	(Johansen, 2002)
Telescript	Proactif	Forte	synchrone	les deux	Telescript	(Domel, 1996)

Tableau IV.1: Tableau comparatif des caractéristiques générales des plates-formes (Cros, 2005).

IV.5. Les travaux relatifs

De nombreux chercheurs dans le domaine affirment que les services Web seuls sont considérés comme passifs, tandis que les agents peuvent fournir des alertes et des mises à jour lorsque de nouvelles informations deviennent disponibles. Malheureusement, ces technologies ont été à l'origine développées séparément avec différentes normes et caractéristiques. En conséquence, leur intégration devient importante dans ce contexte.

Avec ce paradigme, les composants logiciels où chacun représente un service et un agent en collaboration, vont interagir pour fournir des services unifiés. Ceci correspond bien avec la prédiction de (Huhns, 2005) *« Les agents deviendront une partie essentielle de la plupart des applications Web, servant comme une colle qui permet à un système aussi grand que le Web d'être maniable et viable »*.

Plusieurs approches existent dans la littérature afin d'intégrer les services Web et les agents dans des divers domaines d'applications :

- (Nguyen, 2005) proposent une solution améliorée, WS2JADE, pour l'intégration des services Web avec un système multi-agent. Cette approche possède une couche d'interconnexion qui contient des agents spéciaux, appelé "Web Service Agent" ou WSAG. Ces WSAGs collent les agents et les services Web ensemble, ainsi, ils offrent des services Web en tant que leurs propres services. La deuxième couche s'appelle couche de gestion, qui est capable de faire une découverte active des services et de produire et de déployer automatiquement des WSAGs en temps d'exécution. La combinaison de deux couches statique et dynamique est un outil distinct de WS2JADE par rapport aux différents outils existants, qui est conçu pour découvrir activement des services Web et les enregistrer dans le système multi-agents.
- Dans (Baousis, 2006), l'article expose une nouvelle architecture pour la découverte dynamique. De plus, les services Web sont exprimés en OWL-S. Dans ce contexte, Baousis et al exploitent les capacités offertes par les agents mobiles pour interroger et exécuter sémantiquement les services Web. En outre, l'agent mobile est équipé d'une intelligence approprié pour accomplir leur tâche de manière dynamique.
- (Foukarakis, 2007) propose une plateforme qui fournit un temps d'exécution rapide d'un environnement d'agents qui se situe à l'intérieur d'un conteneur Web, ce qui ajoute des fonctionnalités d'agents aux serveurs Web existants. Les composants de la plateforme sont déployés en tant que services Web, avec SOAP (Simple Object Access Protocol) sur HTTP agissant en tant que canal de communication par le biais de

messages XML standards. De cette manière, le support d'agent peut être ajouté à l'infrastructure Web existante, sans besoin de remplacer les composants ou d'installer un logiciel client.

- Ketel décrit une architecture d'intégration des services Web sémantiques avec les agents mobiles (Ketel, 2009). Cette architecture permet l'intégration sans changer les spécifications existantes. De plus, il montre la façon de composer les simples services dans un workflow pour former un service composite complexe.
- Xining et al (Xining, 2010) présentent une architecture, qui intègre la technologie d'agent mobile et la sensibilité du contexte, pour les applications M-commerce. Ils utilisent les ontologies afin de fournir des informations contextuelles personnelles et de l'environnement. De plus, pour soutenir la composition de services sensibles au contexte.
- Chiro Satoh présente un système sensible au contexte basée sur l'agent mobile (Satoh, 2010). Le système est appliqué dans les lieux publics pour ne citer: les musées. Aussi, l'auteur propose un service guide dans un musée. Quand un utilisateur est en face d'une pièce au musée, le système détecte en ce moment même l'emplacement de l'utilisateur. Il envoi l'information à un ordinateur sise à proximité de l'utilisateur via l'agent mobile lui fournissant une annotation sur l'objet. L'utilisateur dispose d'un appareil mobile.
- Dans (El-Sayed, 2012), l'auteur propose une architecture d'intégration des agents mobiles avec les services Web sémantiques RESTful dans les environnements mobiles. L'architecture proposée résout le problème de la déconnexion fréquente.
- Abedi et al proposent une architecture nommée AMF (Adaptation Management Framewok) (Abedi, 2012). Cette architecture est destinée à la gestion du service de paiement mobile. En outre, ils utilisent la technologie d'agent pour un paiement automatique et intelligent.
- Dans le document (Kravari, 2012), les auteurs présentent une conception et implémentation pour une approche de découverte des SW à base d'agents. Ces services se distinguent en deux catégories principales : un registre avancé pour les services Web sémantiques et leur découverte (pages jaunes) ; et la découverte des services Web basée sur la confiance qui supportent les outils de suivi d'interactions. Leur mise en œuvre y compris : l'interface graphique associée et intégrée dans le Framework EMERALD ; un Framework fondé sur la connaissance des agents interopérer.

- Dans l'article (Huang, 2013), la sélection des services Web et le serveur dynamique sont optimisées par la maximisation de la qualité de service (QoS) et la réduction des coûts d'énergie. Les modèles stochastiques pour les systèmes de services Web sont proposés, et les techniques d'analyse quantitative de la performance et la consommation d'énergie sont étudiées. Les auteurs formulent la sélection des services et la mise à l'échelle de la vitesse (speed scaling) comme un problème de décision de Markov, et introduisent des algorithmes associés pour le résoudre. En outre, les auteurs construisent un Framework d'optimisation à base d'agents, ils conçues des algorithmes efficaces pour résoudre le problème dans les systèmes de services Web à grande échelle.

IV.6. Conclusion

Nous avons présenté dans ce chapitre une vision générale sur les agents et les systèmes multi-agents. Ces systèmes qui ont porté résolution aux problèmes de l'intelligence artificielle (IA) classique sont organisés dans des sociétés d'agents qui interagissent, communiquent et coopèrent entre eux pour accomplir une tâche bien déterminée. Ensuite, nous avons décrit la nouvelle technologie, c'est la technologie d'agents mobiles. Les agents mobiles sont des entités logicielles autonomes qui peuvent suspendre leur exécution sur une machine et migrer avec leur code, variables et états vers une autre machine où ils reprennent leur exécution.

Par la suite, nous avons passé en revue les principaux travaux de découverte des services Web en utilisant les systèmes multi-agents et la technologie d'agent mobile. On constate que, malgré l'obtention de résultats non négligeables en ce sens, les approches qui n'utilisent pas l'agent mobile robustes aux différents facteurs de variabilité sont loin d'être atteintes. Par ailleurs, les travaux de recherche qui utilisent la technologie d'agent mobile constitue une alternative prometteuse pour résoudre les problèmes. Ainsi l'avantage principal des approches basées sur l'agent mobile réside dans le fait que l'agent mobile travaille d'une manière asynchrone et autonome.

Dans l'objectif d'optimiser la découverte des services Web en réduisant l'espace de recherche et augmentant le nombre de services pertinents. Nous avons opté à proposer comme première contribution un Framework à base d'agent mobile pour la découverte sémantique des services Web. La description de cette proposition fait l'objet du prochain chapitre.

Deuxième Partie

CONTRIBUTION

Chapitre V

Un Framework à Base d'Agents Mobiles pour la Découverte Sémantique des Services Web

V.1. Introduction

Notre approche de découverte sémantique des services Web fait appel à trois domaines de recherche importants à savoir le Web sémantique, les services Web et la technologie d'agent mobile. Le Web sémantique s'intéresse principalement aux informations statiques disponibles sur le Web et les moyens de les décrire de manière intelligible pour les machines. Les services Web, quant à eux, ont pour première préoccupation l'interopérabilité entre applications via le Web en vue de les rendre plus dynamique (Rezeg, 2010). Le paradigme agent mobile est devenu un formalisme très puissant pour le développement des applications réparties. Un agent mobile est un logiciel qui a la faculté de déplacer d'une machine à une autre pour accomplir des tâches prédéterminées (Benseghier, 2009).

La découverte des services Web représente un axe de recherche émergeant. En général, la découverte assurée par les registres UDDI, est relativement primitive. Elle ne tient pas compte de la croissance continue du nombre de services Web offerts sur le Web et des modifications constantes subies par ces derniers. La norme UDDI permet de découvrir des services Web. Cependant, elle ne permet pas de choisir le meilleur fournisseur d'un service Web parmi plusieurs offrants le même service. Elle n'offre pas de mécanisme qui permette de choisir un service Web en se basant sur sa qualité (Bentamrout, 2006). La norme manque aussi d'une sémantique compréhensible et interprétable par des machines

capables de sélectionner le meilleur service Web disponible. De plus, un UDDI centrale souffre de problème de seul point centralisé (point d'échec) et le coût élevé de la maintenance.

Afin de répondre aux limitations citées ci-dessus, nous avons proposé un nouvel Framework à base de profil utilisateur et métadonnées pour la découverte sémantique des services Web, en s'appuyant sur la capacité de la mobilité des agents. Le Framework proposé est une architecture distribuée composée de trois couches : couche demandeur des services Web, couche registre des services Web, couche fournisseur des services Web (Benseghir, 2015).

La tâche de découverte des services Web est devenue plus efficace et plus dynamique en exploitant le parallélisme et la distribution donnée par la technologie d'agent mobile. Notre travail a pour objectif de simplifier et d'optimiser la découverte des services Web en réduisant l'espace de recherche et augmentant le nombre de services pertinents.

Dans ce chapitre nous faisons en premier lieu un survol sur les concepts exploités dans notre travail à l'image des métadonnées et le profil utilisateur. En deuxième lieu nous présentons le Framework proposé en indiquant l'architecture du Framework suivie de l'aspect architectural et comportemental de chaque composant. Enfin, une étude comparative fait un bilan de ce travail.

V.2. Quelques définitions et formalismes utiles

A partir de l'état de l'art présenté dans les trois chapitres précédents, nous utilisons les concepts et les technologies cités, afin d'introduire une architecture distribuée pour la découverte sémantique des services Web. Dans cette section nous présentons quelques définitions et formalismes utiles qui nous aident à éclaircir le travail proposé.

V.2.1. Les services Web sémantiques

De manière générale, l'objectif visé par la notion de services Web sémantiques est de créer un Web sémantique de services dont les propriétés, les capacités, les interfaces et les effets sont décrits de manière non ambiguë et exploitable par des machines et ce en utilisant les couches techniques sans pour autant en être conceptuellement dépendants (Kellert, 2003).

Dans notre travail, un service Web (WS) est défini par ses fonctionnalités, ses propriétés de QoS, et d'autres informations qui permettent de décrire le service Web sémantiquement, tel que (Benseghir, 2015):

Service =< IOPE; QOS; META >

Où

IOPE = < input ; output ; precondition ; effect >

QOS = < Q_1; Q_2; … ; Q_n >

Q_k = <Fiabilité ; Disponibilité ; Prix_Exécution ; Temps_Réponse ; ...>

Il faut spécifie un poids $0 \leq W_i \leq 1$ pour chaque QoS.

META= < Type ; SpatialData ; AccessConstraints ; Goal ; ...>

V.2.2. Métadonnées et catalogue de métadonnées

Les métadonnées (Nogues, 2010) se définissent comme toutes les informations que l'on peut recueillir et mettre à disposition pour décrire une donnée, un jeu de données, un service (ex : site Web) ou un document (rapport, publication, etc.). Elles se composent d'éléments relatifs à l'identification (intitulé, description, résumé, intervenants, etc.), à la représentation spatiale et temporelle, à la qualité, au contenu, aux modalités d'accès et de diffusion des données ainsi qu'à des informations sur les métadonnées elles mêmes (date de rédaction, auteur, langue, etc.).

Le catalogue de métadonnées est un outil qui permet de rassembler et d'organiser les métadonnées, de les structurer de façon homogène, afin de les valoriser et d'en favoriser la diffusion et la consultation. Les fonctions principales attendues d'un catalogue sont d'assurer (Nogues, 2010):

- ✓ la saisie et la validation de fiches descriptives de métadonnées ;
- ✓ la recherche et la consultation des fiches descriptives ;
- ✓ la publication et la production de fiches dans un format interopérable qui permet de les mettre à disposition des partenaires et les transmettre vers des catalogues publics ou privés;
- ✓ la récupération de métadonnées pour les intégrer dans son propre système ;
- ✓ l'administration de l'ensemble du système.

Les organismes qui établissent des normes de métadonnées sont très nombreux :

Le premier producteur mondial de normes internationales est l'ISO (International Organization for Standardization) (ISO, 2005). Il définit les normes ISO 19119 et 19115 respectivement pour les métadonnées des services Web géographiques et les métadonnées des données et traitements géographiques (ISO, 2001). ISO définit aussi d'autres normes portant sur les métadonnées de façon plus générale.

Une autre organisation importante de standardisation est l'IEEE (Institute of Electrical and Electronics Engineers)[12]. Même si elle est surtout connue pour l'édiction de normes informatiques "bas niveau" (télécommunications), il définit le modèle de métadonnées des objets d'enseignement LOM.

OASIS[13] (Organization for the Advancement of Structured Information Standards), a, pour sa part, défini UDDI, standard pour les annuaires des services Web. Dans le monde du Web, justement, l'organisation incontournable est le W3C (World Wide Web Consortium). Son but est de standardiser les langages du Web. Pour l'heure, les normes W3C qui nous intéressent sont celles qui peuvent être vues comme définissant directement des métadonnées. C'est le cas de WSDL et de SOAP pour les services Web, de MathML pour les notations mathématiques.

Dans le domaine du génie logiciel, l'OMG (Object Management Group)[14] a proposé des normes célèbres comme UML, MOF, CORBA et IDL.

Dans le domaine spécifiquement géographique, l'organisme le plus connu semble être l'OGC (Open Geospatial Consortium). L'OGC s'attache à fournir des solutions techniques sous forme de définitions d'interfaces de services Web (WFS, WMS et WCS pour Web Feature/Map/Coverage Service), sous forme de formats XML de descriptions de services (implémentation d'ISO 19119), et sous forme de format XML de données (GML, Geography Markup Language) (GIS, 2001).

Après avoir étudié des exemples de métadonnées citées ci-dessus, nous avons recueilli 16 éléments des métadonnées qui représentent les attributs les plus répétés et les plus importants de notre point de vue. Le tableau suivant présente les éléments des métadonnées du catalogue proposés pour l'identification et la découverte des services Web.

[12] https://www.ieee.org/
[13] http://xmlfr.org/index/org/oasis/
[14] www.omg.org/

N°	Elément	Type	Description
01	MetadataIdentifier	Integer	Numéro unique permettant de différencier chaque métadonnée par adhérent au sein du catalogue
02	MetadataLanguage	String	La langue de métadonnées dans le catalogue.
03	MetadataCreationDate	Date	Date à laquelle la métadonnée a été créée ou actualisée.
04	MetadataContactPoint	String	Nom d'un organisme (ou d'une personne) assurant la fonction de contact pour la métadonnée.
05	ServiceID	String	Une chaîne de caractères unique permettant de différencier chaque service par adhérent au sein d'un annuaire
06	ServiceTitle	String	Un nom (libellé) donné au service, généralement utilisé pour l'affichage à l'utilisateur comme résultat
07	ServiceType	String	Décrit le type du service et permet de gérer les regroupements des services.
08	ServiceTypeVersion	String	Version de ce type de service implémentée par ce serveur.
09	ServiceDescription	String	Brève description du service, généralement disponible pour l'affichage à l'utilisateur comme résultat
10	ServiceKeywords	String	Liste non ordonnée d'un ou plus mot (s) ou phrase (s) utilisée pour décrire un service.
11	Goal	String	L'objectif visé par ce service
12	Popularity	String	Information sur le nombre d'utilisation de service
13	SpatialData	String	Des données (informations) spatial ou temporel relatives au service.
14	PublicationDate	Date	Information afin de dater les données disponibles dans le catalogue
15	ServiceContact	String	Informations pour contacter le fournisseur de service
16	AccessConstraints	String	Conditions applicables pour l'accès et l'utilisation du service

Tableau V.1: Les éléments des métadonnées proposés pour la description des services Web (Benseghir, 2015)

V.2.3. Le profil utilisateur

Dans un processus de personnalisation l'utilisateur représente le noyau central, la source, le déclencheur d'un processus de découverte et le seul à valider le résultat de cette recherche (Zemirli, 2008).

L'étude des différentes approches utilisées pour représenter le profil utilisateur nous a fait aboutir à la conclusion qu'il est nécessaire d'avoir une représentation multidimensionnelle pour pouvoir capturer l'ensemble des paramètres caractérisant l'utilisateur. Dans ce qui suit nous présentons la catégorisation proposée du profil utilisateur :

▪ Catégorie des préférences

Cette catégorie regroupe l'ensemble des informations correspondant aux préférences de recherche de l'utilisateur. C'est la catégorie la plus importante dans le profil utilisateur. Elle a une incidence directe sur le processus de découverte, puisque c'est à partir de ces informations que le système va pouvoir retourner le service Web pertinent correspondant aux besoins de l'utilisateur. Cette catégorie peut être décomposée en deux sous catégories:

- **Domaine d'intérêts** : les données représentant les centres d'intérêts de l'utilisateur.
- **Qualité** : les données représentant la qualité du service Web attendue ou espérée par l'utilisateur.

▪ Catégorie des données personnelles

Les données personnelles sont la partie statique du profil. Elles contiennent des informations qui décrivent l'utilisateur. Elle se décompose en deux sous catégories :

- **Identité :** regroupe les données d'identification de l'utilisateur.
- **Profession :** regroupe l'ensemble des données professionnelles de l'utilisateur.

La figure V.1 permet de visualiser l'ensemble des dimensions du profil utilisateur:

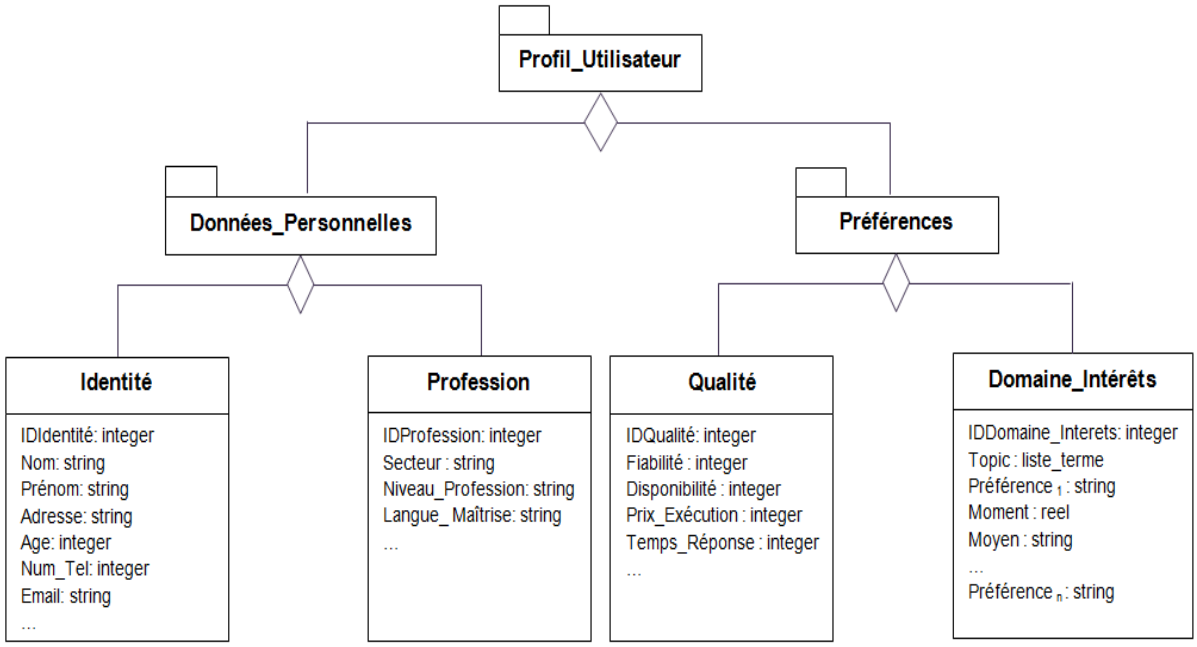

Figure V.1 : Représentation multidimensionnelles du profil utilisateur (Benseghir, 2015)

V.2.4. Requête de découverte des services Web

L'utilisateur peut ne pas avoir une idée complète et assez précise concernant le service qu'il cherche. Dans notre travail, nous considérons la requête comme étant composée d'une : (1) partie fonctionnelle: nom, description, les entrées, les sorties ; (2) partie non fonctionnelle représentée par les préférences de recherche : Type, SpatialData, AccessConstraints, Goal … ; (3) partie non fonctionnelle représentée par la qualité de service Web ; (4) une valeur entière entre 0 et 1 représentant le seuil de correspondance qu'il précise lui même.

Dans la partie non fonctionnelle l'utilisateur doit spécifie un poids $0 \leq W_i \leq 1$ pour chaque préférence (préférences de recherche, QoS).

Donc, nous décrivons une requête de découverte de services par (Benseghir, 2015):

Requête = <Partie_Fonctionnelle ; Partie_Non_Fonctionnelle ; Seuil>

Où :

Partie_Fonctionnelle = <Nom ; Description ; Entrées ; Sorties>

Partie_Non_Fonctionnelle = < Préférences_Recherche ; QoS >

Préférences_Recherche = < Type ; SpatialData ; AccessConstraints ; Goal ;…>

QoS=<Q_1 ; Q_2 ; … ; Q_n>

Q_k= <Fiabilité ; Disponibilité ; Prix_exécution ; Temps_Réponse ;…>

Seuil = <0 ; 0.1 ; … ; 1>

V.3. Architecture générale du Framework proposé

L'architecture de notre Framework est une extension de l'architecture orientée services SOA (Service Oriented Architecture). Dans l'architecture SOA: les échanges des messages sont établis par SOAP ; la description de service Web par WSDL ; la recherche des services Web via le registre UDDI (voir figure V.2).

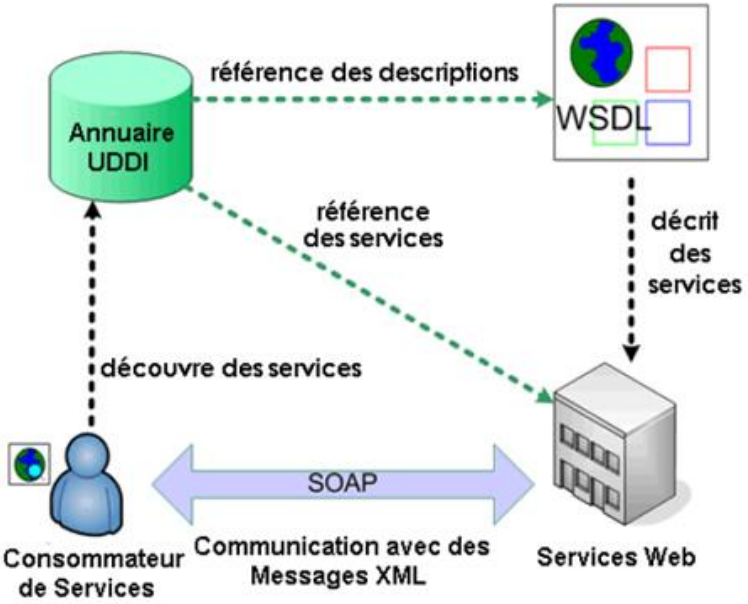

Figure V.2: Architecture Orientée services[15]

La philosophie de UDDI est de centraliser les points d'entrées des Web services, c'est à dire la création d'un annuaire central pour les services Web fournis par les différents propriétaires (entreprise ou individus) (Chauvet, 2002). Le registre UDDI fournit une API d'interrogation de son registre. Elle permet une recherche par mots clés, par tModel et donc, par voie de conséquence, une recherche par spécification (ex : classification, service utilisant WSDL, ...).

[15] wiki.nectec.or.th/bu/ITM532Students_2008/WebService

Malgré sa simplicité et sa facilité d'implémentation, elle présente quelque inconvénients tel que:

- Il souffre de problème de seul point centralisé (point d'échec) et le coût élevé de la maintenance ;
- La méthode renvoie un nombre important de résultats ou au contraire peu de résultats ;
- L'absence de sécurité et de fiabilité ;
- Elle n'offre pas de mécanisme qui permette de choisir un service Web en se basant sur sa qualité ;
- La norme manque aussi d'une sémantique compréhensible et interprétable par des machines capables de sélectionner le meilleur service Web disponible ;

Pour régler ces problèmes, nous avons proposé une architecture distribuée pour la découverte sémantique des services Web à base d'agent mobile, métadonnées et profil utilisateur. Notre architecture est principalement élaborée autour de la distribution d'annuaires sémantiques des services Web. L'utilisation de plusieurs annuaires dans un processus de découverte offre principalement l'avantage de dépasser le problème du "single point of failure" et permet d'éviter les goulots d'étranglements comme lors de l'utilisation d'un annuaire unique centralisé. Elle s'articule autour de trois principales couches en interaction : couche demandeur des services Web, couche registre des services Web, couche fournisseur des services Web (voir figure V.3).

L'annuaire sémantique proposé est basé sur l'automatisation des phases de publication et de découverte pour améliorer les performances et réduire au maximum l'intervention d'utilisateur dans la découverte. Les informations sur un service publié dans notre annuaire sémantique se présentent sous trois catégories:

- ✓ Une description par WSDL pour décrire l'aspect fonctionnel du service Web ;
- ✓ Une description par le catalogue de métadonnées pour décrire l'aspect sémantique du service Web ;
- ✓ Une description concernant la qualité de service Web (QoS) ;

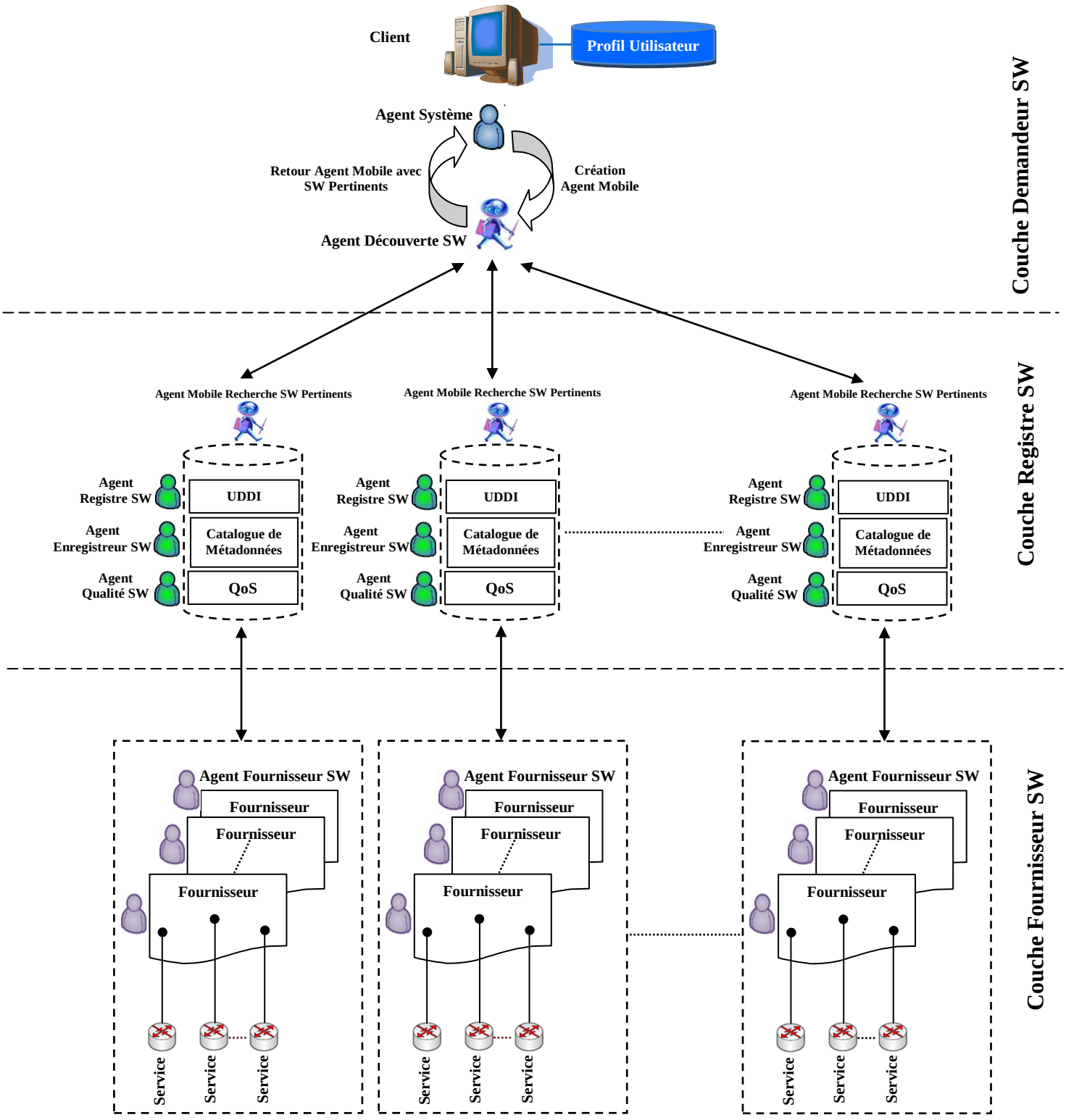

Figure V.3 : Architecture du Framework proposé

Notre travail met en évidence le manque de prise en compte de l'utilisateur et ses préférences lors du processus de découverte de services Web dans l'architecture SOA.

Notre but de recherche est de faire en sorte que l'utilisateur puisse avoir comme résultat un service Web qui répond à son besoin. En d'autres termes, le service retourné par le système proposé doit pouvoir répondre à l'objectif de l'utilisateur tout en étant en adéquation avec les diverses caractéristiques qui constituent son profil.

Dans l'architecture proposée, la phase de découverte passe par cinq étapes (niveaux) comme le montre la figure V.4. Chaque étape utilise le résultat du l'étape précédente pour diminuer le nombre des services Web candidats.

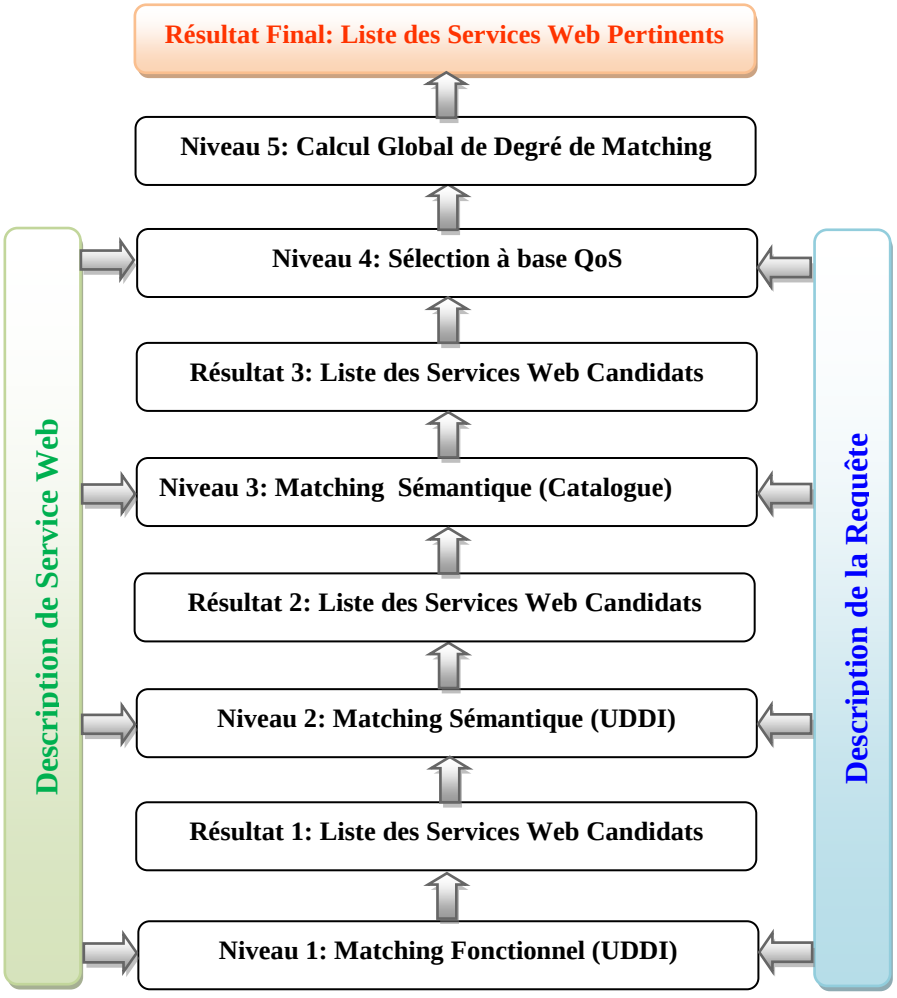

Figure V.4: Phases de découverte sémantique des services Web (Benseghir, 2015).

Nous expliquons brièvement ce qui se passe dans chaque niveau comme suit:

- **Dans le niveau 1 :** nous utilisons l'algorithme de matching basé sur le mécanisme d'interrogation du registre UDDI pour trouver les mises en correspondance entre les fonctionnalités des services Web et celle de la requête, en se basant sur l'aspect syntaxique en utilisant l'interrogation des différentes pages (blanches, jaunes et vertes) basée sur le tModel.
- **Dans le niveau 2 :** nous appliquons l'algorithme de matching (Chabeb, 2011) au niveau UDDI. Il est utilisé pour trouver la similarité sémantique entre les paramètres d'entrée (inputs) et les paramètres de sortie (outputs) de service Web présentés dans la demande de l'utilisateur et ceux de l'annonce (WSDL). Le résultat de la phase précédente sera exploité dans cette étape pour restreindre l'espace de recherche.
- **Dans le niveau 3 :** nous appliquons aussi l'algorithme de matching (Chabeb, 2011) au niveau de catalogue de métadonnées. Il est utilisé pour trouver la correspondance sémantique entre les caractéristiques des services Web présentés dans la demande de l'utilisateur et ceux stockés dans le catalogue de métadonnées des services Web. Le résultat de la phase précédente sera exploité dans cette étape pour restreindre l'espace de recherche.
- **Dans le niveau 4 :** nous calculons le score QoS de chaque service Web candidat issu de la phase précédente.
- **Dans le niveau 5 :** nous appliquons un algorithme efficace pour calculer de manière pertinente le degré global de matching pour la découverte.

V.4. Architecture et fonctionnement des composants

Notre travail propose l'intégration de la technologie d'agent mobile avec les services Web sémantiques. Notre intérêt pour les agents mobiles est dû à plusieurs raisons :

- La mobilité d'agent lui permet de déplacer la charge de calcul d'un site à un autre, afin d'essayer d'équilibrer l'utilisation des ressources dédiées à l'application sur les différents sites participant.
- La mobilité d'agent permet à un client d'interagir localement avec un serveur, et donc de réduire le trafic sur le réseau en ne transmettant que les données utiles.
- L'exécution d'agents spécialisés offre d'avantage de souplesse que l'exécution d'une procédure standard sur les sites serveurs, et permet des transactions plus robustes que les transactions distantes.

- L'asynchronisme et l'autonomie des agents leur permettent de réaliser une tâche tout en étant déconnecté du client, ce qui est particulièrement utile dans le cas de supports physiquement mobiles
- Les agents sont capables de chercher l'information d'une façon plus intelligente, par exemple en cherchant selon des concepts.
- Les agents s'adaptent aux préférences et aux souhaits de chaque usager. Ils peuvent ainsi apprendre de leurs recherches précédentes et par la suite comprendre mieux les besoins des utilisateurs.
- Les agents sont capables de communiquer et coopérer entre eux, ce qui accélère et facilite la recherche.

Dans cette section nous allons présenter l'architecture détaillée de notre Framework en indiquant l'aspect architectural et comportemental de chaque composant.

V.4.1. Agent système (AS)

C'est un agent important dans notre système, il est chargé de gérer le dialogue entre l'utilisateur et notre Framework. Il réalise les tâches suivantes :

- Assurer la bonne communication entre l'utilisateur et le système.
- Assurer l'inscription des nouveaux utilisateurs et l'identification des utilisateurs déjà inscrits.
- Construction du profil en se basant sur les informations fournies directement par l'utilisateur
- Construction de la requête utilisateur.
- Enrichissement de la requête utilisateur en fonction des caractéristiques de l'utilisateur en utilisant les connaissances de son profil
- Génération des agents mobiles pour la découverte des services Web.
- Renvoyer les résultats reçus (liste triée des services pertinents) à l'utilisateur comme une réponse à sa requête
- Mise à jour de profil utilisateur.

La structure de cet agent contient les composants suivants:

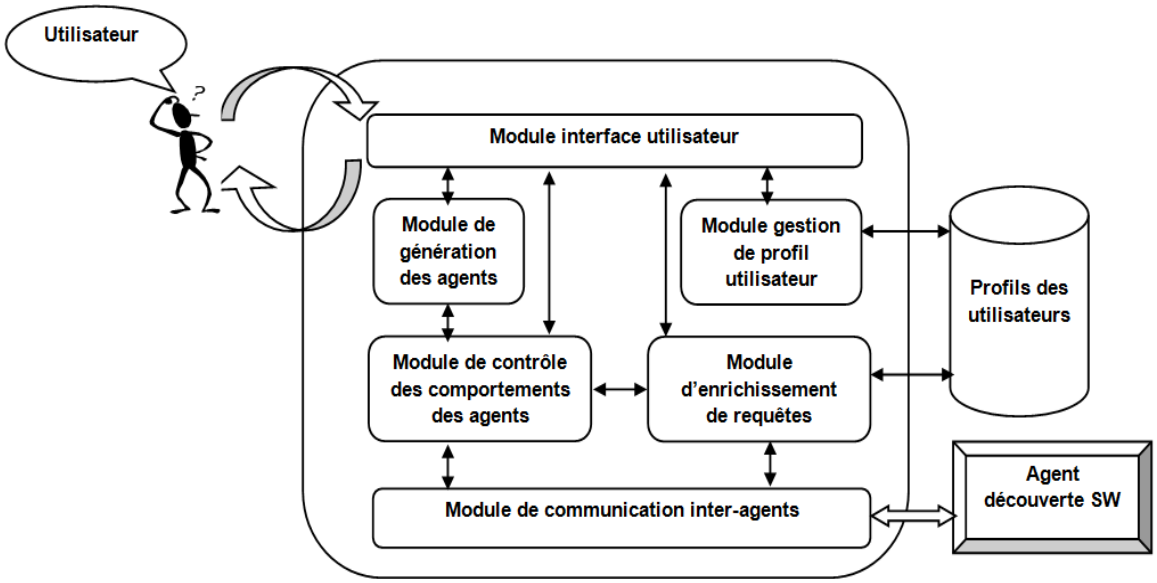

Figure V.5: Architecture interne de l'agent système

- **Module interface utilisateur :** ce module assure l'inscription et l'identification des utilisateurs. Il reçoit les requêtes du client et les transfère au module d'enrichissement de requêtes. En outre, le module interface utilisateur peut recevoir des résultats des autres modules afin de les présenter à l'utilisateur.
- **Module d'enrichissement de requêtes :** l'enrichissement d'une requête exploite le profil de l'utilisateur pour reformuler sa requête en y intégrant des éléments de son centre d'intérêt ou de ses préférences. Cette technique d'enrichissement, courante dans les langages à mots clés en recherche d'information, est très récente en bases de données. Nous avons adapté la méthode d'enrichissement définie dans (Koutrika, 2004) (Koutrika, 2005) pour la découverte des services Web. Dans cette approche, le profil de l'utilisateur est composé d'un ensemble de prédicats pondérés. Le poids d'un prédicat exprime son intérêt relatif pour l'utilisateur. Il est spécifié par un nombre réel compris entre 0 et 1. Ces prédicats sont utilisés pour enrichir la requête utilisateur. Le processus d'enrichissement comporte deux phases principales : (i) sélection des prédicats qui vont enrichir la requête ; (ii) intégration de ces prédicats à la requête.
- **Module gestion de profil utilisateur :** ce module permet la construction et la mise à jour de profil utilisateur en se basant sur les informations fournies directement par l'utilisateur.

- **Module de génération des agents :** pour chaque requête l'agent système va générer un ensemble d'agents mobiles pour êtres déplacés vers le site serveur cible afin de trouver les résultats adéquats. Lors de leur création, il faut définir l'endroit où l'on souhaite qu'ils démarrent leurs activités. Un nom globalement unique (identificateur) doit être attribué à l'agent ce qui va permettre de localiser l'agent. L'agent système peut détruit l'agent mobile quand il termine son travail.
- **Module de contrôle des comportements des agents :** à travers ce module l'agent système peut contrôler tous les agents créés. Un système à agents mobiles doit offrir des mécanismes permettant une exécution sécurisée des agents. Pour augmenter le niveau de sécurité de notre système, l'agent système doit :
  - ✓ Garantir les critères de confidentialité et d'intégrité de l'ensemble des éléments transportés et qui pouvant être cible d'attaque comme : (le code, l'état courant, les données collectées).
  - ✓ Restaurer (régénérer) les agents en cas de défaillance. Plusieurs types de défaillances sont à considérer :
    - Une défaillance par arrêt (crash, panne).
    - Une défaillance temporelle quand la réponse du serveur n'est pas arrivée dans un intervalle de temps donné.
    - Une défaillance par disparition d'un agent après avoir visité le site distant à cause de la déconnexion soudaine et imprévue d'un site sur lequel il s'exécute par exemple.
- **Module de communication inter-agents:** l'agent système communique à travers ce module avec agent découverte SW (agent mobile) en lui envoyant les informations nécessaires au processus de découverte (requête).

V.4.2. Agent fournisseur SW (AFSW)

Cet agent sert d'interface entre le système et le fournisseur du service Web. L'architecture interne de l'agent fournisseur SW est composée de trois modules, comme indiquée en figure V.6. Les trois modules sont comme suit :

- **Module interface fournisseur:** ce module reçoit la description complète des services Web pour l'enregistrement dans : (i) registre UDDI, (ii) catalogue de métadonnées, (iii) le fichier de QoS, et puis les transfère au module de traitement. L'opération inverse est

également disponible, c'est-à-dire que le module peut recevoir des résultats du module de traitement pour les montrer au fournisseur.

- **Module de traitement:** il reçoit des données du module interface fournisseur et fait les contrôles de cohérence de la description des services Web. Il vérifie aussi, pendant chaque période de temps, les changements, dans la qualité ou les fonctionnalités des services enregistrés.
- **Module de communication inter-agents:** il reçoit du module de traitement les informations nécessaires à la procédure de publication, il les transfère également à l'agent registre SW. Dans le cas de l'invocation de services Web l'agent fournisseur SW communique avec l'agent découverte SW.

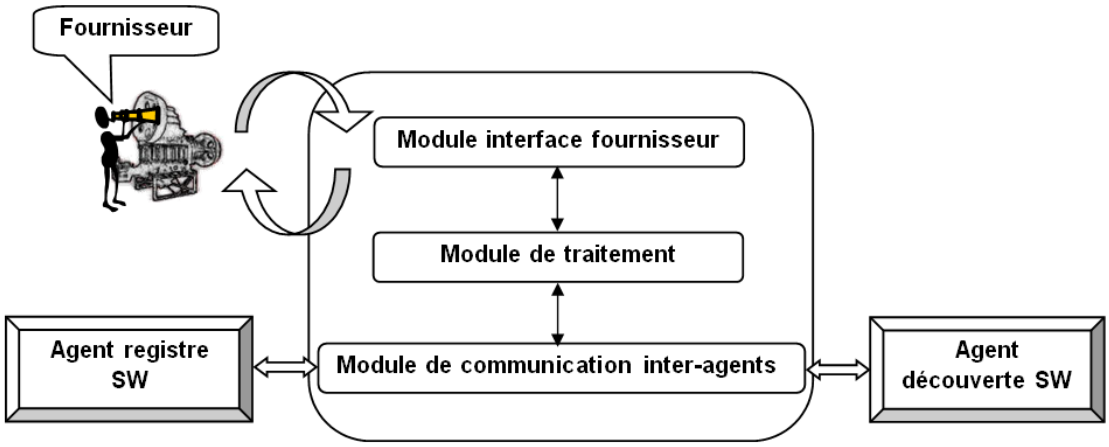

Figure V.6: Architecture interne de l'agent fournisseur SW

V.4.3. Agent registre SW (ARSW)

C'est un agent stationnaire qui agit comme un intermédiaire: entre l'agent fournisseur SW et le registre des services UDDI. Il est composé des modules suivants :

- **Module de traitement :** ce module insère le service Web dans le registre UDDI (publication du service Web) et retourne l'identifiant du service publié (ServiceID)
- **Module de communication inter-agents :** il reçoit du module de traitement la description sémantique de service Web et son identifiant unique (ServiceID) pour les transfère aux agents : agent enregistreur SW et agent qualité SW. En outre, il communique les résultats obtenus (confirmation de la publication du service) avec l'agent fournisseur SW.

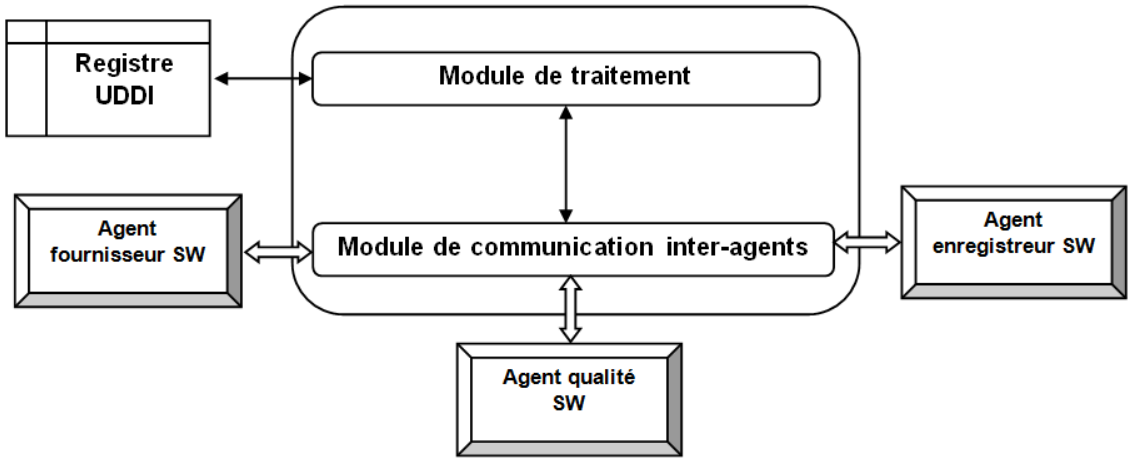

Figure V.7: Architecture interne de l'agent registre SW

V.4.4. Agent découverte SW (ADSW)

Les trois premiers agents avec l'agent enregistreur SW et l'agent qualité SW sont des agents stables, ils fonctionnent continuellement et forment la fondation du notre système. L'agent découverte SW est créé automatiquement en fonction de l'état du système à chaque moment, c'est un agent mobile, qui représente l'utilisateur dans le réseau.

Cet agent est capable de se déplacer à travers un réseau, d'un nœud à un autre pour agir suite à la demande de l'agent système afin de découvrir sémantiquement des services Web satisfaisant le besoin d'utilisateur, il peut également engendrer des clones qui exécutent les services Web sélectionnés en parallèle pour réduire le temps de traitement total. Les clones peuvent migrer et invoquer simultanément les services Web choisis et retourner avec les résultats.

Sa structure contient les composants suivants:

- **Module de communication inter-agents :** comme ses équivalents dans les autres agents, ce module est responsable du traitement des messages entrant et de l'élaboration des messages sortants. A travers ce module, l'agent découverte SW communique avec les autres agents : agent enregistreur SW, agent qualité SW et agent système.
- **Module gestion de migration et de clonage:** ce module spécifie le comportement autonome de l'agent mobile, il permet de gérer la gestion de clonage et de migration d'un agent en cours d'exécution d'un site à un autre à travers le réseau.

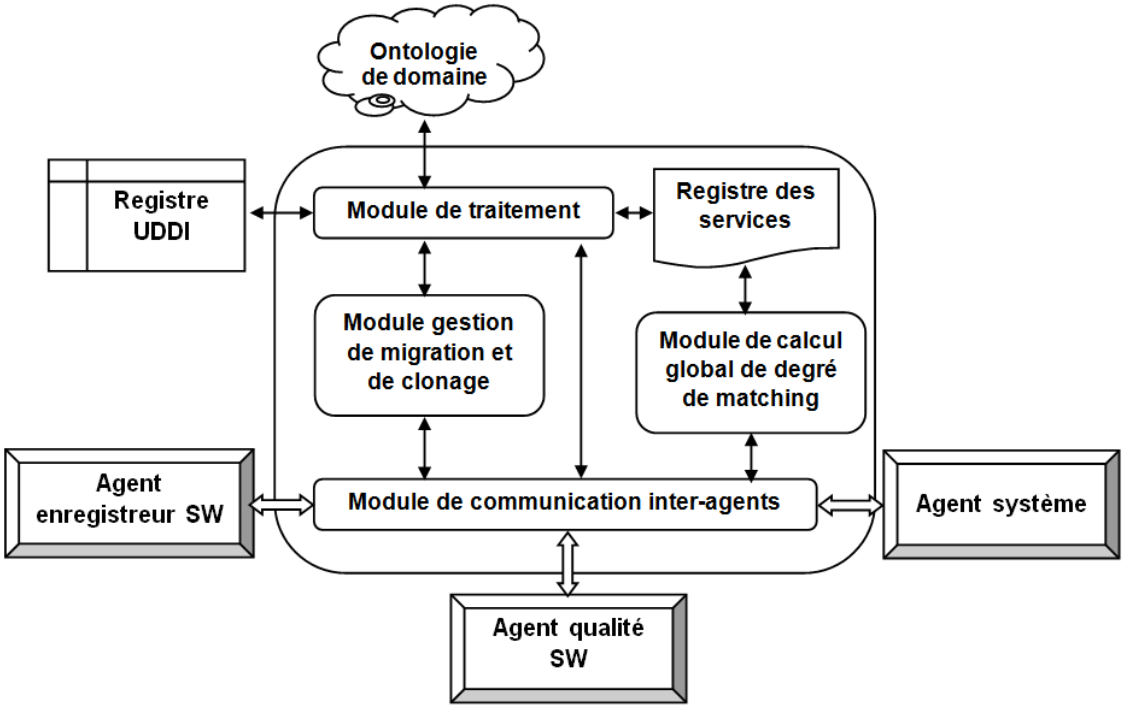

Figure V.8: Architecture interne de l'agent découverte SW

- **Registre des services:** il est utilisé pour stocker les descriptions des services Web satisfaisant la requête de l'utilisateur (liste des services trouvés). Ce registre sera initialisé à chaque requête reçue.
- **Module de traitement :** il applique l'algorithme de matching basée sur le mécanisme d'interrogation du registre UDDI: trouver les mises en correspondance entre les fonctionnalités des services Web et celle de la requête. Le registre UDDI fournit une API d'interrogation de son registre. Elle permet une recherche par mots clés, par tModel (en utilisant par exemple le nom du service ou le nom du fournisseur, …). Ensuite, l'agent découverte SW applique un autre algorithme de matching (Chabeb, 2011) au niveau UDDI pour trouver la similarité sémantique entre les paramètres d'entrée (inputs) et les paramètres de sortie (outputs) de service Web présentés dans la demande de l'utilisateur et ceux de l'annonce (WSDL). Le résultat de la phase de découverte précédente sera exploité dans cette étape pour restreindre l'espace de recherche. La comparaison entre les caractéristiques de service Web et celles présentées dans la requête utilisateur est basée sur l'ontologie du domaine, elle peut avoir six modes de résultat à savoir :

- **Exact**: si le concept offre et le concept demande sont les mêmes (ou équivalents) ;
- **Subsumes**: si le concept offre est un sous concept du concept demande;
- **Subsumes-by:** si le concept demande est un sous concept du concept offre ;
- **Has-same-class**: si les concepts (offre et demande) sont sous concepts du même concept ;
- **Unclassified**: si au moins un des deux concepts n'est pas spécifié ;
- **Fail**: si aucune relation ne peut être déterminée entre les deux concepts.

L'algorithme de comparaison utilise la fonction Englobe donnée ci-dessous: Englobe (O, D) retourne vrai si l'offre O subsume (englobe) la demande D et faux sinon.

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

Figure V.9 : Fonction Englobe (Bouzguenda, 2006)

Ensuite, nous attribuons un score pour chaque mode de matching en appliquant la fonction GetScore.

```
Fonction GetScore (O, D : chaîne) : Entier
% Cette Procédure retourne résultat de comparaison
% D, O sont les éléments de la demande et de l'offre respectivement
Début
   Si O = D Alors Return 5
   Sinon Si Englobe(D, O) Alors Return 4
         Sinon Si Englobe(O, D) Alors Return 3
               Sinon Si Has-same-class(O, D) Alors Return 2
                     Sinon Si Unclassified (O, D) Alors Return 1
                           Sinon Return 0
                           Fin Si
                     Fin Si
               Fin Si
         Fin SI
   Fin Si
Fin
```

Figure V.10 : Fonction d'affection de valeur de matching (Benseghir, 2015)

La valeur globale du matching entre l'offre O et la demande D égale à la somme des valeurs de matching des paires des concepts de l'offre et la demande.

$$semantic\_similarity = \sum_{i=1}^{m} GetScore\left(InputsD_i, InputsO_i\right) + \sum_{i=1}^{m} GetScore\left(OutputsD_i, OutputsO_i\right)$$

Il faut normaliser la similarité sémantique calculée pour chaque service afin d'obtenir des valeurs entre 0 et 1. La normalisation se fait selon la formule suivante:

$$semantic\_similarity\_normalis\acute{e}_{ij} = \frac{semantic\_similarity_{ij} - semantic\_similarity_j^{min}}{semantic\_similarity_j^{max} - semantic\_similarity_j^{min}}$$

On peut décrit le comportement de ce module par l'algorithme suivant :

Algorithme de découverte sémantique de l'agent découverte SW

Inputs: $service = \{service_1, ..., service_n\}$, *requête*

Outputs: *resultat*

$semantic\_similarity$: ***Integer***; $semantic\_similarity\_normalisé$: ***Float;*** $resultat = \phi$;

Begin

Foreach $(service_j)$ ***do***

$$semantic\_similarity_j = \sum_{i=1}^{m} GetScore\left(Inputs_{requête_i}, Inputs_{service_i}\right) + \sum_{i=1}^{m} GetScore\left(Outputs_{requête_i}, Outputs_{service_i}\right);$$

Endfor

Foreach $(service_j)$ ***do***

If $\left(semantic\_similarity\_normalisé_j \neq 0\right)$ ***then***

$resultat = resultat + \{service_j\}$;

Endif

Endfor

End.

Figure V.11 : Le comportement de découverte sémantique de l'agent découverte SW (Benseghir, 2015)

- **Module de calcul global de degré de matching:** ce module permet d'appliquer un algorithme efficace pour calculer de manière pertinente le degré final d'appariement pour la découverte. On peut décrit le comportement de ce module par l'algorithme suivant :

Algorithme de calcul global de degré de matching

Inputs: $service = \{service_1, ..., service_n\}, threshold$

Outputs: *resultat*

$resultat = \phi$; $semantic\_similarity\_global$: *float*;

Begin

Foreach $(service_j)$ ***do***

$$semantic\_similarity\_global_j = \sum semantic\_similarity\_UDDI_j + semantic\_similarity\_Metadata_j + score\_QoS_j;$$

If $\left(semantic\_similarity\_global_j \succ threshold\right)$ ***then***

$$resultat = resultat + \{service_j\};$$

Endif

Endfor

End.

Figure V.12 : Algorithme de calcul global de degré de matching (Benseghir, 2015)

V.4.5. Agent enregistreur SW (AESW)

C'est un agent stationnaire qui agit comme un intermédiaire: entre l'agent découverte SW et le catalogue de métadonnées des services Web (cas de découverte), et aussi entre l'agent fournisseur SW et le catalogue de métadonnées (cas de publication). Il est composé des modules suivants (voir figure V.14):

- **Module de traitement :** Dans le cas de publication, le rôle de cet agent est l'insertion de métadonnée représentant le service Web à publier dans le catalogue des métadonnées, il fait aussi la mis à jour de ce catalogue. Dans le cas de la découverte, Il reçoit la liste ordonnée des services découverts sémantiquement (le résultat de la phase de découverte précédente au niveau UDDI), il applique l'algorithme de matching (Chabeb, 2011) au niveau métadonnées. Cet algorithme sert à trouver les mises en correspondance entre les critères des services Web et celles de la requête utilisateur. Cette mise en correspondance est assurée par la comparaison entre les caractéristiques des services Web présentés dans la requête utilisateur et celles stockées dans le catalogue des métadonnées. La comparaison est basée sur l'ontologie du domaine. On peut décrit le comportement de découverte de l'agent enregistreur SW par l'algorithme suivant :

Algorithme de découverte de l'agent enregistreur SW

***Inputs**: service* $= \{service_1, ..., service_n\}$, *requête*

***Outputs**: resultat*

semantic _ similarity : ***Integer**; semantic _ similarity _ normalisé :* ***Float;*** *resultat* $= \phi$;

Begin

Foreach *(service*$_j$*)* ***do***

$$semantic\_similarity_j = \sum_{i=1}^{m} GetScore(Critère_{requête_i}, Critère_{service_i});$$

Endfor

Foreach *(service*$_j$*)* ***do***

If $(semantic\_similarity\_normalisé_j \neq 0)$ ***then***

$resultat = resultat + \{service_j\}$;

Endif

Endfor

End.

Figure V.13 : Le comportement de découverte de l'agent enregistreur SW (Benseghir, 2015)

- **Module de communication inter-agents :** il communique les résultats obtenus : (i) confirmation de la publication du service avec l'agent registre SW ; (ii) la liste des services Web pertinents (serviceID + semantic_similarity_metadata) avec l'agent qualité SW.
- **Registre des services:** tous les services pertinents seront stockés dans ce registre, ce dernier sera initialisé à chaque requête.

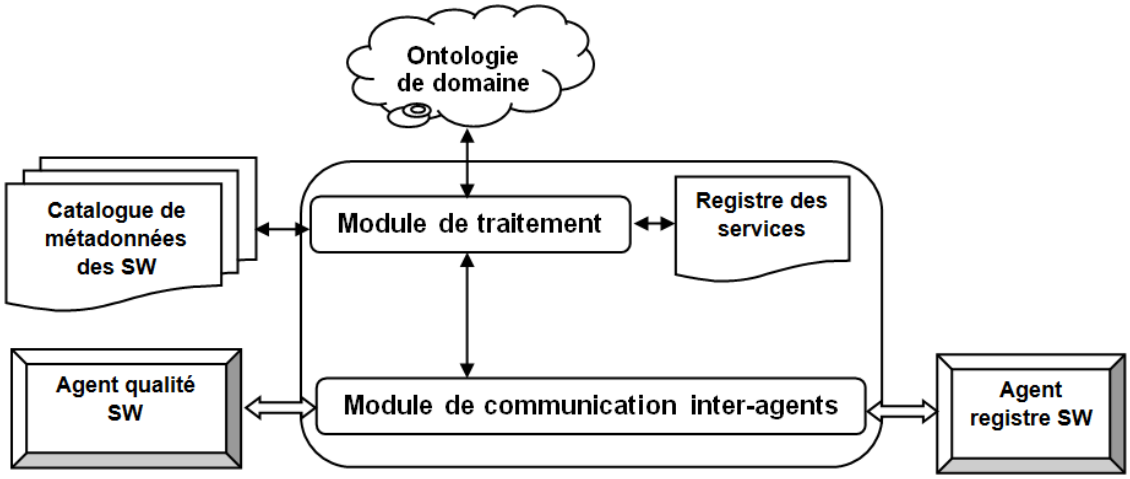

Figure V.14: Architecture interne de l'agent enregistreur SW

V.4.6. Agent qualité SW (AQSW)

C'est un agent stationnaire qui agit comme un intermédiaire: entre l'agent découverte SW et le fichier de QoS (cas de découverte), et aussi entre l'agent fournisseur SW et le fichier de QoS (cas de publication). Il est composé des modules suivants (voir figure V.16):

- **Module de traitement :** Dans le cas de publication, le rôle de cet agent est l'insertion des informations de qualité de service Web dans le fichier QoS, il fait aussi la mis à jour de ce fichier. Dans le cas de la découverte, le rôle de l'agent qualité SW repose essentiellement sur le calcul de score de la qualité des services Web par application de l'algorithme illustré dans la figure V.15. L'ensemble A représente l'ensemble des Attributs de QoS réclamés par l'utilisateur, tel que pour chaque attribut $Qos_i (1 \leq i \leq m)$ un poids (W_i) lui est assigné. SWS est l'ensemble des m services Web résultats de l'étape précédente. $f(sws_i)$ représente la fonction qui calcul le score pour chaque service Web de l'ensemble SWS. V_{ij} représente la valeur QoS pour chaque service Web de l'ensemble SWS. Nous prenons en compte que chaque valeur QoS est un critère positif (ex : fiabilité, disponibilité) ou négatif (ex : le prix d'exécution, temps de réponse).

Algorithme 2 : Sélection basée QoS des services Web sémantiques

Inputs: $A = \{ Qos_1, ..., Qos_n \}$, $SWS = \{ sws_1, ..., sws_m \}$, $W = \{ w_1, ..., w_n \}$,
Outputs: *resultat*
Begin
$resultat = \phi$; $j = 1$;
While $(j \neq m)$ **do**
$f(sws_j) = \sum_{i=1}^{n} V_{ij} \times W_i$;
$resultat = resultat \cup \{ sws_j \}$;
$j = j + 1$;
Endwhile
End.

Figure V.15 : Algorithme de sélection à base QoS (Benseghir, 2015)

- **Module de communication inter-agents :** il communique les résultats obtenus : (i) confirmation de la publication du service avec l'agent registre SW ; (ii) la liste des services Web pertinents (ServiceID + semantic_similarity_metadata + score_QoS) avec l'agent découverte SW.
- **Registre des services:** tous les services pertinents seront stockés dans ce registre, ce dernier sera initialisé à chaque requête.

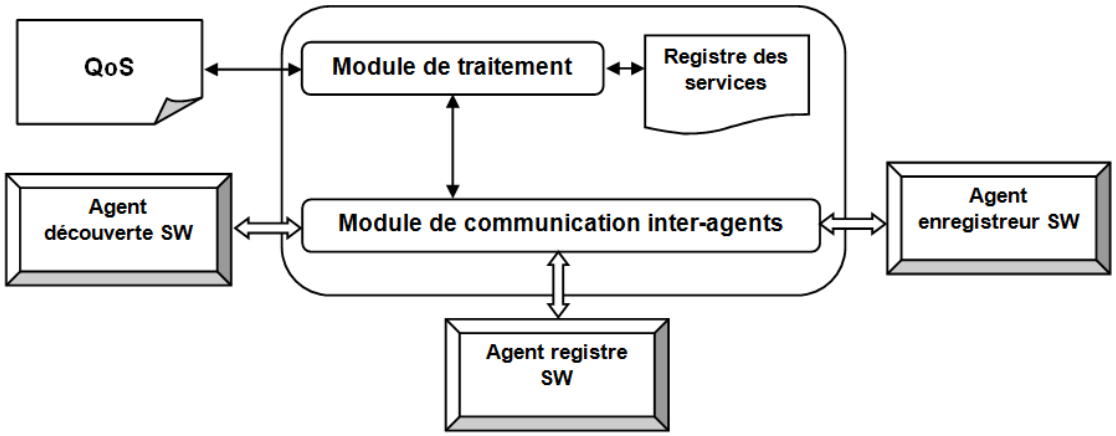

Figure V.16: Architecture interne de l'agent qualité SW

V.5. Scenario de fonctionnement du système

Dans cette section nous présentons les différents scénarios de fonctionnement de notre système. Ils aident à la compréhension et l'explication de processus de découverte des services Web, et le processus de publication des services Web qui sont considérés comme les deux processus les plus importants dans notre système.

V.5.1. Publication des services Web

Dans l'architecture proposée, un fournisseur de service Web assure la publication de son service Web à travers notre annuaire sémantique :(i) UDDI en utilisant le WSDL ; (ii) Catalogue de métadonnées des services Web ; (iii) Critères de QoS. La phase de publication est effectuée à travers les opérations suivantes:

1. Le fournisseur faire une demande de publication, cette demande est assurée par une interface graphique assurant la saisie de la description complète des services Web (UDDI, Catalogue de métadonnées, QoS) ;
2. L'agent fournisseur SW génère et envoie la description complète des services Web vers l'agent registre SW;
3. l'agent registre SW faire l'enregistrement de la description du SW (partie fonctionnelle) dans le registre UDDI. Elle concerne l'enregistrement d'un nouveau service ou bien la modification d'un service existant. Pour chaque WSDL publié, l'agent registre SW lui associe une chaîne de caractères unique, appelée identifiant de service : ServiceID ;
4. L'agent registre SW faire le transfert de la description du SW (QoS) et son identifiant unique (ServiceID) vers l'agent qualité SW ;
5. L'agent registre SW faire le transfert de la description du SW (métadonnées des SW) et son identifiant unique (ServiceID) vers l'agent enregistreur SW ;
6. L'agent enregistreur SW prend en charge de la mise à jour du catalogue des métadonnées en effectuant soit une nouvelle saisie, soit une modification ;
7. L'agent qualité SW prend en charge de la mise à jour du fichier QoS en effectuant soit une nouvelle saisie, soit une modification ;
8. Confirmation de l'enregistrement dans le catalogue de métadonnées : une fois l'action est exécuté un message de confirmation de l'action réalisée sera transmis à l'agent registre SW de l'agent enregistreur SW ;
9. Confirmation de l'enregistrement dans le fichier QoS : une fois l'action est exécuté un message de confirmation de l'action réalisée sera transmis à l'agent registre SW de l'agent qualité SW ;
10. Confirmation de l'enregistrement dans le registre UDDI : l'agent registre SW reçoit la confirmation de l'agent enregistreur SW et l'agent qualité SW qui sera transmise avec la confirmation de l'action réalisée au niveau UDDI à l'agent fournisseur SW ;
11. Confirmation de la publication : l'agent fournisseur SW envoie la confirmation de la publication au fournisseur ;

La figure suivante représente le diagramme de séquences du protocole de publication des services Web dans l'architecture du Framework que nous avons proposé.

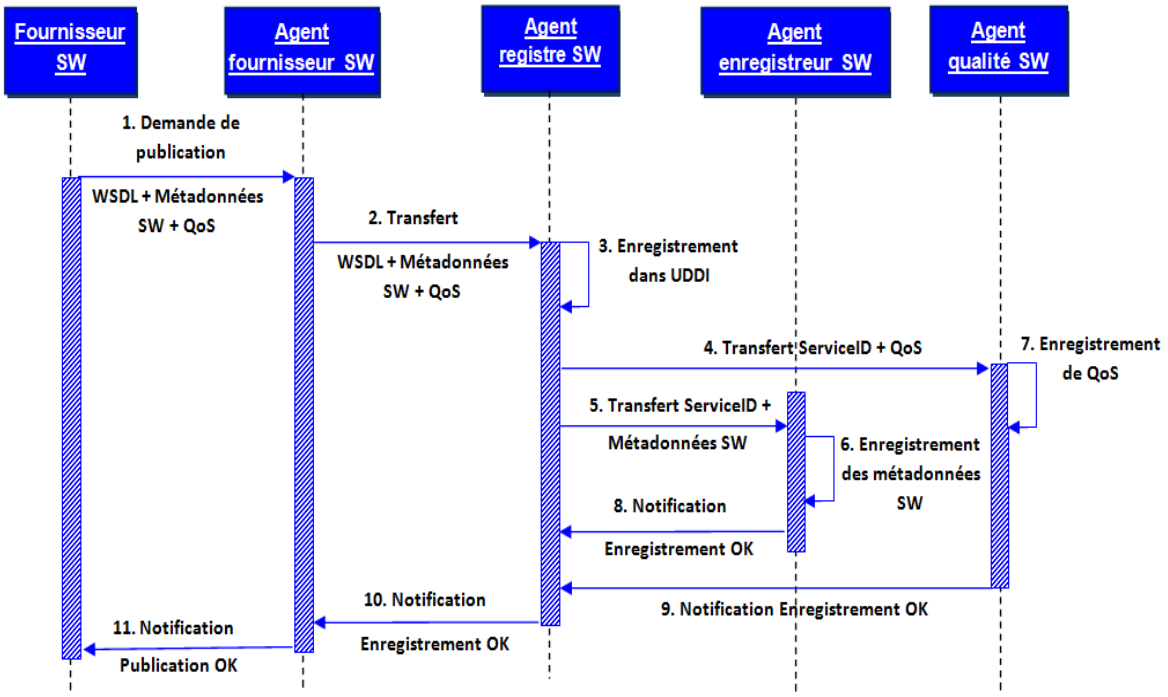

Figure V.17 : Diagramme de séquences de protocole de publication des services Web.

V.5.2. Découverte des services Web

Le client du service présente sa requête suivant un profil d'utilisateur permettant d'optimiser sa demande. Le service demandé sera découvert suivant le diagramme de séquences du protocole de découverte du SW présenté dans la figure V.18. La phase de découverte est effectuée à travers les opérations suivantes :

1. Demande de service : le client présente sa requête à l'agent système selon une interface graphique;
2. Enrichissement de la requête : l'agent système adapte la requête selon le profil utilisateur;
3. Génération des agents mobiles pour la découverte: pour chaque requête reçue l'agent système va générer et puis activer des agents découverte SW ;
4. Transfert de la requête enrichie : l'agent système fait le transfert de requête à l'agent découverte SW ;
5. Migration de l'agent mobile de découverte : après que chaque agent découverte SW avoir sa requête, il va déplacer depuis le site d'origine vers le site cible afin de trouver les services souhaités ;

6. Recherche fonctionnelle au niveau UDDI : l'agent découverte SW applique un matching fonctionnel au niveau UDDI et donne comme résultat une liste non triée des services Web ;

7. Recherche sémantique au niveau UDDI : l'agent découverte SW applique un matching sémantique au niveau UDDI et donne comme résultat une liste triée des services Web ;

8. Transfert de la liste des services trouvés (ServiceID) par l'agent découverte SW et la requête à l'agent enregistreur SW ;

9. Recherche sémantique au niveau catalogue : l'agent enregistreur SW applique un matching sémantique au niveau catalogue des métadonnées des SW ;

10. Transfert de la liste des services trouvés (ServiceID + semantic_similarity_metadata) par l'agent enregistreur SW et la requête à l'agent qualité SW ;

11. Sélection à base QoS : l'agent qualité SW calcule le score QoS de chaque service Web trouvé par l'agent enregistreur SW;

12. Transfert de la liste des services trouvés (ServiceID + semantic_similarity_metadata + score_QoS) par l'agent qualité SW à l'agent découverte SW ;

13. Application d'un algorithme efficace par l'agent découverte SW pour calculer de manière pertinente le degré final d'appariement pour la découverte.

14. Retour de l'agent découverte SW vers le site d'origine avec le résultat trouvé ;

15. Transfert de la liste triée des services pertinents de l'agent découverte SW à l'agent système ;

16. Destruction de l'agent découverte SW par l'agent système ;

17. Présentation des résultats au client.

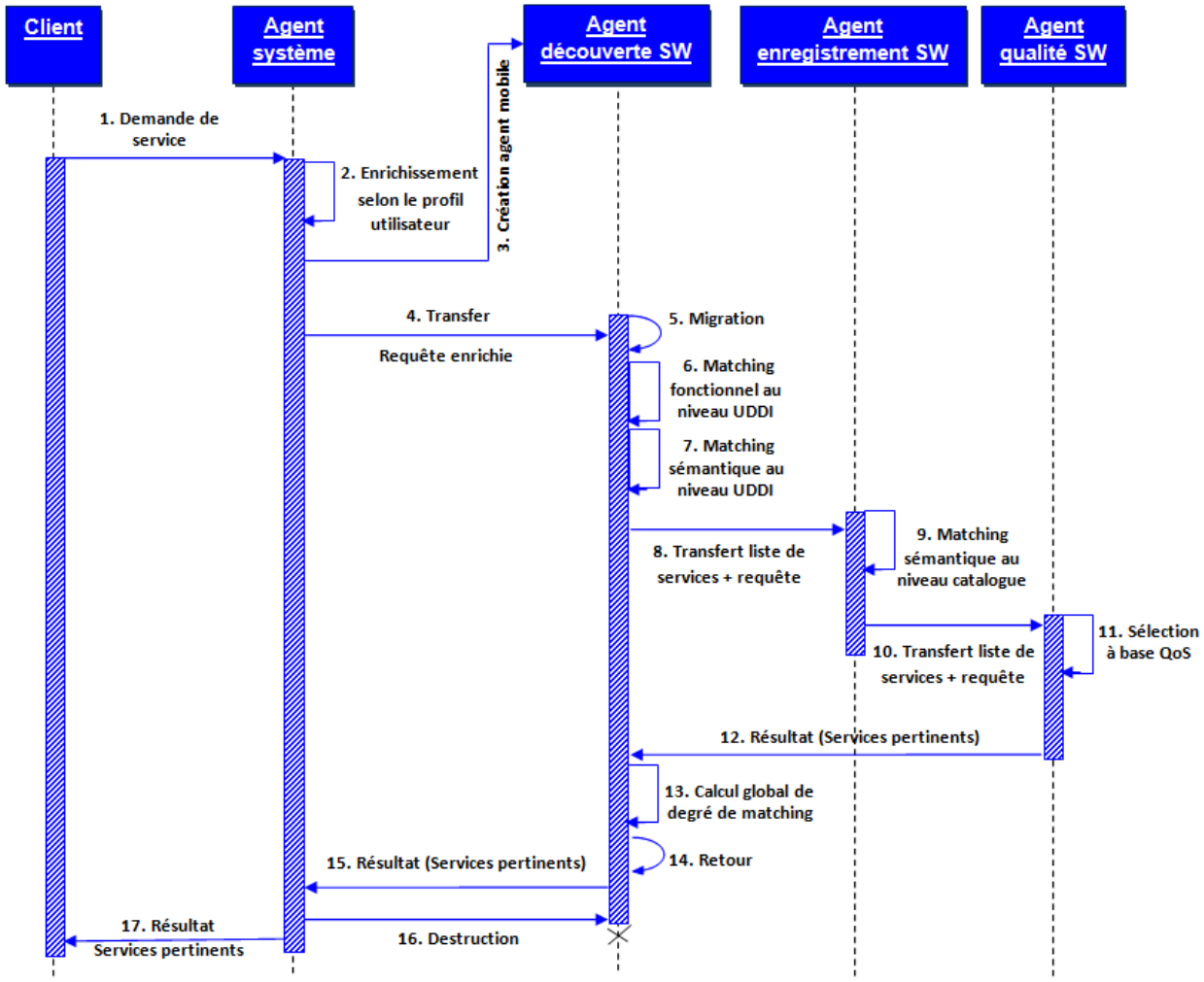

Figure V.18 : Diagramme de séquences de protocole de découverte des services Web

V .6. Etude comparative

Rappelons que la découverte de services Web est le processus de recherche des services qui répondent à certaines exigences spécifiques. Plusieurs approches ont été proposées pour la découverte de service Web, ces approches partagent le but commun de rendre le processus de découverte de service Web plus flexible.

Initialement, la découverte de services Web était basée sur des mots clés. Cependant, avec le développement des technologies du Web sémantique des nouvelles approches basées sur la sémantique des descriptions des services ont été proposées. Le tableau suivant représente une comparaison entre notre approche de découverte et les travaux connexes qui sont présentés dans l'état de l'art.

Critères / Approches	Description des services	Architecture adoptée	Technique de découverte	Technologie adoptée	QoS	Profil utilisateur
UDDI+ (Pokraev, 2003)	*DAML-S*	*Centralisée*	*Ontologies*	*SOAP*	*Non*	*Oui*
IRS-II (Motta, 2003)	*Ontologie de domaine + Ontologie Des PSM*	*Centralisée*	*PSM*	*SOAP*	*Non*	*Non*
OWL-S (Martin, 2004)	*Ontologie OWL*	*Centralisée*	*Algorithme matchmaking*	*SOAP*	*Non*	*Non*
AASDU (Palathingal, 2004)	*WSDL*	*Distribuée*	*TFIDF*	*Multi-agents*	*Non*	*Non*
SPEED-R Sivashanmugam, 2004)	*Mapping entre WSDL et ontologie de domaine*	*Distribuée*	*Mots clés / sémantique*	*Multi-agents sous réseau P2P*	*Non*	*Non*
PSWSD (Vu, 2005)	*Ontologie WSMO+WSDL*	*Distribuée*	*Matchmaker*	*Multi-agents sous réseau P2P*	*Non*	*Non*
ROSSE (Li, 2009)	*WSDL+ Ontologie OWL-S*	*Centralisée*	*Algorithme matchmaking*	*SOAP*	*Non*	*Non*
Approche (Lu, 2012)	*Mapping entre WSDL et ontologie de domaine*	*Centralisée*	*Matchmaker*	*SOAP*	*Non*	*Non*
Approche (Celik, 2013)	*Ontologie OWL-S*	*Distribuée*	*Algorithme de matching sémantique*	*Multi-agents*	*Oui*	*Non*
Approche (Benaboud, 2013)	*OWL-S Ontologie + extension par QoS*	*Distribuée*	*Similarité Syntactique + Similarité sémantique*	*Multi-agents*	*Oui*	*Non*
Approche (Karray, 2013)	*WSDL*	*Distribuée*	*Algorithme de Bees*	*SOAP*	*Oui*	*Non*
Approche (ElBouhissi, 2014)	*Ontologie WSMO*	*Centralisée*	*Algorithme de matching à base WordNet + ontologies de domaine*	*SOAP*	*Non*	*Non*
Notre Approche (Benseghir, 2015)	*Mapping entre WSDL + Métadonnées + QoS*	*Distribuée*	*Matching sémantique + Matching fonctionnel + Sélection à base QoS*	*Multi-agents + technologie d'agent mobile*	*Oui*	*Oui*

Tableau V.2 : Synthèse des approches de découverte des services Web (Benseghir, 2015)

Dans le tableau V.2 on peut remarquer que la technologie multi-agents et plus particulièrement la technologie d'agent mobile est bien adaptée aux applications réparties. On remarque aussi que dans les approches sémantiques proposées, tel que OWL-S (Martin, 2004), Speed-R (Sivashanmugam, 2004), PSWSD (Vu, 2005) et l'approche (Lu, 2012), un processus de découverte de service web est considéré comme un Matchmaker. Mais il est insuffisant car le Matchmaking (Matching) est une seule étape parmi plusieurs étapes qui composent le processus de découverte.

D'autre part, certaines approches tel que (Celik, 2013), (Benaboud, 2013), (Karray, 2013) ont proposé d'intégrer des modèles de découverte de services Web basés sur la qualité de service (QoS) pour permettre de faire des recherches de services plus fines. Cependant, ces approches ne focalisent que sur un seul paramètre : QoS pour lequel le service peut s'adapter. Les mécanismes de découverte doivent prendre en considération le contexte de la découverte de services afin de ne proposer à l'utilisateur que des services qui répondent au mieux à ses besoins. Le contexte de la découverte englobe plusieurs paramètres relatives à *l'utilisateur* et au *service Web* tel que : catégorie du service, qualités de service (QoS), localisation du service, les informations représentant les centres d'intérêts de l'utilisateur, etc.

Notre approche a trois objectifs fondamentaux. Le premier consiste à prendre en compte le profil utilisateur dans le processus de découverte afin de focaliser et de diriger la recherche des services Web. Le deuxième est d'enrichir la description des services Web par l'aspect sémantique en utilisant le catalogue de métadonnées, et les propriétés de QoS qui vise à réduire l'espace de recherche et d'augmenter le nombre de services pertinents. Le troisième est l'utilisation d'agents mobiles comme une entité de communication; Cela permet de réduire le trafic sur le réseau en ne transmettant que les données utiles.

V.7. Conclusion

Dans les chapitres précédents nous avons exposé l'état de l'art relevant du domaine de notre investigation. Ainsi, les modèles et les démarches relatives à ce domaine de recherche ont fait l'objet de descriptions et d'analyses inhérentes à leurs caractéristiques. C'est un domaine large qui connait d'intenses recherches qui portent sur plusieurs aspects, car il se situe à la connexion de plusieurs disciplines comme le Web sémantique, les services Web, et le paradigme agent mobile qui reste toujours d'actualité.

Dans ce chapitre nous avons proposé un nouvel Framework à base de profil utilisateur et métadonnées pour la découverte sémantique des services Web, en s'appuyant sur la capacité de la mobilité des agents. Le Framework proposé est une architecture distribuée composée de trois couches : couche demandeur des services Web, couche registre des services Web, couche fournisseur des services Web.

Donc dans l'objectif de rendre la découverte des services Web plus pertinente, nous avons proposé une phase de publication dont le but est d'enrichir la description des services Web. Cette dernière est effectuée à trois niveaux de description. Le premier niveau concerne le registre UDDI à travers le fichier WSDL, le deuxième niveau concerne le catalogue de métadonnées des services Web, le troisième niveau concerne les critères de qualité de services Web (QoS).

Dans la phase de découverte, cet enrichissement permet de restreindre l'espace de recherche et d'augmenter le nombre de services pertinents à travers l'application de : (1) Matching fonctionnel au niveau UDDI ; (2) Matching sémantique au niveau UDDI ; (3) Matching sémantique au niveau catalogue de métadonnées ; (4) Sélection à base de QoS ; (5) Calcul de degré global d'appariement de découverte.

Pour la mise en œuvre de notre Framework, un prototype de découverte sémantique, à base de qualité, dont le champ d'application est le domaine de tourisme, fait l'objet du prochain chapitre.

Chapitre VI

Etude de Cas et Implémentation

VI.1. Introduction

A travers le cinquième chapitre, nous avons présenté notre Framework à base d'agent mobile pour la découverte sémantique des services Web. Le Framework proposé est une architecture distribuée composée de trois couches : couche demandeur des services Web, couche registre des services Web, couche fournisseur des services Web. Elle est basée sur l'utilisation de : catalogue de métadonnées, critères de qualités des services Web et le profil utilisateur qui sera utilisé pour enrichir et compléter les requêtes de l'utilisateur.

Pour la validation et la réalisation de notre Framework nous avons opté à réaliser un prototype en faisant appel à un ensemble d'outils (Protégé, XML/XML Schemas, Jade, ...), en prenant comme domaine d'application le domaine du tourisme.

Ce chapitre est organisé comme suit. Tout d'abord, nous commençons par la présentation des outils et les plateformes de développement utilisés à l'implémentation des différents composants du prototype. Dans la section 3 nous présentons l'architecture générale de l'application Web proposée, et l'ontologie de domaine exploitée. Dans la section 4, nous exploitons notre réalisation dans un cas d'utilisation. Ensuite, nous présentons les interfaces de notre application et les résultats obtenus dans la section 5. Finalement, nous présentons la conclusion du travail réalisé.

VI.2. Outils et plateformes utilisés

Le prototype a été développé sous le système d'exploitation Windows 7 professionnel avec des outils souvent Open source graphiques et développés en Java. Nous détaillons, dans ce qui suit, chacun des outils et langages utilisés pour la manipulation des données ainsi que l'implémentation des interfaces de notre application.

VI.2.1. Environnement de développement

Afin de faciliter le développement de notre application, nous avons utilisé l'environnement de développement Eclipse Indigo basé sur le langage Java. Eclipse est un Environnement de Développement Intégré (EDI: Environment development integrated). Il est particulièrement bien adapté pour le développement d'applications Web, ainsi il supporte différents langages, comme Python, C, C++, XML, HTML. C'est un IDE moderne qui offre un éditeur avec des codes couleurs et un ensemble de signes, des modèles de projets multi-langage et de différents types (application indépendante, distribuée, plugin, mobiles, ... etc.).

Eclipse est disponible sous Windows, Linux ou sous une version indépendante des systèmes d'exploitation. Un environnement Java Développent Kit JDK est requis pour les développements en Java.

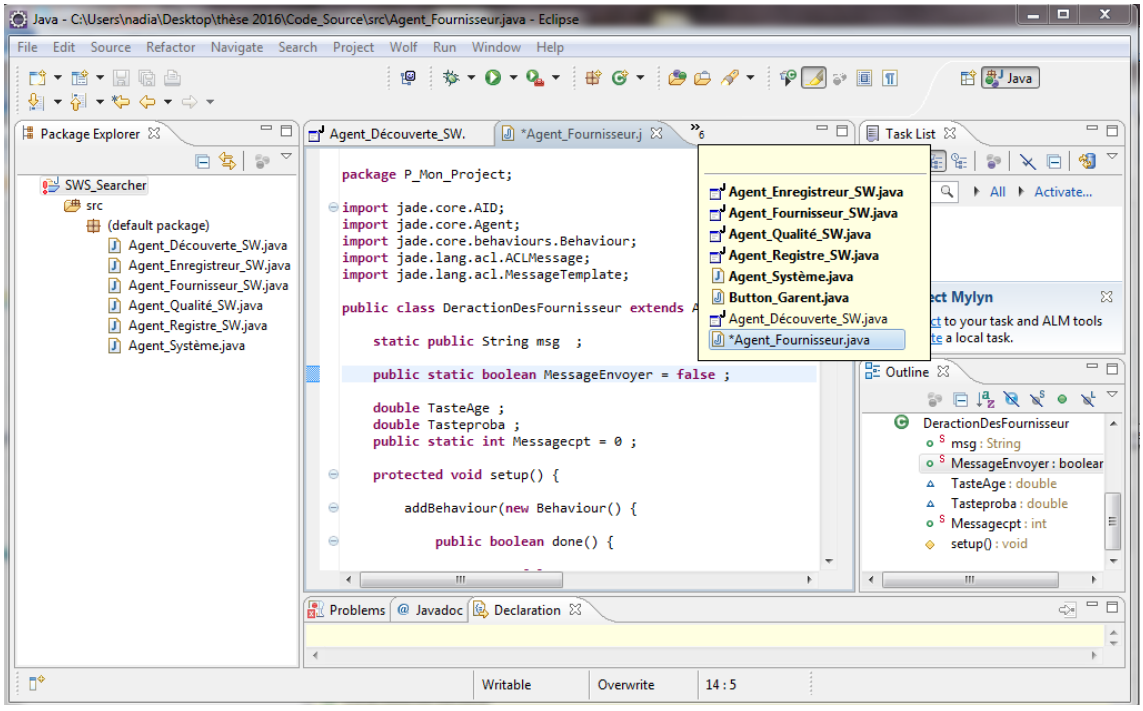

Figure VI.1 : La fenêtre principale de l'Eclipse

VI.2.2. Langage de programmation

Nous avons choisi le langage Java pour la programmation de notre application. La syntaxe générale du langage Java est très proche de celle du langage C. Ce choix a été motivé par les raisons suivantes :

- ✓ Java est un langage orienté objet simple ce qui réduit les risques d'incohérence ;
- ✓ Java est portable. Il peut être utilisé sous Windows, sous Linux, sous Macintosh et sur d'autres plateformes sans aucune modification. Java est donc un langage multiplateformes, ce qui permet aux développeurs d'écrire un code qu'ils peuvent exécuter dans tous les environnements ;
- ✓ Java possède une riche bibliothèque de classes comprenant des fonctions diverses telles que les fonctions standards, le système de gestion de fichiers, les fonctions multimédia et beaucoup d'autres fonctionnalités ;
- ✓ Il existe une API Java fournie avec l'éditeur d'ontologies Protégé, ce qui permet d'accéder à l'ontologie à partir de notre application. De plus, les API des autres langages ne sont pas encore finalisées et doivent encore être mises à jour.

VI.2.3. Plateforme JADE

Afin d'assurer un développement rapide est efficace des agents qui composent notre système, nous utilisons la plateforme de développement des systèmes multi-agent JADE.

JADE est une plateforme implémentée complètement avec le langage Java. Ce Framework est destiné à faciliter le développement des systèmes multi-agents en conformité totale avec le standard FIPA. La plateforme JADE fournit une interopérabilité sans limite pour les applications quelles prend en charge, car cette plateforme est indépendante du système d'exploitation ainsi que du matériel sur laquelle elle est implémenté (Bellifemine, 1999).

VI.2.3.1. Architecture logicielle

Puisque JADE est une plateforme basée sur les standards FIPA, l'architecture de cette plateforme est également basée sur l'architecture proposée par FIPA. AMRM (Agent Management Reference Model) est le modèle de base de l'architecture de la plateforme JADE proposée par FIPA. Chaque module qui compose l'architecture de la plateforme JADE est présenté sous forme de service, ce qui permet aux agents de bénéficier d'une plateforme orientée service, afin de faciliter la communication et la collaboration entre eux. Les principaux modules qui composent l'architecture JADE sont : DF, AMS et également le MTS (Massage Transport Service) qui sert de moyen pour la communication entre plusieurs plateformes JADE. Afin d'assurer un fonctionnement efficace des agents sur la plateforme JADE, cette dernière utilise :

- **AID (Agent Identifier) :** afin de distinguer et d'identifier chaque agent.
- **DF :** qui joue le rôle d'un annuaire servant à enregistrer les compétences de chaque agent. Les pages jaunes fournis par ce service, sont destinées à mettre en relation les différents agents fonctionnels sur la plateforme JADE, et cela pour qu'un agent puisse consulter et interroger ce service, afin d'obtenir des informations sur les compétences des autres agents et afin d'assurer une bonne collaboration entre eux.
- **AMS :** joue le rôle d'un annuaire pour l'enregistrement des adresses de transport des différents agents de la plateforme. Le but c'est de fournir un service de « pages blanches » afin de mettre en correspondance les agents avec l'AID, pour faciliter leur contrôle et supervision.

VI.2.3.2. Langage de communication

Le langage de communication FIPA-ACL (Agent Communication Langage) est le langage adopté par la plateforme JADE. De façon générale, la communication entre agents est en mode asynchrone, et elle est mise en oeuvre en utilisant la classe ACLMessage. La structure générale d'un message envoyé entre deux agents contient plusieurs champs tels que chaque champ comprend une donnée spécifique. La structure d'un message est la suivante :

- **Sender :** contient le nom de celui qui envoie le message.
- **Receiver :** contient la liste des noms des agents qui vont recevoir le message.
- **Performative :** ce champ peut prendre les valeurs suivantes : REQUEST, INFORM, PROPOSE, ACCEPT_PROPOSAL, REJECT_PROPOSAL.
- **Content :** ce champ contient le message à envoyer.

VI.2.3.3. Mobilité

Afin de permettre aux agents de migrer d'une plateforme à une autre ou d'un container à un autre, selon le type de mobilité visé par chaque agent mobile, la plateforme JADE utilise deux types de services:

- **Mobilité Intra-plateforme :** la plateforme JADE utilise un service de gestion de la mobilité des agents (Agent Mobility Service) pour implémenter ce type de mobilité. Avec la mobilité intra-plateforme, l'agent mobile a la faculté d'émigrer d'un container à un autre, mais dans la même plateforme seulement, en utilisant la méthode doMove().

Outre, les méthodes beforeMove() et afterMove() servent comme moyen de déterminer les actions à entreprendre par l'agent mobile avant ou après la migration.

- **Mobilité Inter-plateforme :** la plateforme JADE utilise l'IPMS (Inter-Platform Mobility Service) pour implémenter ce type de mobilité. Avec la mobilité inter-plateforme, l'agent mobile a la faculté d'émigrer d'une plateforme à une autre, en utilisant la méthode move() et power-up(). La première méthode sert à l'émigration de l'agent avec ses données et son code, et la seconde méthode sert à activer l'agent une fois la migration effectuée.

VI.2.4. L'éditeur des ontologies Protégé

Protégé est un éditeur d'ontologies distribué en open source par l'université en informatique médicale de Stanford. Protégé n'est pas un outil spécialement dédié à OWL, mais un éditeur hautement extensible, capable de manipuler des formats très divers. La figure VI.2 illustre notre ontologie de tourisme crée par l'outil Protège 5.0.0-beta-24.

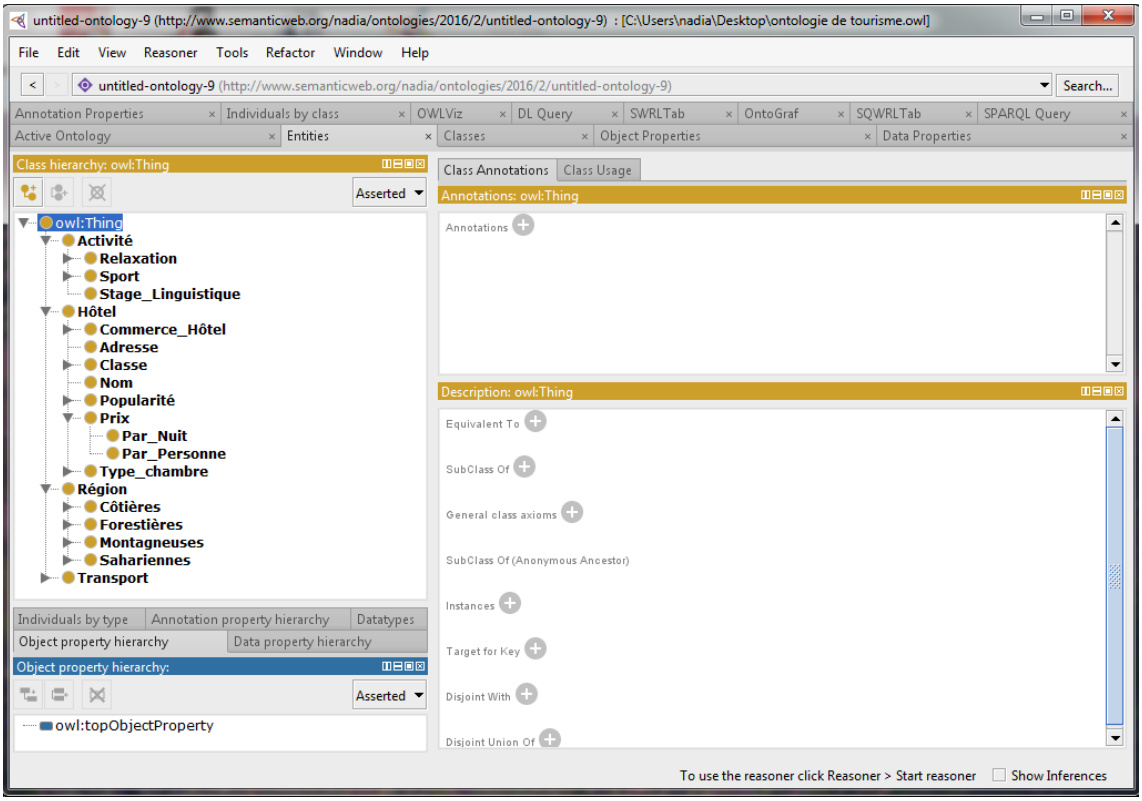

Figure VI.2 : Aperçu de l'ontologie « Tourisme » comme elle apparaît sur l'éditeur Protégé 5.0 (Benseghir, 2015)

Le support d'OWL, comme de nombreux autres formats, est possible dans Protégé grâce à un plugin dédié. Protégé est un outil employé par les développeurs et des experts de domaine pour développer des systèmes basés sur les connaissances (ontologies). Des applications développées avec Protégé sont employées dans la résolution des problèmes et la prise de décision dans un domaine particulier. Protégé est aussi une plate-forme extensible, grâce au système de plug-ins, qui permet de gérer des contenus multimédias, interroger, évaluer et fusionner des ontologies, etc. L'outil Protégé possède une interface utilisateur graphique (GUI) lui permettant de manipuler aisément tous les éléments d'une ontologie : classe, méta-classe, propriété, instance,…etc.

Protégé permet aussi de créer ou d'importer des ontologies écrites dans les différents langages d'ontologies tel que : RDF-Schéma, OWL, DAML, OIL, …etc. Cela est rendu possible grâce à l'utilisation de plugins qui sont disponibles en téléchargement pour la plupart de ces langages.

VI.2.5. JENA

JENA est un Framework Java pour construire des applications Web sémantique. Il fournit un environnement de programmation pour RDF, RDFS, OWL, et SPARQL. JENA comprend un moteur d'inférence à base de règles.

VI.2.6. JSP

Le Java Server Pages ou JSP est une technique basée sur Java qui permet aux développeurs de créer dynamiquement du code HTML, XML ou tout autre type de page Web. Cette technique permet au code Java et à certaines actions prédéfinies d'être ajoutés dans un contenu statique. Depuis la version 2.0 des spécifications, la syntaxe JSP est complètement conforme au standard XML. Les JSP permettent d'écrire facilement des servlets, en incluant dans des balises spécifiques le code JSP au sein du fichier HTML. De cette façon, elles fournissent une technologie rapide afin de créer des pages dynamiques. De plus, les JSP sont basées sur Java côté serveur, elles possèdent toutes les caractéristiques faisant la force de Java : efficace ; facile pour les développeurs du Web ; ...

VI.3. Description générale de l'application

Dans cette section nous présentons l'application qu'on a développée afin de tester le système proposé :

VI.3.1. Architecture d'application Web

La figure VI.3 illustre l'architecture d'application Web que nous avons mise en place :

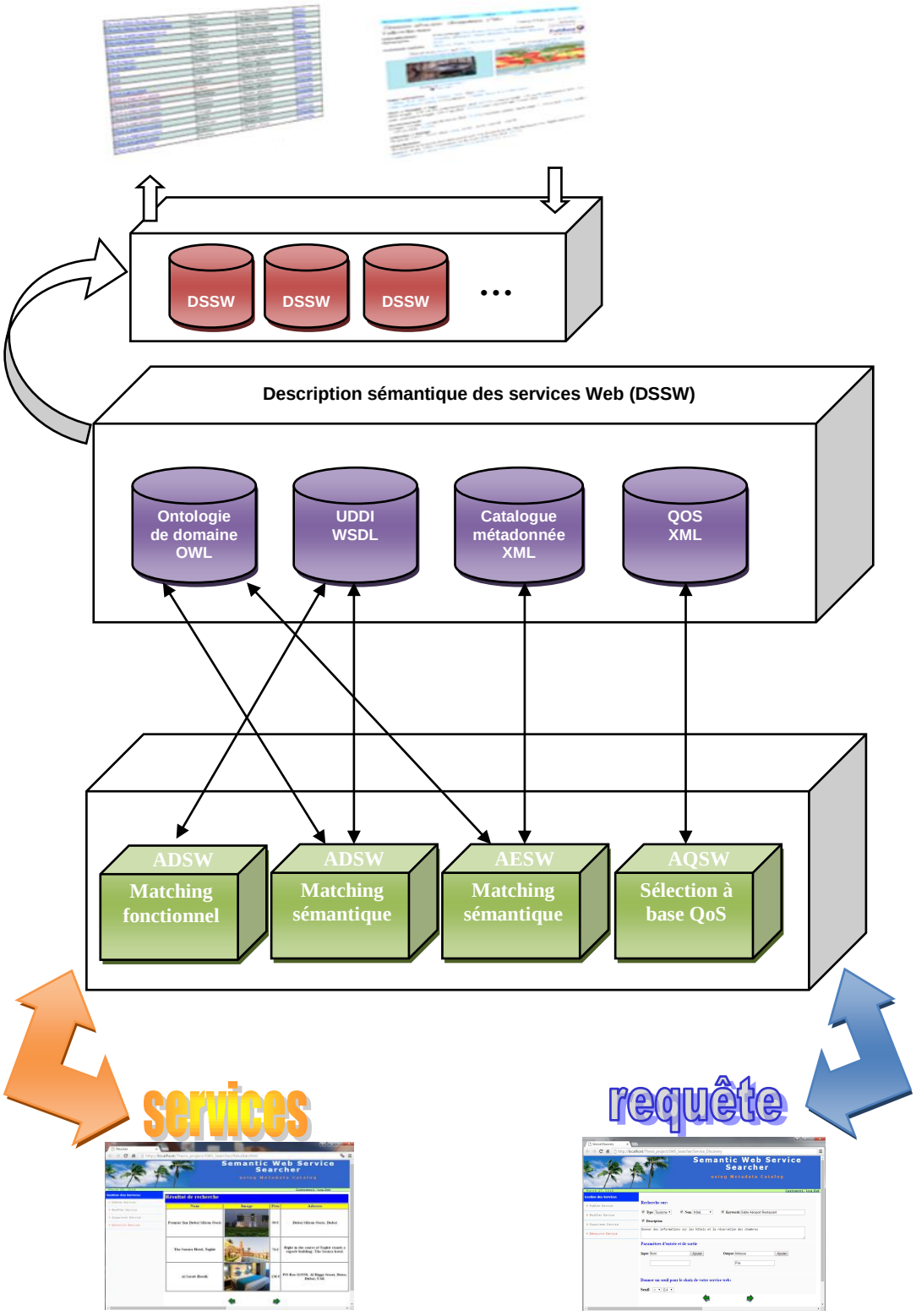

Figure VI.3 : Architecture générale de l'application proposée (Benseghir, 2015).

VI.3.2. Ontologies exploitées

Pour la mise en oeuvre de notre prototype nous faisons appel à une ontologie générale qui consiste à présenter les différents concepts de domaine d'application qui est dans notre cas le domaine de tourisme.

Le tourisme est un domaine qui se caractérise par une multitude de sources de données réparties. Ces sources de données sont souvent hétérogènes, distribuées et autonomes (Jellouli, 2008).

Dans notre travail, nous présentons l'ontologie de tourisme sous forme de quatre arborescences: Hôtel, Activité, Région, Transport (voir figure VI.4).

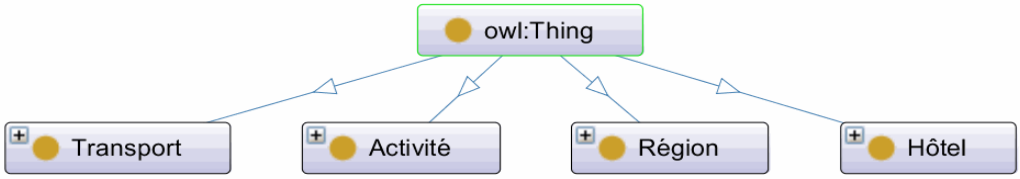

Figure VI.4 : Graphe des relations de l'ontologie « Tourisme ».

Les figures VI.5, VI.7, VI.8, VI.9 représente les graphes des relations des ontologies Hôtel, Activité, Région, Transport respectivement.

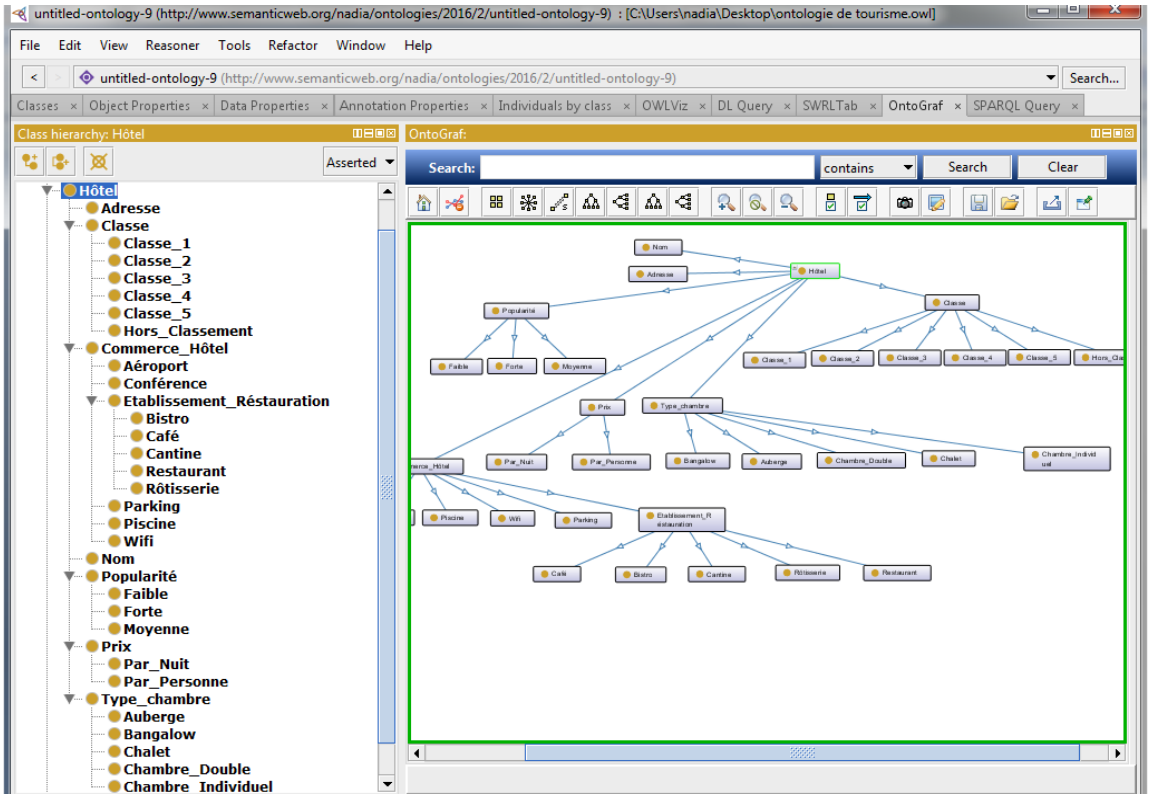

Figure VI.5 : Aperçu de l'ontologie « Hôtel » sous l'outil Protège 5.0.

Protégé offre la possibilité d'avoir le code de cette ontologie en langage XML-RDF comme suit :

```
<?xml version="1.0"?>
<rdf:RDF xmlns="http://www.semanticweb.org/ontologies/2016/2/untitled-ontology-9"
    xml:base="http://www.semanticweb.org/ontologies/2016/2/untitled-ontology-9"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:xml="http://www.w3.org/XML/1998/namespace"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
   <owl:Ontology rdf:about="http://www.semanticweb.org/ontologies/2016/2/untitled-ontology-9"/>
<owl:DatatypeProperty rdf:about="http://www.semanticweb.org/ontologies/2016/2/untitled-ontology-9#Adresse">
       <rdfs:subPropertyOf rdf:resource="http://www.w3.org/2002/07/owl#topDataProperty"/>
       <rdfs:domain rdf:resource="http://www.semanticweb.org/ontologies/2016/2/untitled-ontology-9#Hôtel"/>
       <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
   </owl:DatatypeProperty>
<owl:DatatypeProperty rdf:about="http://www.semanticweb.org/ontologies/2016/2/untitled-ontology-9#Nom">
       <rdfs:subPropertyOf rdf:resource="http://www.w3.org/2002/07/owl#topDataProperty"/>
       <rdfs:domain rdf:resource="http://www.semanticweb.org/ontologies/2016/2/untitled-ontology-9#Hôtel"/>
       <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
   </owl:DatatypeProperty>
<owl:Class rdf:about="http://www.semanticweb.org/ontologies/2016/2/untitled-ontology-9#Classe">
       <rdfs:subClassOf rdf:resource="http://www.semanticweb.org/ontologies/2016/2/untitled-ontology-9#Hôtel"/>
   </owl:Class>
  <owl:DatatypeProperty rdf:about="http://www.semanticweb.org/ontologies/2016/2/untitled-ontology-9#Classe_1">
       <rdfs:subPropertyOf rdf:resource="http://www.w3.org/2002/07/owl#topDataProperty"/>
       <rdfs:domain rdf:resource="http://www.semanticweb.org/ontologies/2016/2/untitled-ontology-9#Classe"/>
       <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#int"/>
   </owl:DatatypeProperty>
<owl:DatatypeProperty rdf:about="http://www.semanticweb.org/ontologies/2016/2/untitled-ontology-9#Classe_2">
       <rdfs:subPropertyOf rdf:resource="http://www.w3.org/2002/07/owl#topDataProperty"/>
       <rdfs:domain rdf:resource="http://www.semanticweb.org/ontologies/2016/2/untitled-ontology-9#Classe"/>
       <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#int"/>
   </owl:DatatypeProperty>
...
   <owl:Class rdf:about="http://www.semanticweb.org/ontologies/2016/2/untitled-ontology-9#Commerce_Hôtel">
       <rdfs:subClassOf rdf:resource="http://www.semanticweb.org/ontologies/2016/2/untitled-ontology-9#Hôtel"/>
   </owl:Class>
   <owl:Class rdf:about="http://www.semanticweb.org/ontologies/2016/2/untitled-ontology-9#Conférence">
       <rdfs:subClassOf rdf:resource="http://www.semanticweb.org/ontologies/2016/2/untitled-ontology-
9#Commerce_Hôtel"/>
   </owl:Class>
   <owl:Class rdf:about="http://www.semanticweb.org/ontologies/2016/2/untitled-ontology-9#Popularité">
       <rdfs:subClassOf rdf:resource="http://www.semanticweb.org/ontologies/2016/2/untitled-ontology-9#Hôtel"/>
   </owl:Class>
  <owl:Class rdf:about="http://www.semanticweb.org/ontologies/2016/2/untitled-ontology-9#Forte">
       <owl:equivalentClass>
          <owl:Restriction>
             <owl:onProperty rdf:resource="http://www.w3.org/2002/07/owl#topDataProperty"/>
             <owl:someValuesFrom rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
          </owl:Restriction>
       </owl:equivalentClass>
       <rdfs:subClassOf rdf:resource="http://www.semanticweb.org/ontologies/2016/2/untitled-ontology-
9#Popularité"/>
   </owl:Class>
...
```

Figure VI.6 : La syntaxe XML/RDF de l'ontologie « Hôtel ». Générer par OWL API (version 4.2) https://github.com/owlcs/owlapi

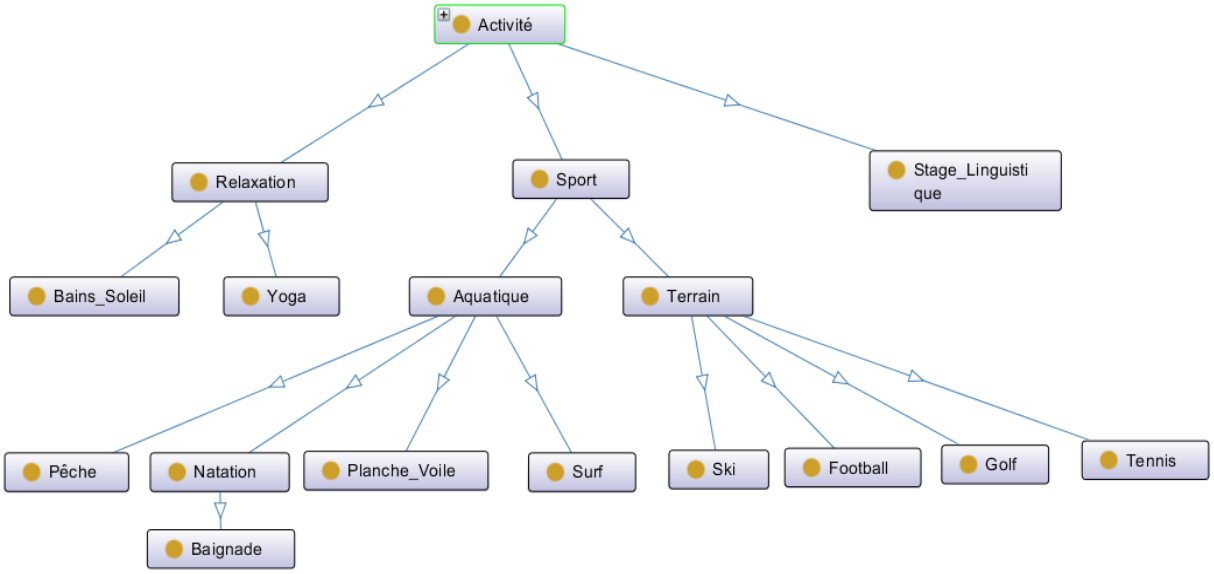

Figure VI.7 : Graphe des relations de l'ontologie « Activité ».

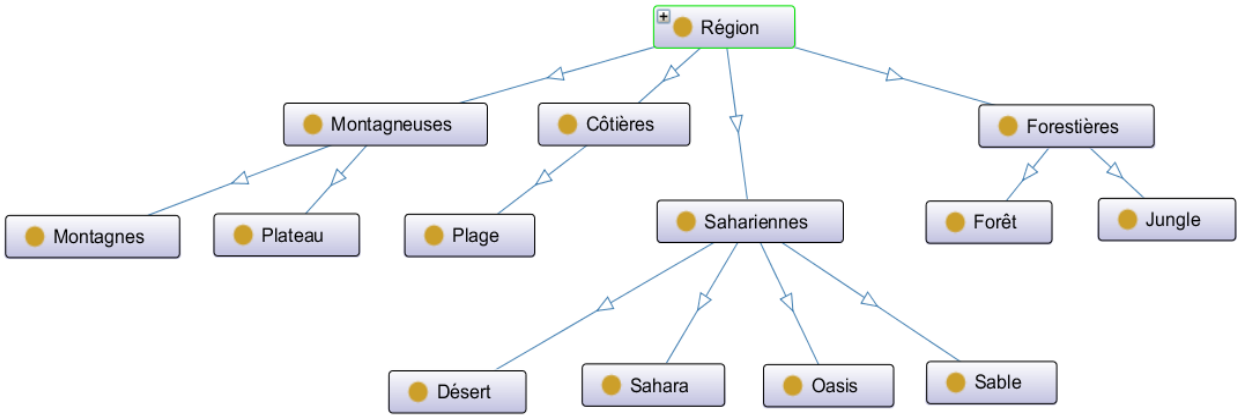

Figure VI.8 : Graphe des relations de l'ontologie « Région ».

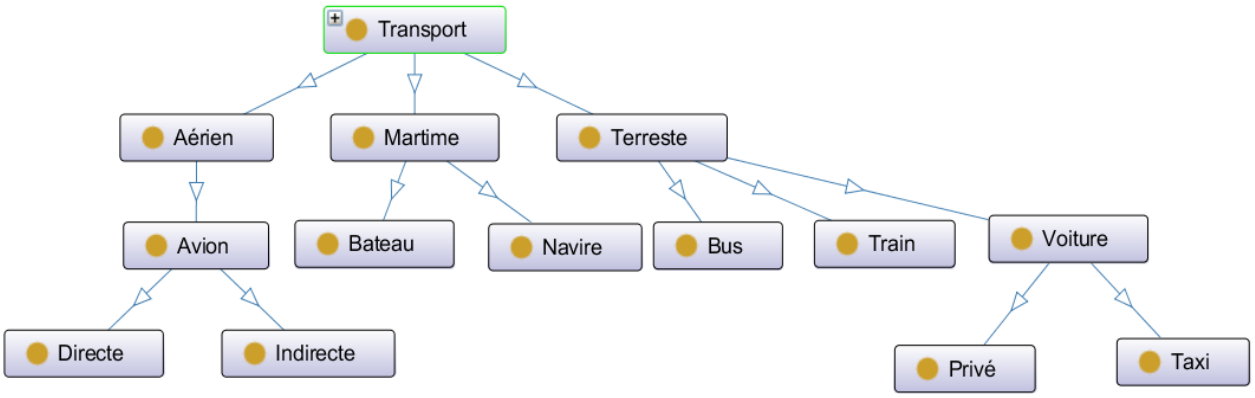

Figure VI.9 : Graphe des relations de l'ontologie « Transport ».

VI.4. Exploitation de la réalisation dans un cas d'utilisation

VI.4.1. Présentation de l'étude de cas

Notre étude embrasse plusieurs cas et elle est destinée à la démonstration de la validité et la fiabilité de notre architecture. Pour exprimer en détail le déroulement de notre approche de découverte sémantique des services Web, dans notre travail, une agence de voyages est notre étude de cas.

Une agence de voyages fournit plusieurs services Web tel que: consultation, réservation, paiement et annulation de billets de voyages, des chambres d'hôtel et des locations de voitures, etc. A cet effet, nous avons proposés trois services Web différents dans notre étude de cas :

- **Hôtel :** retourne les informations concernant la réservation d'un hôtel : les noms, les adresses, le nombre d'étoiles, le prix de la chambre, . . . etc;
- **Réservation :** retourne les informations concernant la réservation d'un voyage par avion, par train ou bien par autre moyen de transport qui est programmée selon la date ou la ville sélectionnée ;
- **Location :** retourne la disponibilité et les informations d'un modèle de voitures au près des agences de location de voiture selon la date ou la ville sélectionnée ;

Exemple : nous supposons qu'il ya dix services publiés sur le Web. Les tableaux VI.1, VI.2, VI.3 illustrent la description de ces services:

ServiceID	Nom	Inputs	Outputs	Classe	Prix
SWS1	Hôtel	Nom, Date	Adresse, Classe, Type_Chambre, Prix	5	318 €
SWS2	Hôtel	Nom	Nom, Classe, Type_Chambre, Prix	3	51 €
SWS3	Hôtel	Nom	Adresse, Nom, Prix_Nuit	4	138 €
SWS4	Hôtel	Nom, Date_Entrée, Date_Sortie	Adresse, Prix_Personne	4	91 €
SWS5	Hôtel	Nom, Date	Adresse, Prix_Nuit	4	94 €
SWS6	Hôtel	Nom	Adresse, Classe, Prix	3	90 €
SWS7	Réservation	Ville_Départ, Ville_Arrivé, Date	Code, Type, Prix	/	/
SWS8	Hôtel	Nom	Adresse, Prix	3	40 €
SWS9	Hôtel	Nom	Adresse, Prix, Classe	4	56 €
SWS10	Location	Lieu	Type_Véhicule, Prix	/	/

Tableau VI.1 : Description des paramètres fonctionnels des services Web.

ServiceID	Type	Nom	Spatial Data	Popularité	Description	Keywords
SWS1	Tourisme	Hôtel	Désert	Forte	Anantara qasr al sarab, Emirats Arabes Unis	Sable Rôtisserie Sport Natation
SWS2	Tourisme	Hôtel	Plateau	Moyenne	Golden tulip sfax, Avenue habib bourguiba,	Restaurant Piscine Conférence
SWS3	Tourisme	Hôtel	Sahara	Forte	Al sarab hôtel Deira, Dubaï	Sable, Wifi Aéroport Restaurant
SWS4	Tourisme	Hôtel	Montagneux	Faible	Novotel Constantine**,** Algeria	Cantine Parking
SWS5	Tourisme	Hôtel	Côtier	Forte	Sofitel algiers hamma garden Alger	Restaurant Baignade
SWS6	Tourisme	Hôtel	Oasis	Forte	Premier Inn Dubaï Silicon oasis	Restaurant Piscine, Sable Aéroport
SWS7	Tourisme	Réser-vation	**C**ôtier	Forte	Air Algérie	Avion, Algérie France
SWS8	Tourisme	Hôtel	Désert	Faible	EL Djamil Ouargla	Wifi, Sable Restaurant
SWS9	Tourisme	Hôtel	Oasis	Forte	The Saoura Hotel, Taghit Bechar	Restaurant Wifi Sable
SWS10	Tourisme	Locat-ion	Désert	Forte	RayMylV.I.P / Location de voitures - Rent a car Aïn Benian, Alger	Mercedes Citroën Hyundai

Tableau VI.2 : Description des paramètres non fonctionnels des services Web.

ServiceID	Temps_Réponse	Fiabilité	Disponibilité	Prix_Exécution
SWS1	0.2	0.6	0.5	0.3
SWS2	0.1	0.7	0.6	0.5
SWS3	0.5	0.6	0.4	0.2
SWS4	0.2	0.5	0.5	0.1
SWS5	0.7	0.3	0.6	0.5
SWS6	0.4	0.8	0.7	0.3
SWS7	0.6	0.2	0.4	0.6
SWS8	0.7	0.3	0.6	0.6
SWS9	0.3	0.5	0.7	0.4
SWS10	0.4	0.7	0.1	0.3

Tableau VI.3 : Description des paramètres de qualité des services Web.

Pour mieux expliquer le processus de découverte des services Web sémantiques, prenons l'exemple d'un enseignant chercheur qui veut planifier un voyage. Il exprime son besoin sous forme d'une requête qui comprend la réservation d'un hôtel. Cette requête modélise un service Web à travers les paramètres : input, output, description de service et le nom de service Web (voir figure VI.19).

Afin de satisfaire la demande de l'utilisateur, nous allons dans ce qui suit décrire les six étapes de la découverte dynamique des services Web sémantiques d'une façon automatique, en partant d'une requête déclarative simple spécifiée par un utilisateur.

VI.4.2. Enrichissement de requête selon le profil

L'enrichissement d'une requête exploite le profil de l'utilisateur pour reformuler sa requête en y intégrant des éléments de son centre d'intérêt ou de ses préférences. Cette technique d'enrichissement, courante dans les langages à mots clés en recherche d'information, est très récente en bases de données.

Nous avons adapté la méthode d'enrichissement définie dans (Koutrika, 2004) (Koutrika, 2005) pour la découverte des services Web. Dans cette approche, le profil de l'utilisateur est composé d'un ensemble de prédicats pondérés. Le poids d'un prédicat exprime son intérêt relatif pour l'utilisateur. Il est spécifié par un nombre réel compris entre 0 et 1. Ces prédicats sont utilisés pour enrichir la requête utilisateur.

Le processus d'enrichissement comporte deux phases principales :

(i) Sélection des prédicats qui vont enrichir la requête ;

(ii) Intégration de ces prédicats à la requête.

Exemple : soit un utilisateur ayant les préférences illustrées dans la figure VI.18, ces préférences peuvent être exprimées avec l'ensemble de prédicats suivants :

Profile:

Moyen_Transport='avion'	*0.5*	*(a)*
Prix_Hôtel < 150 €	*1.0*	*(b)*
Type_Voyage = 'directe'	*0.4*	*(c)*
Niveau_Confort_Transport = 2	*0.3*	*(d)*
Nombre_Jours_Voyage = 7	*0.2*	*(e)*
Classe_Hôtel < 5	*0.6*	*(f)*
Région_Hôtel = 'sahara'	*0.9*	*(g)*
Popularité_Hôtel='forte'	*0.7*	*(h)*
Temps_Réponse =	*0.8*	*(i)*
Fiabilité=	*0.6*	*(j)*
Disponibilité=	*0.55*	*(k)*
Prix_Exécution=	*0.7*	*(l)*
Sécurité=	*0.4*	*(m)*

Figure VI.10 : Représentation du profil utilisateur selon un ensemble de prédicats.

VI.4.2.1. Sélection des prédicats

Le profil contient deux types de prédicats : (i) prédicats de sélection et (ii) prédicats de jointure. Le poids de chaque prédicat de sélection exprime son importance par rapport aux autres prédicats de sélection. Par exemple, l'utilisateur a une plus forte préférence pour les voyages aux régions sahariennes (g), et il préfère également voyager en avion (a). Les prédicats de sélection ayant un poids égal à 1.0 (par exemple le prédicat (b)) sont considérés comme étant obligatoires et doivent être toujours satisfaits. La sélection des prédicats pour l'enrichissement de la requête utilisateur consiste à choisir les Top K prédicats qui sont en relations avec la requête. Le choix des Top K préférences peut être fait selon différents critères comme par exemple : choisir au plus K prédicats, choisir les prédicats dont le poids satisfait un seuil donné, etc.

Dans notre approche, le degré d'intérêt (poids) est le seul critère de sélection des prédicats du profil utilisateur. Notre méthode de sélection consiste à choisir les prédicats dont le poids satisfait un seuil donné par l'utilisateur pendant le chargement de son profil. Dans le profil illustré dans la figure VI.18, l'utilisateur donne un seuil supérieur à 0.5.

Donc suivant ce procédé, on peut déduire les huit préférences de profil qui sont liées à la requête utilisateur :

Prédicats Choisis:

Prix_Hôtel< 150 €	*1.0*	*(b)*
Classe_Hôtel < 5	*0.6*	*(f)*
Région_Hôtel = 'sahara'	*0.9*	*(g)*
Popularité_Hôtel='forte'	*0.7*	*(h)*
Temps_Réponse=	*0.8*	*(i)*
Fiabilité=	*0.6*	*(j)*
Disponibilité=	*0.55*	*(k)*
Prix_Exécution=	*0.7*	*(l)*

Figure VI.11 : Les Top K prédicats qui sont en relations avec la requête.

VI.4.2.2. Intégration des prédicats

La dernière étape de l'algorithme d'enrichissement est l'intégration des prédicats du profil à la requête. Cette étape est guidée par le paramètre K qui est le nombre de prédicats du profil devant être pris en compte.

Exemple : prenons par exemple la requête initiale illustrée dans la figure VI.19:

Requête Initiale:

```
For $z in $doc // ("DSSW_File.xml")
Where $z/type= "tourisme" and $z/nom ="Hôtel" and $z/keywords = "sable aéroport restaurant" and $z/description = "donner des informations sur les hôtels et la réservation des chambres" and $z/inputs="nom" and $z/outputs="adresse" and $z/outputs="prix"
Return $z/ServiceID
```

Figure VI.12 : Demande du client sous forme requête XQuery.

Après l'intégration des prédicats du profil illustrés dans la figure VI.11 à la requête initiale montrée dans la figure VI.12, nous obtenons le résultat final de l'enrichissement sous forme d'union des sous requêtes.

Requête Enrichie:

For $w in $doc// ("WSDL_File.xml")
Where $w/nom="Hôtel" and $w/inputs="nom" and $w/outputs="adresse" and $w/outputs="prix" and $w/Prix_Hôtel<150 € and $w/Classe_Hôtel < 5
Return $w/ServiceID
Union
For $z in $doc // ("Metedata_File.xml")
Where $z/type= "tourisme" and $z/nom="Hôtel" and $z/keywords = "sable aéroport restaurant" and $z/description = "donner des informations sur les hôtels et la réservation des chambres" and $z/inputs="nom" and $z/outputs="adresse" and $z/outputs="Prix" and $z/ Région_Hôtel = "sahara" and $z/ Popularité_Hôtel ="forte"
Return $z/ServiceID
Union All
For $p in $doc// ("QoS_File.xml")
Where $p/nom="Hôtel" and $p/Temps_Réponse = 0.8 and $p/Fiabilité = 0.6 and $p/Disponibilité = 0.55 and $p/Prix_Exécution = 0.7
Return $p/ServiceID

Figure VI.13 : Requête enrichie selon le profil utilisateur.

VI.4.3. Matching fonctionnel au niveau UDDI

Pour cette phase, initialement nous optons à appliquer l'algorithme de matching en se basant sur l'aspect syntaxique en utilisant l'interrogation des différentes pages (blanches, jaunes et vertes) basée sur le tModel (en utilisant par exemple le nom du service et le prix) ce qui nous donne comme résultat un ensemble de services Web mais sans aucune classification. Le résultat de cette phase est : SWS2, SWS3, SWS4, SWS5, SWS6, SWS8, SWS9.

VI.4.4. Matching sémantique au niveau UDDI

Pour enrichir la phase de recherche précédente, nous optons à renforcer l'aspect sémantique même au niveau registre UDDI en se basant sur le calcul de la similarité

sémantique entre les paramètres d'entrée (inputs) et les paramètres de sortie (outputs) de la requête et ceux de l'annonce (WSDL), ce qui nous permet d'avoir une classification des services Web proposés comme services pertinents. Le résultat de la phase précédente sera exploité dans cette étape pour restreindre l'espace de recherche. Le tableau VI.6 représente le résultat final de la similarité sémantique des paramètres fonctionnels au niveau UDDI.

ServiceID	**Inputs (service)**	**Inputs (requête)**	**∑ Similarité Sémantique**
		Nom	
SWS2	Nom	5	5
SWS3	Nom	5	5
SWS4	Nom	5	5
	Date_Entrée	0	
	Date_Sortie	0	
SWS5	Nom	5	5
	Date	0	
SWS6	Nom	5	5
SWS8	Nom	5	5
SWS9	Nom	5	5

Tableau VI.4 : Résultat de la similarité sémantique au niveau UDDI (Inputs).

ServiceID	**Outputs (service)**	**Outputs (requête)**		**∑ Similarité Sémantique**
		Adresse	Prix	
SWS2	Nom	2	2	19
	Classe	2	2	
	Type_Chambre	2	2	
	Prix	2	5	
SWS3	Adresse	5	2	16
	Nom	2	2	
	Prix_Nuit	1	4	
SWS4	Adresse	5	2	12
	Prix_Personne	1	4	
SWS5	Adresse	5	2	12
	Prix_Nuit	1	4	
SWS6	Adresse	5	2	18
	Classe	2	2	
	Prix	2	5	
SWS8	Adresse	5	2	14
	Prix	2	5	
SWS9	Adresse	5	2	18
	Prix	2	5	
	Classe	2	2	

Tableau VI.5 : Résultat de la similarité sémantique au niveau UDDI (Outputs).

ServiceID	Inputs	Outputs	∑ Degré Sémantique	Degré Normalisé
SWS2	5	19	24	1
SWS3	5	16	21	0.5714
SWS4	5	12	17	0
SWS5	5	12	17	0
SWS6	5	18	23	0.8571
SWS8	5	14	19	0.2857
SWS9	5	18	23	0.8571

Tableau VI.6 : Résultat final de la similarité sémantique au niveau UDDI.

Dans cette étape, on élimine les services qui ont un degré de similarité normalisé égal à 0 (on applique la formule VI.2 pour normaliser les degrés de similarité sémantique). Donc, on obtiendra comme résultat une liste triée des services Web pertinents: SWS2, SWS6, SWS3, SWS8, SWS9.

VI.4.5. Matching sémantique au niveau catalogue de métadonnées

Dans cette étape, nous appliquons aussi l'algorithme de matching (Chabeb, 2011) au niveau de catalogue de métadonnées. Il est utilisé pour trouver la correspondance sémantique entre les caractéristiques des services Web présentés dans la demande de l'utilisateur et ceux stockés dans le catalogue de métadonnées des services Web. Le résultat de la phase précédente (UDDI) sera exploité dans cette étape pour restreindre l'espace de recherche. Le tableau VI.7 représente le résultat final de la similarité sémantique des paramètres non fonctionnels des services Web.

ServiceID	Service / Requête	Type	Nom	Spatial Data	Popularité	Keywords			∑ Degré Sémantique	Degré Normalisé
		Tourisme	Hôtel	Sahara	Forte	Sable	Aéroport	Restaurant		
SWS2	Tourisme	5	3	3	3	3	3	3	112	0
	Hôtel	4	5	1	3	1	3	3		
	Plateau	4	1	1	1	1	1	1		
	Moyenne	4	4	1	2	1	1	1		
	Restaurant	4	4	1	1	1	1	5		
	Piscine	4	4	1	1	1	2	1		
	Conférence	4	4	1	1	1	2	1		
SWS6	Tourisme	5	3	3	3	3	3	3	135	0.9200
	Hôtel	4	5	1	3	1	3	3		
	Oasis	4	1	2	1	2	1	1		
	Forte	4	4	1	5	1	1	1		
	Restaurant	4	4	1	1	1	1	5		
	Piscine	4	4	1	1	1	2	1		
	Aéroport	4	4	1	1	1	5	1		
	Sable	4	1	2	1	5	1	1		
SWS3	Tourisme	5	3	3	3	3	3	3	137	1
	Hôtel	4	5	1	3	1	3	3		
	Sahara	4	1	5	1	2	1	1		
	Forte	4	4	1	5	1	1	1		
	Restaurant	4	4	1	1	1	1	5		
	Wifi	4	4	1	1	1	1	1		
	Aéroport	4	4	1	1	1	5	1		
	Sable	4	1	2	1	5	1	1		
SWS8	Tourisme	5	3	3	3	3	3	3	114	0.0800
	Hôtel	4	5	1	3	1	3	3		
	Désert	4	1	2	1	2	1	1		
	Faible	4	4	1	2	1	1	1		
	Restaurant	4	4	1	1	1	1	5		
	Wifi	4	4	1	1	1	1	1		
	Sable	4	1	2	1	5	1	1		
SWS9	Tourisme	5	3	3	3	3	3	3	117	0.2000
	Hôtel	4	5	1	3	1	3	3		
	Oasis	4	1	2	1	2	1	1		
	Forte	4	4	1	5	1	1	1		
	Restaurant	4	4	1	1	1	1	5		
	Wifi	4	4	1	1	1	1	1		
	Sable	4	1	2	1	5	1	1		

Tableau VI.7 : Résultat final de la similarité sémantique au niveau catalogue de métadonnées.

Dans cette étape, on élimine les services qui ont un degré de similarité normalisé égal à 0 (on applique la formule VI.2 pour normaliser les degrés de similarité sémantique). Donc, on obtiendra comme résultat une liste triée des services Web pertinents: SWS3, SWS6, SWS9, SWS8.

VI.4.6. Sélection à base de QoS

Cette étape consiste à calculer le score QoS de chaque service Web issu de la phase précédente. Pour faire ce calcul, il faut normaliser les poids de chaque qualité de service afin d'obtenir des valeurs entre 0 et 1. En se basant sur la méthode d'évaluation de qualité SAW (Simple Additive Weighting), la normalisation se fait selon les formules suivantes:

Pour le poids de chaque QoS représenté dans la description de requête :

$$W_j' = \frac{w_j}{\sum_{j=1}^{m} w_j} \qquad \textbf{(Formule VI.1)}$$

Pour la valeur de chaque QoS représenté dans la description de service :

- Pour les critères positifs : $V_{ij} = \frac{d_{ij} - d_j^{min}}{d_j^{max} - d_j^{min}}$ **(Formule VI.2)**
- Pour les critères négatifs : $V_{ij} = \frac{d_j^{max} - d_{ij}}{d_j^{max} - d_j^{min}}$ **(Formule VI.3)**

Où : $d_j^{max} \equiv Max_i[d_{ij}]$ *et* $d_j^{min} \equiv Min_i[d_{ij}]$

Le score de qualité de chaque service est donné par cette formule :

$$Score(QoS_i) = \sum W_j V_{ij} \qquad 0 \leq Score(QoS_i) \leq 1 \qquad \textbf{(Formule VI.4)}$$

Le tableau VI.8 représente le résultat de l'algorithme de calcul de score de qualité des services Web.

		Temps_Réponse	Fiabilité	Disponibilité	Prix_Exécution	Score QoS
	Poids	0.301	0.226	0.207	0.264	
ServiceID	SWS3	0.5	0.6	0	1	0.5501
	SWS6	0.75	1	1	0.75	0.8567
	SWS9	1	0.4	1	0.5	0.7304
	SWS8	0	0	0.666	0	0.1378

Tableau VI.8 : Résultat de l'algorithme de calcul de score QoS.

VI.4.7. Calcul global de degré de matching

Dans cette étape, on fait la somme des degrés de similarité sémantique calculés au niveau UDDI et au niveau catalogue de métadonnées avec le score de qualité des services Web. Le tableau suivant résume ce calcul :

ServiceID	Similarité Sémantique (UDDI)	Similarité Sémantique (Catalogue)	Score QoS	Score Global	Score Global Normalisé
SWS3	0.5714	1	0.5501	2.1215	0.7595
SWS6	0.8571	0.9200	0.8567	2.6338	1
SWS8	0.2857	0.0800	0.1378	0.5035	0
SWS9	0.8571	0.2000	0.7304	1.7875	0.6027

Tableau VI.9 : Calcul de score global

Selon le tableau VI.9 et selon le seuil donné par l'utilisateur (seuil > 0.4, voir figure VI.19), la liste finale des services Web pertinents par ordre est : SWS6, SWS3, SWS9.

VI.5. Présentation des interfaces du système

Nous allons illustrer les fonctionnalités de notre système en effectuant quelques prises d'écran, qui nous montrent les étapes de notre système de publication et de découverte dynamique des services Web sémantiques à base des ontologies OWL. Dans cette section nous présentons les interfaces qui permettent aux clients d'accéder et d'utiliser l'application proposée, ces interfaces sont développées avec le langage JSP (Java Server Page).

VI.5.1. Page d'accueil

Cette interface représente le premier contact de l'utilisateur avec notre système dans la phase de découverte ou la phase de publication des services Web. Elle a la forme classique d'une interface d'authentification.

Figure VI.14 : La page d'accueil.

VI.5.2. Interface de publication des services Web

Pour la publication, notre approche conçoit un store sémantique basé sur une base de données. La base de données est créée afin de correspondre aux spécifications de la description sémantique des services Web. Pour cela, nous avons créé une base de données MySQL[16].

En pratique, la publication se fait en trois étapes. Premièrement, notre agent fournisseur SW (AFSW) reçoit une description des services Web (WSDL, Métadonnées, QoS) et l'analyse. Si elle est une description valide alors il l'envoie à l'agent registre SW (ARSW) pour faire l'enregistrement du fichier WSDL dans le registre UDDI. Deuxièmement, une fois la description publiée dans le UDDI, l'agent registre SW retourne l'identifiant du service publié (ServiceID). Troisièmement, la description de service et son identifiant sont fusionnés et transférés aux : (i) agent enregistreur SW (AESW) pour enregistrer la description dans le catalogue de métadonnées au format XML; (ii) agent qualité SW (AQSW) pour enregistrer les informations de qualité de service au format XML.

[16] http ://www.mysql.fr

La phase de publication est assurée à travers deux interfaces :

Figure VI.15 : Interface de publication des services Web.

Figure VI.16 : Interface de description sémantique des services Web.

VI.5.3. Interface d'authentification de l'utilisateur

Afin de pouvoir accéder au notre système, chaque utilisateur, qui est déjà inscrit dans le système, doit s'identifier en saisissant dans la page d'authentification son login, son mot de passe (voir la figure VI.14). La vérification de ces informations se fait par l'agent système (AS), et ceci afin d'autoriser ou de refuser les accès aux ressources de la plateforme.

L'interface d'authentification utilisateurs offre la possibilité d'inscrire des nouveaux utilisateurs. L'utilisateur débute son inscription en appuyant sur le lien «User Registration» se trouvant en bas de la page d'accueil, cela l'amène à remplir des formulaires qui lui permet d'introduire son profil.

La figure VI.17 est utilisée pour saisir les données personnelles de l'utilisateur. Ces dernières représentent la partie statique du profil. Elles contiennent des informations qui décrivent l'utilisateur et ne dépendent pas du système à interroger. Ces informations sont classées selon deux catégories: (i) Identité, (ii) Profession.

Figure VI.17 : Interface des données personnelles d'utilisateur.

La figure VI.18 est utilisée pour saisir les préférences de l'utilisateur. Ces dernières représentent la partie la plus importante dans le profil utilisateur. Elle a une incidence directe sur le processus de découverte, puisque c'est à partir de ces informations que le système va pouvoir retourner le service Web pertinent correspondant aux besoins de l'utilisateur. Ces préférences sont classées selon deux catégories: (i) Domaine d'intérêts, (ii) Qualité de service Web attendue.

Nous proposons que l'utilisateur doive donner un poids dont la valeur entre 0 et 1 pour chaque préférence entrée, pour dire qu'une préférence est plus importante que l'autre. Un poids égal à 0 indique « l'absence de tout intérêt », alors qu'un poids égal à 1 indique « extrême intérêt ».

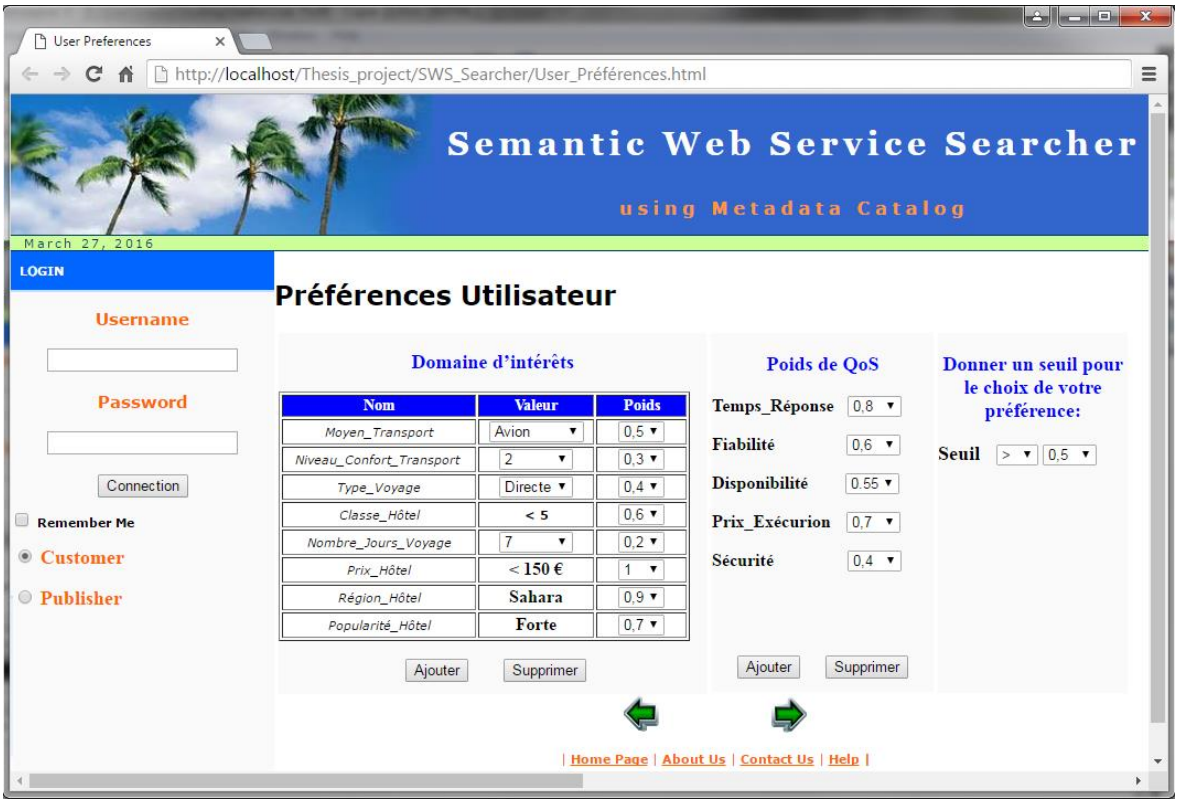

Figure VI.18 : Interface des préférences d'utilisateur.

Après une brève cérémonie de saisie, l'utilisateur valide ses informations et celles-ci seront enregistrées dans une base de profils utilisateurs.

VI.5.4. Interface de requête utilisateur

Une fois que l'utilisateur valide son profil, il pourra utiliser le système pour taper et introduire sa requête de découverte à travers les paramètres: input, output, type, nom de service Web, etc.

Figure VI.19 : Interface de requête utilisateur.

A la fin de cet étape d'introduire la requête par l'utilisateur, notre agent système commence d'exécuter l'algorithme d'enrichissement de cette requête selon le profil afin de découvrir des services Web satisfaisants le besoin d'utilisateur, et puis il lance le processus de découverte à travers la création d'un ensemble d'agents mobiles (ADSW). La section précédente (section VI.4) présente en détail l'algorithme d'enrichissement des requêtes utilisateur et le processus de découverte des services Web.

VI.5.5. Interface des résultats

Cette interface permet d'afficher les services Web pertinents au client. Pour l'exemple de requête illustré dans la figure VI.19, nous affichons à l'utilisateur l'interface suivante qui présente le résultat final.

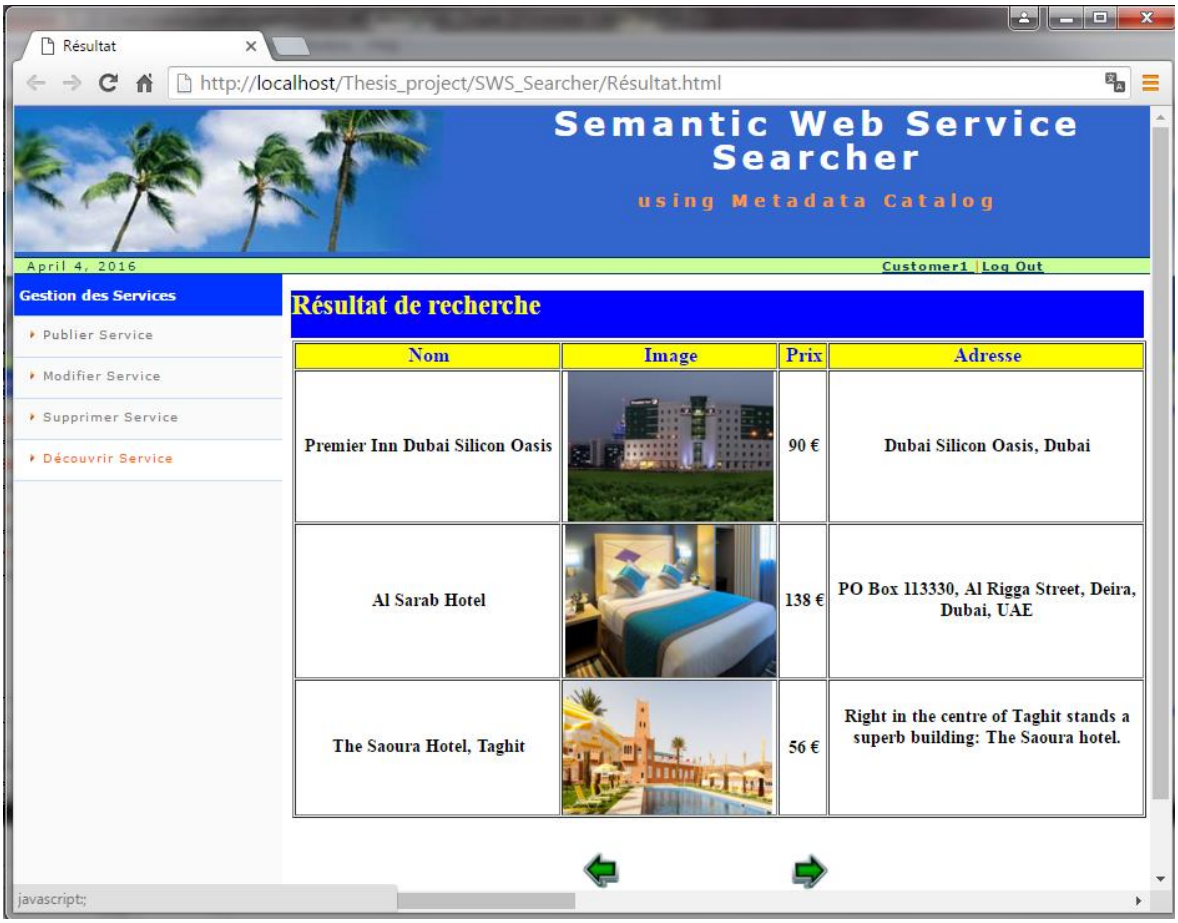

Figure VI.20 : Interface de résultat final.

VI.6. Conclusion

Au cours de cette dernière étape de notre travail, nous avons réalisé un prototype de système de découverte de services Web sémantiques, et nous avons présenté les outils utilisés pour l'implémentation. On a montré à travers ce système l'intérêt de l'utilisation des métadonnées pour la description sémantique des services Web, et leurs apports à la découverte de ces services Web, ainsi que l'intérêt de la prise en compte des préférences de l'utilisateur et l'utilisation des ontologies et les mesures de similarités pendant le processus de découverte. Nous avons utilisé plusieurs technologies de pointe inhérentes au domaine de services Web sémantiques de manière générale et au domaine des ontologies de manière plus particulière.

Notre première et grande satisfaction de ce travail est la maitrise acquise de ces hautes technologies durant cette période. Notre prototype permet d'une part d'enrichir la description des services Web en renforçant la description WSDL des services par une

couche de description sémantique, et d'autre part rendre la découverte des services Web plus pertinente.

De plus, à travers l'utilisation de la technologie d'agent mobile, notre système réduit les communications dans le réseau pour surmonter la bande passante faible et les fréquentes déconnexions du réseau. D'une seule interaction, l'utilisateur peut exécuter un ensemble de services. L'architecture proposée améliore la fonctionnalité de service en accomplissant sa tâche sans interruption et sans l'intervention constante de l'utilisateur.

Notre prototype représente une application Web qui propose aux utilisateurs humains d'utiliser les fonctionnalités de publication et de découverte des services Web. Notre souhait est de pouvoir concrétiser ce travail concrètement sur le terrain.

Chapitre VII

Conclusion Générale et Perspectives

VII.1. Bilan

Les services Web sont une technologie émergeante. Ce sont des composants softwares qui fournissent des fonctionnalités accessibles via des protocoles standardisés du Web. Basés sur le standard XML, ils sont caractérisés par leur indépendance aux plateformes et aux systèmes d'exploitation, ce qui a impliqué leur adoption par différentes organisations commerciales et industrielles offrant leurs services à travers le Web, et par conséquent l'augmentation du nombre des services Web offerts.

Le travail décrit dans cette thèse porte sur la problématique de découverte des Web services. Il a pour principal objectif de proposer un nouvel Framework pour la découverte des Web services et de concevoir une architecture permettant de concrétiser ce Framework. A l'origine, le problème de découverte des Web services est natif de deux sous problèmes rencontrés dans l'architecture des Web services, à savoir :

- **La description technique décrite en WSDL** : Le langage WSDL étant une spécification de bas niveau du fonctionnement d'un service, et étant insuffisantes d'une utilisation automatique des services Web.
- **Le point central de publication et de recherche** : le répertoire central de publication et de recherche souffre de plusieurs problèmes. En plus des difficultés classiques des systèmes centralisés, UDDI n'est pas conçu pour les environnements importants comme Internet. Il est plus adapté aux milieux fermés comme l'Intranet et l'Extranet.

Les travaux de recherche menés sur la description de services Web utilisent de plus en plus les ontologies et les métadonnées pour fournir une représentation de l'information sémantique. Pour le stockage de services, les architectures d'annuaires de services Web n'offrent pas assez de composants dédiés à la prise en charge complète de la sémantique. Pour la découverte de services, les approches sémantiques actuelles présentent plusieurs lacunes.

Notre objectif en termes de description de services Web est de proposer une description sémantique et complète, qui couvre les principaux éléments fonctionnels et non fonctionnels des services Web.

Notre objectif en termes de découverte de services Web est de proposer une architecture progressive et distribuée à base d'agents mobiles pour la découverte dynamique des services Web sémantiques avec des techniques d'appariement précises qui identifient précisément la similarité entre les éléments des deux services, requis et offert au moment de la découverte.

Les résultats finaux consistent en (i) un annuaire de services Web sémantiques qui supporte la description de services et des composants de publication et de découverte ; (ii) des algorithmes de matching et de découverte sémantiques de services Web.

La particularité de notre thèse se résume aux points suivants:

- **La richesse de la description des services Web :** la description des services Web est basée sur trois niveaux de description: le premier concerne le service lui même en utilisant le WSDL. Le deuxième concerne la couche sémantique dans notre cas est représentée par le catalogue de métadonnées des services Web. Le troisième concerne les critères de qualité des services Web (QoS).
- **La pertinence de la découverte :** la phase de découverte permet de restreindre l'espace de recherche et d'augmenter le nombre de services pertinents à travers l'application d'un :
  1) Algorithme de matching fonctionnel au niveau UDDI, en se basant sur l'aspect syntaxique en utilisant l'interrogation des différentes pages (blanches, jaunes et vertes).
  2) Algorithme de matching sémantique au niveau UDDI, il est utilisé pour trouver la similarité sémantique entre les paramètres d'entrée (inputs) et les paramètres de sortie (outputs) de service Web présentés dans la demande de l'utilisateur et ceux de l'annonce (WSDL).
  3) Algorithme de matching sémantique au niveau catalogue de métadonnées, il est utilisé pour trouver la correspondance sémantique entre les caractéristiques des services Web présentés dans la demande de l'utilisateur et ceux stockés dans le catalogue de métadonnées des services Web.
  4) Algorithme de calcul de score QoS de chaque service Web issu de la phase précédente.

5) Algorithme d'agrégation efficace pour calculer de manière pertinente le degré global d'appariement pour la découverte.

- **La décentralisation de la publication et la découverte :** notre architecture est principalement élaborée autour de la distribution d'annuaires de services. L'utilisation de plusieurs annuaires dans un processus de découverte offre principalement l'avantage de dépasser le problème du *"single point of failure"* et permet d'éviter les goulots d'étranglements comme lors de l'utilisation d'un annuaire unique centralisé.
- **L'utilisation des ontologies:** dans notre travail, nous avons utilisé une ontologie générale qui consiste à présenter les différents concepts de domaine d'application qui est dans notre cas le domaine de tourisme. Elle est utilisée pour le matching sémantique aux niveaux registre UDDI et le catalogue de métadonnées.
- **L'utilisation de la technologie d'agent mobile :** Le but du déplacement des agents est généralement d'accéder localement à des données ou à des ressources initialement distantes, d'effectuer le traitement en local et de ne déplacer que les données utiles. L'avantage de l'architecture proposée est qu'elle utilise les agents mobiles comme une entité de communication. Il s'agit ici de réduire le trafic sur le réseau et de diminuer la quantité d'informations échangées, ce qui minimisent le temps d'attente du client.

VII.2. Perspectives

Dans le cadre de notre objectif global qui consiste à permettre à l'utilisateur de profiter du service Web le plus approprié, nous proposons d'enrichir notre Framework en s'intéressant aux points suivants:

- **L'expérimentation de notre prototype dans plusieurs domaines :** cette perspective permet d'enrichir notre annuaire sémantique par d'autres spécifications sémantiques qui décrivent les services Web. Cet enrichissement permet, d'une part, d'avoir une ontologie de qualité des services Web englobant la plupart des concepts de qualité, d'autre part de faciliter la découverte sémantique des services Web.
- **La composition de services Web :** nous optons à faire des extensions à l'architecture proposée pour permettre la composition dynamique des services Web. Cette extension permet l'enrichissement de notre annuaire sémantique pour prendre en charge la découverte des services Web composables, en assurant les conditions nécessaires pour compléter la composition des services découverts.

NOS CONTRIBUTIONS

Publications Internationales

- Nadia Ben Seghir, Okba Kazar: « An architecture based on mobile agents for retrieving information from distributed sources », MASAUM Journal Of Reviews and Surveys (MJRS), Volume 1, Issue 1 (September 2009), pp 14-19
  http://www.masaumnet.com/archives/mjrs/volume1/issue1/mjrs010103.pdf

- Nadia Ben Seghir, Okba Kazar, Khaled Rezeg: "A Decentralized Framework for Semantic Web Services Discovery Using Mobile Agent". International Journal of Information Technology and Web Engineering, 10(4), 20-43, October-December 2015. DOI:10.4018/IJITWE.2015100102.
  www.igi-global.com/article/a-decentralized-framework-forsemantic-Web-services-discovery-using-mobileagent/147631?camid=4v1.

Communications Internationales

- Nadia Ben Seghir, Okba Kazar: « An architecture based on mobile agents for searching information from distributed systems », In the Proceedings of the 2nd UAE Symposium on Web Services (SWS'09), College of Information Technology, Zayed University, Dubai, UAE, April 15-16 2009.

- Nadia Ben Seghir, Okba Kazar: « Une architecture basée agents mobiles pour la recherche d'information dans des sources hétérogènes et réparties », In the Proceedings of the 2nd Conférence Internationale sur l'Informatique et ses Applications (CIIA'09), Saida, Algeria, May 3-4, 2009.

Bibliographie

(Abedi, 2012) L. Abedi, M. Nematbakhsh, A. Abdolmaleki: "A Model for Context Aware Mobile Payment". Journal of Theoretical and Applied Electronic Commerce Research, vol 7, issue 3, pp 1-10, december 2012.

(Acher, 2008) M. Acher : "Vers une ligne de services pour la grille : application à l'imagerie médicale". Rapport de stage Master recherche PLMT, Université de Nice Sophia-Antipolis, juin 2008.

(Amato, 1999) G. Amato, U. Straccia: "User Profile Modeling and Applications to Digital Libraries", Proc. 3rd European Conf. Research and Advanced Technology for Digital Libraries, ECDL, 1999.

(Ametller, 2003) J. Ametller, S. Robles, J. Borrell: "Agent migration over FIPA ACL messages". In MATA, pages 210–219, 2003.

(APML, 2007) APML http://www.masternewmedia.org/online_marketing/attention-profilingapml/ apml-beginners-guide-attention-profile-20071113.htm. http://liako.biz/2007/10/explaining-apml-what-it-is-why-you-want-it/

(Arabshian, 2006) K. Arabshian, H. Schulzrinne : "An Ontology-based Hierarchical Peer-to- Peer Global Service Discovery System". Journal of Ubiquitous Computing and Intelligence (JUCI), volume 1 (2), pp. 133-144, 2006.

(Arcangeli, 2002) J. P. Arcangeli, A. Hameurlain, G. Bernard, J. Francois Monineds : "Agents et code mobiles". Numéro thématique de la Revue des sciences et technologies de l'information, série Technique et science informatiques (RSTI-TSI), Hermès Science Publications - Lavoisier, Paris, 2002.

(Arcangeli, 2004) J. P. Arcangeli, S. Leriche, M. Pantel : "Development of Flexible Peer-to-Peer Information Systems using Adaptable Mobile Agents". In 1st Int. Workshop on Grid and Peer-to-Peer Computing Impacts on Large Scale Heterogeneous Distributed Database Systems (GLOBE'04), Zaragoza, Spain, pages 549–553. IEEE Computing Society, 30 août-3 septembre 2004.

(Bahloul, 2006) D. Bahloul : "Une approche hybride de gestion des connaissances basée sur les ontologies, application aux incidents informatiques". Thèse de doctorat, INSA de Lyon, 2006.

(Kashani, 2004) F. B. Kashani, C. Chen, C. Shahabi : "WSPDS: Web Services Peer-to-Peer Discovery Service". International Conference on Internet Computing (IC'04), 733–743, Las Vegas, Nevada, USA, 2004.

(Baousis, 2006) V. Baousis, E. Zavitsanos, V. Spiliopoulos, S. Hadjiefthymiades, L. Merakos, G. Veronis : "Wireless Web Services using Mobile Agents and Ontologies". IEEE International Conference on Pervasive Services (ICPS), Lyon, France, pp : 69 – 77, 26-29 June 2006.

(Baumann, 2000) J. Baumann : "Mobile Agents: Control Algorithms". Lecture Notes in Computer Science, Springer Berlin Heidelberg, ISBN: 978-3-540-41192-5. 2000.

(Bellifemine, 1999) F. Bellifemine, A. Poggi, G. Rimassa: "JADE — A FIPA-compliant agent framework". In Proceedings of the 4th International Conference on the Practical Applications of Agents and Multi-Agent Systems (PAAM-99), pages 97–108, London, UK, 1999. The Practical Application Company Ltd.

(Benaboud, 2013) R. Benaboud, R. Maamri, Z. Sahnoun : "Agents and owl-s based semantic Web Service discovery with user preference Support". International Journal of Web & Semantic Technology (IJWesT) , 4 (2). 2013.

(Benattou, 2002) M. Benattou, J. M. Bruel: "Active objects for coordination in distributed testing". Lecture Notes in Computer Science, 2425 :348–357, 2002.

(Benmokhtar, 2004) S. Benmokhtar : "Synthèse ad hoc de services Web dans les environnements de l'informatique diffuse". Rapport de stage DEA Systèmes Informatiques Répartis (SIR), Université Pierre et Marie Curie 2003-2004.

(Benna, 2008) A. Benna, N. Boudjlida, H. Talantikite : "SAWSDL, Mediation and XQUERY for Web Services Discovery". NOTERE'08: 8ème Conférence Internationale sur les Nouvelles TEchnologies de la REpartition , Lyon France, 23-27 juin 2008.

(Benseghir, 2009) N. Benseghir, O. Kazar: " Une architecture basée agents mobiles pour la recherche d'information dans des sources hétérogènes et réparties". In the Proceedings of the 2nd Conférence Internationale sur l'Informatique et ses Applications (CIIA'09), Saida, Algeria, May 3-4, 2009.

(Benseghir, 2015) N. Benseghir, O. Kazar, K. Rezeg: "A Decentralized Framework for Semantic Web Services Discovery Using Mobile Agent". International Journal of Information Technology and Web Engineering, 10(4), 20-43, October-December 2015. DOI:10.4018/IJITWE.2015100102. www.igi-global.com/article/a-decentralized-framework-forsemantic-Web-services-discovery-using-mobileagent/147631?camid=4v1.

(Bentamrout, 2006) R. Bentamrout: "Jregistre: Un Registre Uddi Extensible". Mémoire présenté comme exigence partielle de la maîtrise en informatique. Université Du Québec À Montréal. Janvier 2006.

(Bernard, 1999) G. Bernard : "Technologie du code mobile : état de l'art et persecptive". In Colloque Francophone sur l'Ingénierie des Protocoles (CFIP'99), Nancy, France, Avril 1999.

(Berners-Lee, 2001) T. Berners-lee, J. Hendler, O. Lassila : "The semantic Web". Scientific American, Mai 2001.

(Bernier, 2000) S. Bernier: "Conception et implantation basées sur des Composants répartis d'une station terrestre Virtuelle de communication satellite". Mémoire présenté au département de mathématiques Et d'informatique en vue de l'obtention du grade de maîtres sciences, septembre 2000.

(Booth, 2004) D. Booth, H. Haas, F. McCabe, E. Newcomer, M. Champion, C. Ferris, D. Orchard: "Web services architecture". W3C. 11 février 2004.

(Bottraud, 2004) J. C. Bottraud , G. Bisson , M. F. Bruandet : "Apprentissage de profils pour un agent de recherche d'information". Coria'04, IRIT Toulouse France, 2004.

(Bourdon, 2006) S. ElFalou, F. Bourdon : "Agent mobile et recherche d'information sur le Web : une solution basée sur le MDP". Reconnaissance des Formes et Intelligence Artificielle (RFIA06), 2006.

(Bourdon, 2007) J. Bourdon, P. Beaune, H. Fiorino : "Architecture multi-agents pour la composition automatique de Web services", Atelier IAWI - Plate-forme AFIA, 2007.

(Bouzeghoub, 2005) M. Bouzeghoub, D. Kostadinov: "Personnalisation de l'information: aperçu de l'état de l'art et définition d'un modèle flexible de profils". CORIA 2005: 201-218.

(Bouzeghoub, 2007) M. Bouzeghoub, S. Calabretto, N. Denos, R. Harrathi, D. Kostadinov, A. Nguyen, V. Peralta : "Accès personnalisé aux informations: approche dirigée par la qualité". INFORSID 2007: 105-120.

(Bouzguenda, 2006) L. Bouzguenda, R. Bouaziz, E.Andonoff : "Utilisation d'Ontologies pour la Coordination dans le Workflow". Inter-Organisationnel Lâche, (2006), 10-11.

(Brahimi, 2009) M. Brahimi, L. Seinturier, M. Boufaïda: "Multi-Agent Architecture for Developing Cooperative E-Business Applications". International Journal of Information Systems and Supply Chain Management (IJISSCM), volume 2, N°4: pp.43-62, 2009.

(Bressoles, 2003) G. Bressoles : "La qualité de service traditionnelle versus la qualité de service électronique : similarités, différences et voies futures de recherche". Actes des 2èmes Journées Normandes de Recherche sur la consommation « Sociétés et Consommation », Rouen, 2003.

(Bruijn, 2007) J. D. Bruijn, J. Domingue, D. Fensel, H. Lausen, A. Polleres, D. Roman, M. Stollberg: "Enabling Semantic Web Services: The Web Service Modeling Ontology". Edition Springer, 2007.

(Brustoloni, 1991) J. C. Brustoloni : "Autonomous agents: characterisation and requirements". Carnegie Mellon Technical Report CMU-CS-91-204, Carnegie Mellon University, Pittsburgh. 1991.

(Burstein, 2004) M. Burstein, J. Hobbs, O. Lassila, D. Martin, D. McDermott, S. McIlraith, S. Narayanan, M. Paolucci, B. Parsia, T. Payne, E. Sirin, N. Srinivasan, K. Sycara : "OWL-S : Semantic markup for Web services". Pages 63-100, 2004.

(Cardoso, 2006) J. Cardoso, A. Sheth : "The semantic Web and its applications". In Semantic Web Services, Processes and Applications, volume 3 of Semantic Web and Beyond, pages 3-33. Springer US, 2006.

(Cardoso, 2007) J. Cardoso: "Semantic Web Services: Theory, Tools, and Applications". University of Madeira, Portugal, Information Science reference, édition Dunod, 2007.

(Casati, 2000) F. Casati, S. Ilnicki, L. Jin, V. Krishnamoorthy, M. C. Shan : "Adaptive and dynamic service composition in E-Flow". In Proceeding of the 12th Conference on Advanced Information Systems Engineering (CAISE), S. Verlag, pages 13-31, Stockholm, Sweden, June 2000.

(Cerami, 2002) E. Cerami : "Web Services Essentials". Edition O'Reilly, Février 2002.

(Celik, 2013) D. Celik, A. Elçi : "A broker-based semantic agent for discovering Semantic Web services through process similarity matching and equivalence considering quality of service". Science China Information Sciences (SCI) , 56 (1), 012102:1–012102:24. 2013.

(Chabeb, 2011) Y. Chabeb : "Contributions à la Description et la Découverte de Services Web Sémantiques". Thèse de doctorat de Télécom SudParis dans le cadre de l'école doctorale S&I en co-accréditation avec l' Universite d'Evry-Val d'Essonne. Soutenue le novembre 2011.

(Chappell, 2002) D. Chappell, T. Jewell : "Java Web Service". Edition O'Reilly, Mars 2002.

(Châtel, 2010) P. Châtel : "Une approche qualitative pour la prise de décision sous contraintes non fonctionnelles dans le cadre d'une composition agile de services". Thèse de Doctorat de l'université Pierre et Marie Curie Paris 6, de l'Ecole Doctorale d'Informatique, Télécommunication et Electronique de Paris, 2010.

(Chatterjee, 2003) S. Chatterjee, J. Webber: "Developing Enterprise Web Services". Edition Prentice Hall PTR, le 14 November 2003.

(Chauvet, 2002) J. M. Chauvet : "Services Web avec SOAP, WSDL, UDDI, ebXML…". Edition EYROLLES, 2002.

(Chess, 1994) D. Chess, C. Harrison, A. Kershenbaum: "Mobile Agents : Are They a Good Idea ?". Technical Report RC 19887, IBM Research Division, T.J. Watson Research Center, 1994.

(Chiaramonti, 2003) C. Chiaramonti : "Semantic Web ou Services Web". Edition O'Reille, le 16 Mai 2003.

(Cros, 2005) C. C. D. Cros : "Agents Mobiles Coopérants pour les Environnements Dynamiques". Thèse de doctorat, Institut National Polytechnique de Toulouse, 2 décembre 2005.

(Colaço, 1997) J. L. Colaço : "Analyses Statiques de Langages d'Acteurs par inférence de types". Thèse de doctorat, ENSEEIHT, Toulouse, octobre 1997.

(Daconta, 2003) M. Daconta, L. Obrst, K. Smith: "Developing the Semantic Web: a Guide to the Future of XML, Web Services, and Knowledge Management". Edition Wiley, 2003.

(Dallons, 2004) G. Dallons: "DAML-S : interactions, critique et évaluation". Institut d'informatique des FUNDP, Namur, Belgique, 2004.

(Dardailler, 2002) D. Dardailler : "Du Web sémantique aux Web services, nous automatisons le travail des applications". Article paru au journal du net, le 08 Février 2002.

(Dietze, 2011) S. Dietze, A. Gugliotta, J. Domingue, M. Mrissa : " Mediation Spaces for Similarity-Based Semantic Web Services Selection". International Journal of Web Services Research, 8(1), 1-20, January-March 2011. www.igi-global.com/article/mediation-spaces-similaritybased-semantic/50490?camid=4v1.

(Dolog, 2003) P. Dolog, W. Nejdl : "Challenges and Benefits of the Semantic Web for User Modelling", 2003.

(Domel, 1996) P. Domel : "Mobile Telescript agents and the Web". In Digest of Papers. COMPCON '96. Technologies for the Information Superhighway. Forty-First IEEE Computer Society International Conference, pages 52–57. IEEE Computer Society Press, 2. 1996.

(Dou, 2012) W. Dou, C. Lv, X. Zhang, J. Chen: "A Collaborative QoS-Aware Service Evaluation Method Among Multi-Users for a Shared Service". International Journal of Web Services Research, 9(1), 30-50, January-March 2012. www.igi-global.com/article/collaborative-qos-aware-serviceevaluation/64222?camid=4v1.

(ebXML) e-business XML, spécification d'OASIS, disponible à http://www.ebxml.org/

(ElBouhissi, 2014) H. ElBouhissi, M. Malki, M. A. Sidi Ali Cherif: "From User's Goal to Semantic Web Services Discovery: Approach Based on Traceability". International Journal of Information Technology and Web Engineering, 9(3), 15-39, July-September 2014. www.igi-global.com/article/from-users-goal-to-semantic-webservices-discovery/123182?camid=4v1.

(ELFalou, 2006) S. ELFalou : "Programmation répartie, optimisation par agent mobile". Thèse du Doctorat soutenue le 29 Novembre 2006, de l'université de CAEN.

(Elfirdoussi, 2014) S. Elfirdoussi, C. Ayyad, Z. Jarir, M. Quafafou : "Ranking Web Services Using Web Service Popularity Score". International Journal of Information Technology and Web Engineering, 9(2), 78-89, April-June

2014. www.igi-global.com/article/ranking-Web-services-using-webservice-popularity-score/115936?camid=4v1

(El-Sayed, 2012) M. El-Sayed Mohamed: "Wireless Semantic RESTFUL Services Using Mobile Agents". OJEEE, Volume (4) No (1), janvier 2012.

(Emekci, 2004) F. Emekci, O. D. Sahin, D. Agrawal, A. ElAbbadi : "A Peer-to-Peer Framework for Web Service Discovery with Ranking". In the Proc of the IEEE International Conference on Web Services (ICWS'04), California USA, 2004, pp. 192-199.

(Essafi, 2005) T. Essafi, N. Dorta, D. Seret : "A Scalable Peer-to-Peer Approach To Service Discovery Using Ontology". In the Proc of 9th World Multiconference on Systemics, Cybernetics and Informatics. Orlando, 2005.

(Falquet, 2001) G. Falquet, C. L. Mottaz-Jiang : "Navigation hypertexte dans une ontologie multi-points de vue". Nîmes TIC, 2001.

(Finder) Service finder: "a search engine for Web services discovery", disponible à www.service-finder.eu

(FIPA , 2004) FIPA, http://www.fipa.org

(Foukarakis, 2007) I. Foukarakis, A. Kostaridis, C. Biniaris, D. Kaklamani, I. Venieris : "Webmages: An agent platform based on Web services". Computer Communications, 30(3) :538{545. 2007.

(Friesen, 2005) A. Friesen, M. Altenhofen : "Semantic Discovery Optimization Matching composed semantic Web services at publishing time". WEBIST 2005: 347-350.

(Gardien, 2002) G. Gardien : "XML des bases de données aux Services Web". Edition Dunod, 2002.

(Gauch, 2003) S. Gauch, J. Chaffe, A. Pretschner: "Ontology-Based User Profiles for Search and Browsing", To appear in J. User Modeling and User-Adapted Interaction, the Journal of Personalization Research , Special Issue on User Modeling for Web and Hypermedia Information Retrieval, 2003.

(Geib, 1999) J. M. Geib, C. Gransart, P. Merle: "CORBA : des concepts à la pratique". Editions Dunod, Paris, France, Octobre 1999.

(Gesnu, 2003) X. Gesnu : "Développer des Services Web XML et des composants serveurs". Edition Dunod, 2003.

(GIS, 2001) Open GIS Consortium: "Basic Model Draft Candidate Implementation Specification 0.0.8". 2001.

(Goldbeck, 2005) J. Golbeck: "Personalizing applications through integration of inferred trust values in semantic Web-based social networks". Department of Computer Science, University of Maryland, 2005.

(Gray, 1996) R. S. Gray, D. Kotz, S. Nog, D. Rus, G. Cybenko: "Mobile Agents for Mobile Computing". Technical Report PCS-TR96-285, Department of Computer Science, Dartmouth College. 1996.

(Gray, 2001) R. S. Gray, G. Cybenko, D. Kotz, D. Rus: "Mobile agents: Motivations and State of the Art". AAAI/MIT-Press, 2001.

(Gray, 2002) S. R. Gray, G. Cybenko, D. Kotz, R. A. Peterson, D, Rus: "D'agents : Applications and performance of a mobile-agent system". Softw, Pract. Exper, 32(6): 543–573, 2002.

(Gronroos, 1982) C. Gronroos: "Strategic Management and Marketing in the Service Sector". Helsingfors: Swedish School of Economics and Racine.- Administration, 1982.

(Gruber, 1993) T. R. Gruber: "Toward Principles for the Design of Ontologies Used for Knowledge Sharing". Presented at International Workshop on Formal Ontology, Padova, Italy, 1993.

(Guidara, 2015) I. Guidara, N. Guermouche, T. Chaari, M. Jmaiel, S. Tazi: "Time-Dependent QoS Aware Best Service Combination Selection". International Journal of Web Services Research, 12(2), 1-25, April-June 2015. www.igi-global.com/article/time-dependent-qos-aware-bestservice-combination-selection/126288?camid=4v1.

(Guillermo, 2010) G. Guillermo Valente : "Enrichissement de requêtes et visualisation sémantique dans une coopération de systèmes d'information : méthodes et outils d'aide à la recherche d'information", Thèse Pour l'obtention du titre de Docteur de l'Université de Bourgogne, Soutenue le 14 Décembre 2010.

(Gunther, 1998) N. J. Gunther: "The Practical Performance Analyst". Published by McGraw-Hill, 1998.

(Hu, 2005) J. Hu, C. Guo, H. Wang, P. Zou: "Web Services Peer-to-Peer Discovery Service for Automated Web Service Composition". Springer-Verlag Berlin Heidelberg (LNCS 3619), ICCNMC 2005, Zhangjiajie China, 2005, pp. 509-518.

(Huang, 2013) J. Huang, C. Lin: "Improving Energy Efficiency in Web Services: An Agent-Based Approach for Service Selection and Dynamic Speed Scaling". International Journal of Web Services Research, 10(1), 29-52, January-March 2013 29 www.igi-global.com/article/improving-energy-efficiency-inweb-ervices/86261?camid=4v1.

(Huhn, 1999) M. N. Huhn, L.M. Stephens: "Personal Ontologies". Internet Computing, Vol. 3, No. 5, pp. October 1999.

(Huhns, 2005) M. Huhns, M. Singh: "Service-oriented computing: Key concepts and principles". IEEE Internet Computing, volume 9(1), pages 75-81.2005.

(Inagaki, 1999) A. Y. Inagaki, K. Toyama, N. Kawaguchi: "Magnet: Adhoc network system based on mobile agents". September 30 1999.

(Ismael, 1999) L. Ismael, D, Hagimont: "A performance evaluation of mobile agent paradigm". In Proceedings of the International Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA' 99), pages 306–313, Novembre 1999.

(ISO, 2000) ISO 840: "Quality Management and Quality Assurance Vocabulary". Technical Report, International Organization for Standardization, 2000.

(ISO, 2001) ISO 19119: "Geographic information – Services". Draft International Standard, 2001. http://www.ncits.org/ref-docs/DIS19119.PDF.

(ISO, 2005) ISO: "ISO en bref". 2005. http://www.iso.org/iso/fr/aboutiso/isoinbrief/isoinbrief.html.

(Jeffrey, 2006) J. P. T. Jeffrey, M. Lu: "Security Modeling and Analysis of Mobile Agent Systems". Series In Electrical And Computer Engineering, World Scientific Publishing Company, ISBN-13: 978-1860946349. 2006.

(Jellouli, 2008) I. Jellouli, M. El Mohajir, E. Zimanyi: "Classification conceptuelle et ontologie de domaine pour l'intégration sémantique des données". e-TI - la revue électronique des technologies d'information, Numéro 5, 5 novembre 2008.

(Johansen, 1998) D. Johansen: "Mobile agent applicability". Lecture Notes in Computer Science, 1477 :80–98, 1998.

(Johansen, 2002) D. Johansen, K. J. Lauvset, R. V. Renesse, F. B. Schneider, N. P. Sudmann, K. Jacobsen: "A TACOMA retrospective". Software & Practice and Experience, 32(6): 605–619, May 2002.

(Johansen, 2004) D. Johansen: "Mobile agents: Right concept, wrong approach". In Mobile Data Management, pages 300–301, 2004.

(Jones, 2002) M. T. Jones: "Java mobile agents and the Aglets SDK". Dr. Dobb's Journal of SoftwareTools, 27(1): 42, 44, 46–48, January 2002.

(Jun, 2002) L. Jun, L. Xianlang, H. Hong, Z. Xu: "Application of mobile agent in wide area network". In IEEE, 2002.

(Kadima, 2003) H. Kadima, V. Monfort: "Les Web services techniques, démarches et outils". Edition Dunod, Paris, France, 2003.

(Karray, 2013) A. Karray, R. Teyeb, M. Ben Jemaa: "A heuristic approach for Web service discovery and selection". International Journal of Computer Science and Information Technology (IJCSIT) , 5 (2), 2013.

(Kellert, 2003) P. Kellert, F. Toumani : "Les Web services sémantiques". Spécifique 32 CNRS / STIC, Web sémantique Rapport final décembre 2003.

(Ketel, 2009) M. Ketel: "A Mobile Agent Based Framework for Web Services". Proceedings of the 47th Annual Southeast Regional Conference. March 19-21, 2009.

(Kobayashi, 1995) N. Kobayashi, M. Nakade, A. Yonezawa: "Static analysis of communication for asynchronous concurrent programming languages". In Second International Static Analysis Symposium (SAS'95), volume 983, pages 225–242. Springer-Verlag, 1995.

(Kostadinov, 2007) D. Kostadinov : "Personnalisation de l'information : une approche de gestion de profils et de reformulation de requêtes". Thèse de doctorat, Université de Versailles Saint- Quentin-En-Yvelines, 19 Décembre 2007.

(Koutrika, 2004) G. Koutrika, Y. Ioannidis: "Personalization of Queries in Database Systems". Proceedings of the 20th International Conference on Data Engineering, (pp. 597-608). Boston, Massachusetts, USA. 2004.

(Koutrika, 2005) G. Koutrika, Y. Ioannidis: "Personalized Queries under a Generalized Preference Model". Proceedings of the 21st International Conference on Data Engineering, (pp. 841-852). Tokyo, Japan. 2005.

(Kravari, 2012) K. Kravari, N. Bassiliades: "Advanced Agent Discovery Services" WIMS'12, June 13-15, 2012, Craiova, Romania. Copyright © 2012 ACM 978-1-4503-0915-8/12/06.

(Kreger, 2001) H. Kreger: "Web Service Conceptual Architecture". Edition IBM Software Group, Mai 2001.

(Kurki, 1999) T. Kurki, S. Jokela, R. Sulonen, M. Tirpeinen: "Agents in delivering personalized content based on semantic metadata". In Proc, AAAI Sping Symposium, 1999.

(Li, 2009) M. Li, B. Yu, V. Sahota, M. Qi: "Web Services Discovery with Rough Sets". International Journal of Web Services Research, 6(1), 69-86, January-March. 2009.

(Libsie, 2002) M. Libsie, H. Kosch : "Content adaptation of multimedia delivery and indexing using MPEG-7". In Proceedings of the tenth ACM international conference on Multimedia (MM-02), pages 644–646, New York, December 1–6 2002. ACM Press.

(Lopes, 2005) D. C. P. Lopes : "Etude et applications de l'approche MDA pour des platesformes de Services Web". Thèse de Doctorat, UFR Sciences et Techniques, Université de Nantes, le 11 Juillet 2005.

(Lopez, 2008) C. Lopez-Velasco : "Sélection et composition de services Web pour la génération d'applications adaptées au contexte d'utilisation". Thèse de doctorat, École Doctorale Mathématiques, Sciences et Technologie de l'Information, Informatique, Grenoble, 2008.

(Loureiro, 2001) S. Loureiro : "Mobile Code Protection". Thèse de doctorat, ENST Paris / Institut Eurecom, 2001.

(Lu, 2012) G. Lu, T. Wang, G. Zhang, S. Li : "Semantic Web Services Discovery Based on Domain Ontology". World Automation Congress (WAC) (pp. 1-4). Puerto Vallarta, Mexico: IEEE, 2012.

(Maes, 1994) P. Maes: "Agents that reduce work and information overload". Communications of the ACM, Vol. 37, No. 7, pp. 30-40. 1994.

(Mandreoli, 2007) F. Mandreoli, A. M. Perdichizzi, W. Penzo: "A P2P-based Architecture for Semantic Web Service Automatic Composition". IEEE computer Society DOI 10.1109/DEXA2007, Regensburg Germany, 2007, pp. 429-433.

(Martin, 2004) D. Martin et al.: "Owl-s: Semantic markup for Web services". Technical report, W3C. 2004.

(Melliti, 2004) T. Melliti: "Interopérabilité des Services Web complexes". Thèse de Doctorat, Université Paris IX Dauphine, le 8 Décembre 2004.

(Mika, 2005) P. Mika: "Flink : Semantic Web Technology for the Extraction and Analysis of Social Networks, Department of Computer Science". Vrije Universiteit Amsterdam (VUA), 2005.

(Milojicic, 1999) D. Milojicic, F. Douglis, R. Wheeler: "Mobility : processes, computers and agents". Addison-Wesley, 1999.

(Mlungisi, 2008) D. Mlungisi: "Agents, Agent architectures and Multi-agent systems". Master of science, Ehlers, E.M., Oosthuizen, O.L., Johnnesburg, 143 p. 2008.

(Modi, 2002) T. Modi: "WSIL: Do we need another Web Services Specification? Explaining the difference between UDDI", 2002.

(Motta, 2003) E. Motta, J. Domingue, L. Cabral, M. Gaspari: "IRS-2: A framework and infrastructure for semantic Web services". In International Semantic Web Conference, pages 306–318, 2003.

(Neches, 1991) R. Neches, R. Fikes, T. Finin, T. Gruber, R. Patil, T. Senator, W. Swartout: "Enabling technology for knowledge sharing". AI Magazine, 12(3) :36-56, 1991.

(Newcomere, 2004) E. Newcomere: "Understanding Web Services Xml WSDL SOAP And UDDI". Edition O'Reilly, 2004.

(Nguyen, 1998) A. Nguyen, I. Stewart, X. Yang: "A Mobile Agent Model: Applications for ECommerce". Proceedings of Seventh Australian World Wide Web Conference, Brisbane Australia, pp. 21 - 25. 1998.

(Nguyen, 2005) T. X. Nguyen, R. Kowalczyk: "Ws2jade : Integrating Web service with jade agents". Technical Report, SOCAB05. 2005.

(Nogues, 2010) L. Nogues, I. Le Berre : "Fiche technique n° 4a : Rédaction et publication d'une fiche de métadonnées". MIMEL – DREAL de Basse-Normandie, juin 2010. http://www.donnees.normandie.developpement-durable.gouv.fr/pdf/MIMEL/FT4a_MIMEL.pdf

(Ohsuga, 1997) A. Ohsuga, Y. Nagai, Y. Irie, M. Hattori, S. Honiden: "PLANGENT: an approach to making mobile agents intelligent". IEEE Internet Computing, 1(4): 50–57, July/August 1997.

(Palathingal, 2004) P. Palathingal, S. Chandra: "Agent approach for service discovery and utilization". In HICSS, 2004.

(Parasuraman, 1988) A. Parasuraman, A. Z. Valarie, L. B. Leonard: "SERVQUAL : A Multiple- Item Scale for Measuring Consumer Perceptions of Service Quality". Journal of Retailing, 64, 1, 12-40, 1988.

(Perret, 1997) S. Perret : "Agents mobiles pour l'accès nomade à l'information répartie dans les réseaux de grande envergure". Thèse de Doctorat. Université Joseph Fourier - Grenoble I, 19 novembre 1997.

(Picco, 1998) G. P. Picco: "Understanding, Evaluating, Formalizing, and Exploiting Code Mobility". PhD thesis, Politecnico di Torino, Italy, February 1998.

(Picco, 1999) G. P. Picco, A. L. Murphy, G. C. Roman: "Lime: Linda meets mobility". In Proceedings of the 21st International Conference on Software Engineering (ICSE'99), pages 368–377, Los Angeles, California, USA, May 1999. http ://www.cs.rochester.edu/u/murphy/papers/icse99.pdf.

(Pierce, 1993) B. C. Pierce, D, Sangiorgi: "Typing and subtyping for mobile processes". In Proceedings 8th IEEE Logics in Computer Science, pages 376–385, Montreal, Canada, 1993.

(Pokraev, 2003) S. Pokraev, J. Koolwaaij, M. Wibbels: "Extending UDDI with context-aware features based on semantic service descriptions". In ICWS, pages 184–190, 2003.

(Ponge, 2008) J. Ponge: "Model Based Analysis of Time-aware Web Services Interactions". Thèse de Doctorat de l'Université Blaise Pascal - Clermont-Ferrand II, dans le cadre de l'Ecole Doctorale des Sciences pour l'Ingénieur, France, 2008.

(Pretschner, 1999) A. Pretschner, S. Gauch : "Ontology Based Personalized Search". In Proceedings of the 11th IEEE International Conference on Tools with Artificial Intelligence (ICTAI), November 1999.

(Rampacek, 2006) S. Rampacek : "Sémantique, interactions et langages de description des services Web complexes". Thèse de Doctorat, 10 Novembre 2006.

(Ratnasamy, 2001) S. Ratnasamy, P. Francis, M. Handley, R. Karp, S. Shenker: "A scalable content addressable network", In Proc of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications, ACM SIGCOMM, California, USA, 2001, pp. 161-172.

(Rezeg, 2010) K. Rezeg : " Découverte des services dans les systèmes d'information géographiques répartis sous réseau ad-hoc". Thèse de doctorat, Université de Biskra, 2010.

(Rouvrais, 2002) S. Rouvrais : "Utilisation d'Agents Mobiles pour la Construction de Services Distribués". Thèse, Université de Rennes 1, 2002.

(Sacramento, 2009) E. R. Sacramento : "Automatic Discovery and Composition of OGC Web Services". Thèse de doctorat, université fédérale de Cesra, Brésil, 2009.

(Sahai, 1998) A. Sahai, C. Morin: "Mobile agents for enabling mobile user aware applications". Proceedings of the 2nd International Conference on Autonomous Agents (Agents'98), pages 205–211, New York, May 9–13, 1998. ACM Press.

(Sahin, 2005) O. D. Sahin, C. E. Gerede, D. Agrawal, A. El Abbadi, O. Ibarra, J. Su: "SPiDeR: P2P-Based Web Service Discovery", In the Proc of Third International Conference on Service-Oriented Computing (ICSOC), Amsterdam, The Netherlands, 2005, pp. 157-169.

(Salton, 1971) G. Salton: "The SMART Retrieval System: Experiments in Automatic Document Processing". Prentice-Hall Inc, NJ.1971.

(Satoh, 2002) I. Satoh: "Physical mobility and logical mobility in ubiquitous computing environments". Lecture Notes in Computer Science, 2535:186–202, 2002.

(Satoh, 2010) I. Satoh: "Mobile Agent-based Context-aware Services". Journal of Universal Computer Science, vol. 16, no. 15, pp: 1929-1952, 2010.

(Scioscia, 2014) F. Scioscia, M. Ruta, G. Loseto, F. Gramegna, S. Ieva, A. Pinto, E. D. Sciascio: "A Mobile Matchmaker for the Ubiquitous Semantic Web". International Journal on Semantic Web and Information Systems, 10(4), 77-100, October-December 2014. www.igi-global.com/article/a-mobile-matchmaker-for-theubiquitous-semantic-Web/129763?camid=4v1

(Seghrouchni, 2004) A. E. F. Seghrouchni, S. Haddad, T. Melitti, A. Suna : "Interopérabilité des systèmes multi-agents à l'aide des services Web ". Les Journées Francophones sur les Systèmes Multi- Agents, Paris, France, pp. 91-104, 2004.

(Sheth, 2003) A. Sheth: "Semantic Meta Data for enterprise information integration". DM Review magazine, July 2003.

(Shuping, 2003) R. Shuping: "A Model for Web Services Discovery With QoS". ACM, 2003.

(Sivashanmugam, 2004) K. Sivashanmugam, K. Verma, R. Mulye, Z. Zhong, A. Sheth: "Speed-R: Semantic P2P environment for diverse Web Service registries". http://webster.cs.uga.edu/~mulye/SemEnt/Speed-R.html. (2004).

(SOAP, 2004) W3C World Wide Web Consortium: "SOAP Version 1.2 Part 1:Messaging Framework", (2004). http://www.w3.org/TR/soap12-part1/.

(Smith, 1994) D. C. Smith, A. Cypher, J. Spohrer: "Kidsim: programming agents without a programming language". Communications of the ACM, Vol. 37, No. 7, pp. 54–67. 1994.

(Somlo, 2003) G. L. Somlo, A. E. Howe: "Using Web Helper Agent Profiles in Query Generation International Conference on Autonomous Agents". Proceedings of the 2international joint conference on Autonomous agents and multi agent systems Melbourne, Australia Web technologies. 2003.

(Sycara, 1998a) K. P. Sycara: "The many faces of agents". Al Magazine, Vol. 19, No. 2, pp. 11-12. 1998.

(Sycara, 1998b) K. P. Sycara : "Miltu-agent system". Al Magazine, Vol. 19, No. 2, pp. 79–92. 1998.

(Tamrout, 2006) R. B. Tamrout : "JREGISTRE: Un Registre UDDI Extensible". Mémoire de la maitrise en informatique, Université du Québec à Montréal, CANADA, 2006.

(Tan, 2006) K. L. Tan, C. S. Lee, H. N. Chua: "Models to Customise Web Service Discovery Result using Static and Dynamic Parameters", Proceedings of world academy of sciences engineering, Volume 12, 2006.

(Tang, 2014) M. Tang, Z. Zheng, L. Chen, J. Liu, B. Cao, Z. You: "A Trust-Aware Search Engine for Complex Service Computing". International Journal of Web Services Research, 11(1), 57-75, January-March 2014. www.igi-global.com/article/a-trust-aware-search-engine-forcomplex-service-computing/110874?camid=4v1

(Troispoux, 2007) G. Troispoux : "La qualité des données géographiques au service des utilisateurs". CERTU (centre d'études sur les réseaux, les transports,

l'urbanisme et les constructions publiques, ministère de l'écologie, du développement et du management durables), 2007.

(Ulrich, 1999) A. Ulrich, H. König: "Architectures for testing distributed systems". In Testing of Communicating Systems: Method and Applications, IFIP TC6 12th International Workshop on Testing Communicating Systems (IWTCS), IFIP Conference Proceedings, pages 93–108, Budapest, Hungary, September 1-3, 1999.

(Vadivelou, 2011) G. Vadivelou, E. IIavarasan, S. Prasanna: "Algorithm for Web Service Composition using Multi-Agents". International Journal of Computer Applications (0975 – 8887),Volume 13– No.8, January 2011.

(Valarie, 2000) A. Z. Valarie, A. Parasuraman, A. Malhotra: "A Conceptual Framework for Understanding e-Service Quality: Implications for Future Research and Managerial Practice". MSI Working Paper Series, Report No. 00-115, 2000.

(Verma, 2005) K. Verma, K. Sivashanmugam, A. Sheth, A. Patil, S. Oundhakar, J. Miller: "METEOR-S WSDI: A Scalable Infrastructure of Registries for Semantic Publication and Discovery of Web Services", Journal of Information Technology and Management, Special Issue on Universal Global Integration, 6(1):17–39, 2005.

(Vigna, 2004) G. Vigna: "Mobile agents: Ten reasons for failure". In 5th IEEE International Conference on Mobile Data Management (MDM 2004), pages 298–299. IEEE Computer Society, 19-22 January 2004.

(Vogel, 1995) A. Vogel, B. Kerhervé, G. V. Bochmann, J. Gecsei: "Distributed multimedia and qos: A survey". IEEE MultiMedia, 2(2):10–19, 1995.

(Vu, 2005) L. H. Vu, M. Hauswirth, K. Aberer: "Towards P2P-based semantic Web service discovery with Qos support". In Business Process Management Workshops, (pp. 18–31), 2005.

(WSIL) WSIL : "Web Services Inspection Language, spécification contribuée par IBM et Microsoft", disponible à http://www.ibm.com/developerworks/library/specification/ws-wsilspec/

(W3C, 2004) W3C Working Group. W3c working group note, February 2004. http://www.w3.org/TR/ws-gloss/.

(Xining, 2010) X. Li, J. Lin, L. Li : "On the Design of a Mobile Agent Environment for Context-aware M-commerce". Computer Science and Information Technology (ICCSIT), 2010.

(Yann, 2006) A. Yann : "Conception et exploitation d'une base de méta données de traitements informatiques, représentation opérationnelle des connaissances d'expert, application au domaine géographique". Thèse de doctorat, IGN FRANCE, Université de CEAN, 2006.

(Zemirli, 2008) W. N. Zemirli : "Modèle d'accès personnalisé à l'information basé sur les Diagrammes d'Influence intégrant un profil utilisateur évolutif". Thèse de doctorat, l'Université Paul Sabatier de Toulouse III, 12 Juin 2008.

(Zgaya, 2007) H. Zgaya: "Conception et optimisation distribuée d'un système d'information d'aide à la mobilité urbaine : Une approche multi-agent pour la recherche et la composition des services liés au transport". Doctorat délivré par l'Ecole Centrale de Lille soutenue le 6 juillet 2007.