Faculté de Technologie — Département d'Informatique

Découverte des Services Web : Approche basée sur les préférences des utilisateurs

THESE

DOCTORAT EN SCIENCES

(Spécialité Informatique)

Option : Interopérabilité et Intégration des Systèmes d'Information dans le Web

Présentée par

Houda EL BOUHISSI

Soutenue devant le jury composé de :

Président :	**Dr Sidi-Mohamed BENSLIMANE**	**Professeur**	UDL-SBA
Directeur de thèse :	**Dr Mimoun MALKI**	**Professeur**	UDL-SBA
Examinateur :	**Dr Yahia LEBBAH**	**Professeur**	UNIV-ORAN
Examinateur :	**Dr Abdelmalek AMINE**	**MCA**	UNIV-SAIDA
Examinateur :	**Dr Djelloul BOUCHIHA**	**MCA**	CU-NAAMA
Examinateur :	**Dr Djamel AMAR BENSABER**	**MCA**	UDL-SBA

Evolutionary Engineering and Distributed Information Systems Laboratory

Remerciements

Le travail que j'ai réalisé au cours de ma thèse n'aurait certainement pas été possible sans l'aide, les encouragements et les conseils de nombreuses personnes qui y ont été impliquées de près ou de loin. C'est pourquoi, je tiens à leur exprimer par ces quelques lignes toute ma reconnaissance, en particulier :

Je tiens à exprimer mes remerciements et toute ma gratitude au professeur Mimoun MALKI, pour avoir accepté de m'encadrer pendant cette thèse et aussi pendant le mémoire de magister, ainsi que pour la confiance qu'il m'a accordée. Ses critiques constructives ont été d'un apport déterminant pour l'aboutissement de cette thèse.

Je remercie sincèrement Messieurs le président et les membres de jury qui ont accepté d'examiner mon travail : Dr Sidi Mohamed BENSLIMANE, Professeur à l'université Djillali Liabés de Sidi-bel-Abbés d'avoir accepté de présider le jury de ma soutenance, Dr Yahiya LEBBAH, Professeur à l'université d'Oran pour sa disponibilté, Dr Abdelmalek Amine, Maître de Conférences à l'université de Saida, pour ses précieuses remarques, Dr Djelloul BOUCHIHA, Maître de Conférences au centre universitaire de Naama, pour sa disponibilité et ses encouragements et Dr Djamal AMAR BENSABER, Maître de Conférences à l'université Djillali Liabés de Sidi-Bel-Abbés pour les débats fructueux que j'avais toujours avec lui.

Je tiens à remercier tous les membres de ma famille de leur soutien continu et plus particulièrement ma mère, qui m'a toujours aidée à aller plus loin et qui m'a motivée.

Je remercie chaleureusement tous mes amis et plus particulièrement Djamila, Farid et Mahmoud pour leurs soutien et encouragement continus.

Enfin, je réserve une pensée particulière à mon ami Amar pour avoir toujours su trouver les mots qu'il faut dans les meilleurs comme dans les pires moments et pour m'avoir soutenu, encouragé et supporté durant ces années de thèse, en particulier pendant ma transition du secteur de la formation professionnelle à l'université. Dieu merci de m'avoir accordé ton aide à travers toutes ces personnes.

Houda EL BOUHISSI

Dédicaces

A la mémoire de mon défunt Père, aucune dédicace ne saurait exprimer l'amour, l'estime, le dévouement et le respect que j'ai toujours eu pour toi, rien au monde ne vaut les efforts fournis jour et nuit pour mon éducation et mon bien être, que Dieu le tout puissant t'accorde sa grâce et sa miséricorde et t'accueille dans son vaste paradis.

Ce travail est le fruit de tes sacrifices que tu as consentis pour mon éducation et ma formation, merci baba.

A ma Mère, affable, honorable, aimable : tu représentes pour moi le symbole de la bonté par excellence, la source de tendresse et l'exemple du dévouement qui n'a pas cessé de m'encourager et de prier pour moi. Ta prière et ta bénédiction m'ont été d'un grand secours pour mener à bien mes études, je te dédie ce travail en témoignage de mon profond amour. Puisse Dieu, le tout puissant, te préserver et t'accorder santé, longue vie et bonheur.

A mes frères et soeurs.

A ma sœur Oum-Mahmoud, son mari Mokhtar et ses petits enfants Aridj et Mahmoud Moataz Billah.

A toutes ces personnes, les mots ne suffisent guère pour exprimer l'attachement, l'amour et l'affection que je porte pour vous !.

Houda EL BOUHISSI

Résumé

Il y a eu un intérêt croissant pour les Services Web Sémantiques (SWS) comme une solution proposée pour faciliter la découverte automatique, la composition et le déploiement des Services Web. Pour profiter pleinement des avantages de cette technologie, un processus de ré-ingénierie des Services Web vers les Services Web Sémantiques est nécessaire. Malgré son importance, il n'y a que peu de travaux qui ont traité le problème de ré-ingénierie des Services Web vers les Services Web Sémantiques. La plupart de ces travaux n'ont pas traité les particularités des Services Web. Aussi la découverte des Services Web est l'un des défis majeurs du paradigme émergent SOA (Service Oriented Architecture). La majorité des travaux existants proposent une recherche à base de mots clés. Cependant, ce type d'approche ne permet pas de récupérer certains Services Web susceptibles de satisfaire les besoins des utilisateurs.

L'objectif de cette thèse est double : d'une part, proposer une ré-ingénierie de Services Web vers les Services Web Sémantiques, et d'autre part, proposer une approche de découverte des Services Web basée sur les préférences des utilisateurs. Notre approche de découverte positionne la découverte des services dans une perspective centrée buts("Goals") dans laquelle les services sont décrits en termes d'exigences qu'ils permettent de satisfaire. Nos contributions utilisent l'ontologie de service WSMO comme modèle sémantique de spécification des Services Web.

Nous avons abordé ces problématiques à la fois d'un point de vue théorique et pratique. En plus de la proposition d'une ré-ingénierie de services et de la découverte des services pertinents, nous avons implémenté ces deux concepts en développant deux prototypes. WSDL2WSMO-LITE permet de convertir un fichier WSDL en des éléments de l'ontologie de services WSMO-LITE. Pour la découverte, nous avons développé un cadre de travail complet pour la recherche des Services Web susceptibles de satisfaire les exigences des utilisateurs. Enfin, pour valider notre démarche, nous avons appliqué nos approches à des cas d'études réels.

Mots Clés : Ré-ingénierie, Découverte, Ontologie, WSMO, WSMO-Lite, WSDL, Service Web, Service Web Sémantique.

Abstract

There has been an increasing interest in Semantic Web services (SWS) as a proposed solution to facilitate automatic discovery, composition and deployment of existing syntactic Web services. To enjoy the full benefits of this technology, a re-engineering process of Web Services to the Semantic Web Services is required. Despite its importance, there are only few works that have addressed the issue of re-engineering to the Web Services Semantic Web Services. Most of these works did not address the particularities of Web Services. Also the discovery of Web Services is one of the major challenges of the emerging paradigm SOA (Service Oriented Architecture). Most existing works offer a keyword-based discovery. However, this approach does not allow to recover some Web services to best meet the user needs.

The objective of this thesis is twofold : on the one hand, to propose Web Services re-engineering to the Semantic Web Services, and on the other hand, to provide a Web services discovery approach based on user preferences. Our discovery approach positions the discovery of services in a perspective centered goals ("Goals") in which the services are described in terms of requirements they can satisfy. Our contributions use the WSMO service ontology as a semantic specification model .

We addressed these issues from both a theoretical and practical perspective. In addition to the proposal for a re-engineering of services and the discovery of the relevant services, we have implemented these concepts by developing two prototypes. WSDL2WSMO-LITE converts a WSDL file to WSMO-Lite elements. For the discovery, we developed a comprehensive framework for Web services research to best meet user requirements. Finally, to validate our proposal, we applied our approach to real case studies.

Keywords : Reengineering, Discovery, Ontology, WSMO, WSMO-Lite, WSDL, Web Service, Semantic Web Service.

Liste des abréviations

BPEL	Business Process Execution Language
HTML	HyperText Markup Language
HTTP	Hyper Text Transfer Protocol
OWL	Ontology Web Language
OWL-S	Web Ontology Language for Services Web
RDF	Resource Description Framework
RDFS	RDF Schema
RPC	Remote Procedure Call
SAWSDL	Semantic Annotations for WSDL and XML Schema
SOA	Service Oriented Architecture
SOAP	Simple Object Access Protocol
UDDI	Universal Description, Discovery and Integration
URI	Uniform Resource Identifier
W3C	World Wide Web Consortium
WSDL	Web Service Description Language
WSDL-S	Web Service Description Language-Semantic
WSML	Web Service Modeling Language
WSMO	Web Service Modeling Ontology
WSMO-Lite	Lightweight Semantic Descriptions for Services on the Web
WWW	World Wide Web
XML	eXtensible Markup Language

Table des figures

Liste des tableaux

Listings

Liste des Algorithmes

Table des matières

Chapitre

1 Introduction et Motivation

Sommaire

1.1. Introduction

La technologie des Services Web et l'idée de l'architecture orientée service (SOA) pour les applications Web, implémentation modulaire des systèmes distribués et complexes, semble acquérir un succès énorme. Dans peu d'années simplement, l'approche orientée services n'a pas acquis uniquement un intérêt considérable dans les recherches en informatique, mais aussi une grande attention avec une unanimité unique de tous les partenaires dans l'industrie IT, tels que IBM, Microsoft et Hewlet Packard.

Les systèmes distribués ont dominé de plus en plus plusieurs domaines dans la vie quotidienne, et les logiciels eux-mêmes deviennent de plus en plus puissants. Par conséquent, la préoccupation majeure est comment structurer des systèmes modulaires et comment atteindre l'interopérabilité entre les parties hétérogènes. La réalisation effective et efficace de tels systèmes interopérables, modulaires et à grande échelle est facilitée par les Services Web et SOA car ils fournissent une architecture standardisée. Des systèmes modulaires, faiblement couplés, pour créer de nouvelles fonctionnalités à partir de blocs de programmes déjà existants et pour permettre notamment la communication entre les éléments hétérogènes.

1.2. Contexte et problématique

Le domaine des Services Web Sémantiques est en plein expansion et très effervescent et loin des différentes discussions techniques qui se font autour du sujet.

Bien qu'il existe à l'heure actuelle certains efforts dans le domaine de l'intégration ou de l'interopérabilité sémantique (par exemple OWL-S, WSMO et bien d'autres qu'on verra en détail au chapitre 3), il n'en demeure pas moins que la plupart de ces initiatives sont en cours de développement et/ou manquent de maturité. Par conséquent, l'intégration sémantique constitue une problématique toujours ouverte, aussi bien dans le contexte de l'intégration intra-entreprise que dans le contexte de grandes entreprises caractérisées par un environnement dynamique.

Cependant, les technologies proposées n'exploitent pas les informations contenues dans les interfaces du service telles que le WSDL et exigent l'intervention d'un utilisateur humain et des experts du domaine pour l'exécution des tâches telles que la découverte, la sélection, l'invocation et la composition.

Toutes les approches de conception des Services Web Sémantiques souffrent plus ou moins du manque de méthodologies, du manque d'architectures flexibles pouvant décrire efficacement la sémantique liée aux applications, et aussi du manque de maturité ou parfois même de l'absence de mécanismes de description et de médiation efficaces.

En conséquence du succès des architectures orientées services au cours de ces dernières années, un nombre important de Services Web est devenu disponible en ligne. Avec cette augmentation, plusieurs problèmes ont été soulevés, notamment ceux liés à la découverte, la disponibilité, la qualité de services, etc.

Les techniques de découverte de services traditionnelles peuvent être inadécquates dans des contextes réels (un grand nombre de services et d'annuaires, manque d'expressivité des descriptions de services, etc.).

La découverte des Services Web représente un axe de recherche émergent. Au début, la découverte se faisait au niveau du registre UDDI, elle était basée essentiellement sur la recherche syntaxique des descriptions WSDL des Services Web. Mais avec le développement des technologies du Web sémantique, les techniques de découverte sont devenues essentiellement sémantiques.

1.3. Objectifs et Contributions

Ce travail de recherche a pour but principal de proposer de nouvelles approches pour la ré-ingénierie et à la découverte des Services Web basée sur les préférences des utilisateurs.

L'approche de ré-ingénierie des Services Web vers les Services Web Sémantiques permet la spécification de l'ontologie WSMO, elle se base sur l'analyse en amont d'un document WSDL d'un Service Web donné, et des ontologies de domaine et renvoie en sortie des spécifications incomplètes de l'ontologie WSMO.

L'ontologie WSMO a été proposée ces dernières années comme un cadre de travail pour l'automatisation totale/partielle des tâches telles que : la découverte, la sélection, la composition, la médiation, l'exécution et le monitoring impliquées à la fois dans l'intégration intra et inter-entreprise des Services Web.

WSMO est reconnu parmi les technologies qui sont en cours d'étude. Il complète les standards syntaxiques des Services Web en fournissant un modèle conceptuel et un langage pour l'annotation sémantique décrivant tous les aspects des services généraux qui sont accessibles à travers une interface de Service Web.

L'approche de ré-ingénierie est supportée par un outil logiciel et se déroule en deux principales étapes : (1) Une étape de rétro-ingénierie pour l'identification des informations petinentes et (2) Une étape d'ingénierie pour la génération des Services Web Sémantiques. Ces deux étapes comportent plusieurs phases :

- Une première phase de classification et catégorization du Service Web , en vue de trouver l'ontologie correspondant parfaitement au domaine du Service Web.
- Une deuxième phase d'annotation du document WSDL en utilisant l'ontologie de domaine sélectionnée durant la première phase.
- Une troisième phase d'extraction des informations pertinentes du document WSDL annoté.
- La quatrième phase consiste à générer les fichiers de sortie qui représentent des éléments du WSMO(Ontologie et Service Web) selon la syntaxe du langage WSML.
- Enfin, la dernière phase représente la phase de validation qui permet de vérifier et corriger les éléments produits relativement à la syntaxe du langage WSML.

La deuxième contribution de cette thèse est la proposition d'une approche de découverte des Services Web basée sur les préférences des utilisateurs. Toujours, dans le cadre de l'ontologie WSMO, l'approche de découverte consiste à formuler un but("Goal") à partir d'une requête

fournie de l'utilisateur à travers une page HTML et chercher les Services Web susceptibles de satisfaire ce but. La système mis en oeuvre reçoit en entrée des données fournies par l'utilisateur et retourne en sortie un ou plusieurs Services Web correspondants aux besoins des utilisateurs.

L'approche de découverte comprend trois principales étapes : (1) Une catégorisation des services basée sur la sémantique ; (2) Une Découverte sémantique des Services Web pertinents et (3) Une sélection des meilleurs Services. La première étape concerne l'évolution et la maintenance du système et la mise à jour de l'ensemble des référentiels. Les deux dernières étapes concernent le processus de découverte dans son ensemble : découverte des Services Web pertinents et la sélection du meilleur Service susceptible de répondre aux besoins des utilisateurs.

1.4. Méthodologie de travail

Dans ces grandes lignes, la démarche adoptée pour notre travail est guidée par de nombreuses questions issues des préoccupations de la communauté des Services Web Sémantiques. Etant donnée que la problématique de l'intégration est complexe, nous nous sommes forcés tout au long de cette thèse de prendre suffisamment de recul afin de proposer un cadre méthodologique relativement global et suffisamment complet afin de mieux aider et mieux guider les utilisateurs dans leur processus.

Notre démarche de travail repose plus précisément sur les étapes suivantes :

- (i) étape de recherche et d'analyse : qui établit un état de l'art des différentes technologies proposées dans le cadre de la ré-ingénierie et la découverte des Services Web avec une comparaison des avantages et inconvénients de chaque approche proposée.
- (ii) étape d'identification du problème et la solution : qui permet de définir la problématique et la solution proposée en vigueur.
- (iii) étape d'implémentation et d'expérimentation des systèmes proposés : qui met en évidence le système proposé, son fonctionnement et son intérêt, accompagnée d'une étude de cas pour la validation.

1.5. Organisation de la thèse

Le reste de la thèse est organisé comme suit :

- **Chapitre 2** : Ce chapitre aborde un volet relatif à l'évolution du web et les deux technologies qui en découlent, à savoir les Services Web et le Web Sémantique. Notre recherche étant en lien avec le Web Sémantique, nous présentons ce en quoi il consiste. Nous verrons par la suite la notion d'ontologie.

- **Chapitre 3** : Le troisième chapitre met en relief de manière assez sommaire le domaine des Services Web Sémantiques, étant donné qu'il constitue le noyau de notre tâche tentant ainsi de couvrir la majorité des approches connexes à nos travaux de recherche.
- **Chapitre 4** : Le quatrième chapitre porte sur les domaines de la ré-ingénierie et de la découverte des Services et les différents travaux relatifs à ces domaines. Ainsi, une étude comparative des différentes approches sera présentée.
- **Chapitre 5** : Ce chapitre présente en détail nos contributions utilisées au cours de nos recherches. Nous présentons aussi les différentes étapes de nos approches avec des exemples explicatifs.
- **Chapitre 6** : Le sixième chapitre couvre l'expérimentation de nos approches. Il présente les différents aspects liés à l'implémentation des prototypes que nous avons développés et effectués dans le cadre du déploiement de nos prototypes sur des cas d'études réels.
- **Chapitre 7** : Enfin, cette thèse est clôturée par le chapitre sept qui donne les conclusions et perspectives de ce travail. Ce chapitre propose une synthèse et un bilan du travail effectué durant cette thèse et un ensemble de perspectives liées notamment à la poursuite de ce travail ainsi qu'aux nouveaux thèmes de recherche qui nous paraissent les plus pertinents.

Chapitre

2 BackGround

Sommaire

2.1. Introduction

Le Web était un système d'hypertextes distribués permettant l'accès à des sources de données hétérogènes en utilisant un ensemble de standards pour : (1) décrire les données (HTML [1]), (2) localiser les documents sur le réseau (URI [2]) et (3) accéder à ces documents (HTTP [3]). Du fait de la simplicité du HTML qui permet la création de pages Web avec un minimum d'efforts, le Web a bénéficié d'un grand succès dans un temps relativement court. Et depuis lors, le nombre de sites Web et de serveurs Web a rapidement augmenté.

Suite à cette explosion du Web, la recherche de la bonne information devient de plus en plus difficile même avec l'appui des moteurs de recherche tels que "Google" et "Bing". La technique de recherche courante est fondée sur les mots-clés. Cette technique fournit comme résultat une

1. HTML : Hypertext Markup Language.
2. URI : Uniform Resource Identifier.
3. HTTP : HyperText Transfer Protocol.

quantité assez importante d'informations et de liens au-delà des préoccupations des utilisateurs (tous les sites qui mentionnent le(s) mot(s) clé(s) donné(s) sont répertoriés(s)).

De ce fait, le taux de rappel est assez fort mais la précision est faible parce que seulement peu de pages recherchées contiennent l'information dont l'utilisateur en a besoin. En conséquence, il est difficile de répondre aux requêtes complexes parce que : (1) une telle information spécifique n'est pas souvent disponible en tant que telle sur le Web ou (2) si elle est disponible, elle peut prendre différentes formes syntaxiques.

Comme solution à ces insuffisances, le Web a subi deux changements révolutionnaires comme présentés dans la figure 2.1.

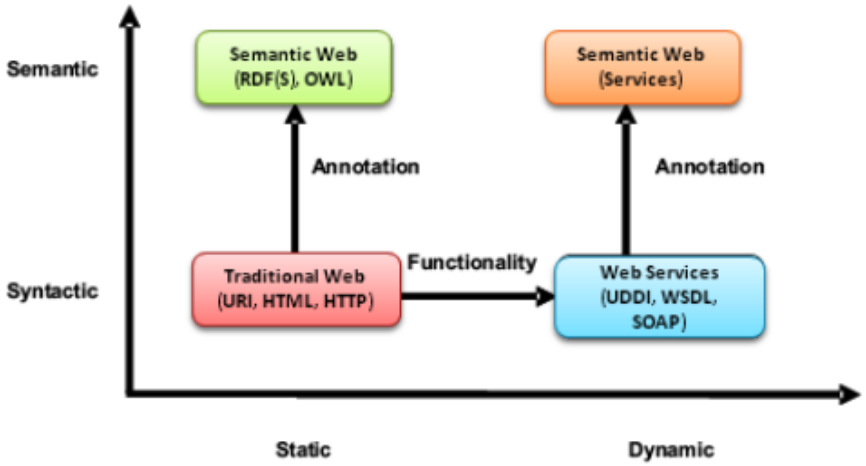


Fig 2.1 – Evolution du Web

Premièrement, la communauté du Web vise une extension sémantique du Web actuel syntaxique en augmentant les informations existantes sur le Web avec des descriptions formelles de leurs significations en utilisant des descriptions à base de logiques. Cette annotation sémantique sera appréhendable par la machine et permettra donc un accès et une intégration plus faciles de la quantité énorme d'information disponible, par rapport à ce qui a été réalisé avec la recherche basée sur les mots-clés.

En second lieu, le Web change d'une collection de pages statiques en un environnement dynamique avec l'arrivée de la technologie des Services Web qui rend les composants logiciels accessibles via des protocoles Web.

La technologie des Services Web Sémantiques a donc vu le jour à travers ces deux dernières technologies en appliquant les techniques du Web sémantique sur les Services Web.

Dans le reste de ce chapitre, nous allons présenter brièvement ces deux technologies du Web, à savoir, les Services Web et le Web Sémantique.

2.2. Les Services Web

Ces dernières années ont vu l'émergence d'architectures logicielles fondées sur les services qui visent à mettre en place des processus métier performants ainsi que des systèmes d'information constitués de services applicatifs indépendants et interconnectés. Ces architectures sont connues sous le nom d'Architectures Orientées Services (SOA : Service Oriented Architecture) [Thoma, 2005]. Elles facilitent l'exposition, l'interconnexion et la réutilisation d'applications à base de services. Ainsi, de nouveaux services peuvent être créés à partir d'une infrastructure informatique de systèmes déjà existante. Ces services peuvent être utilisés par des processus métier ou par des clients dans différentes applications.

Les Services Web sont la réalisation la plus importante d'une architecture SOA. Ce sont des applications auto descriptives, modulaires et faiblement couplées fournissant un modèle simple de programmation et de déploiement d'applications. Ils reposent principalement sur des technologies basées sur "SOAP" pour la structure et le contenu de messages échangés entre services, "WSDL" pour la description des services, "UDDI" pour la découverte des services et "BPEL" pour leur composition.

Le W3C [4] définit un Service Web comme étant une application ou un composant logiciel vérifiant les propriétés suivantes [Booth *et al.*, 2004] :

- Il est identifié par une URI (Uniform Ressource Identifier) ;
- Ses interfaces et ses liens peuvent être décrits à base du 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 et en utilisant des protocoles Internet.

2.3. Infrastructure des Services Web

Bien que le web soit constitué de plateformes totalement hétérogènes où les intérêts des différents acteurs du marché s'entremêlent, ceci ne l'a pas empêché de se developper et d'être universel. Ce succès est dû essentiellement à l'édition d'un ensemble de standards ouverts, dont les plus connus est le protocole HTTP. Le HTTP offre un mécanisme d'échange de données de toute sorte quelque soit la nature des plateformes impliquées.

Le cycle de vie d'un Service Web se déroule de la façon suivante : une fois créé, le service est déployé sur le réseau (local ou Internet). Puis un utilisateur ayant des besoins spécifiques

4. World Wide Web Consortium : Organisme de normalisation des standards du Web.

va rechercher un service correspondant à ses besoins à l'aide d'un annuaire spécialisé. Enfin, une fois le service trouvé, l'utilisateur va invoquer le service : une communication va être mise en place entre l'utilisateur et le Service Web. Ce cycle de vie, représenté par la figure 2.2 [Chauvet, 2002], fait appel à trois grandes technologies : SOAP, WSDL et UDDI.

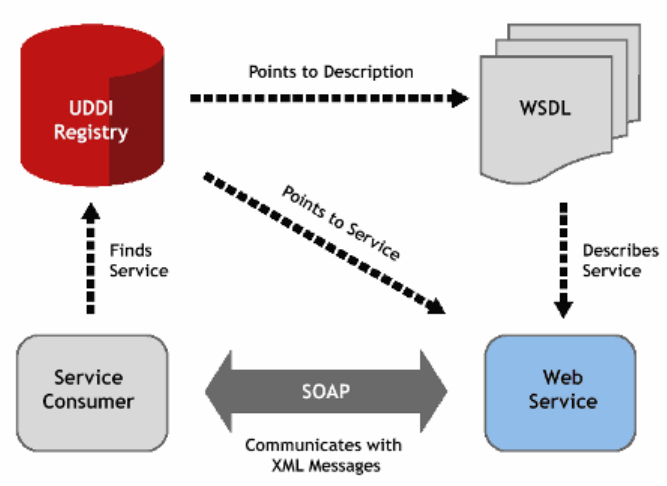


Fig 2.2 – Infrastructure des Services Web.

- SOAP (Simple Object Access Protocol) : assure la communication avec et inter-Services Web ;
- WSDL (Web Services Description Langage) : offre un schéma formel de description des Services Web ;
- UDDI (Universal Description, Discovery and Integration) : offre une manière uniforme de définir des registres des Services Web et un schéma uniformément extensible de descriptions des Services Web.

2.3.1 SOAP

SOAP est un protocole pour l'échange d'informations structurées avec les Services Web, recommandé par le W3C. Il est basé sur le XML pour le format des messages.

SOAP forme la couche inférieure de la pile des protocoles des Services Web, fournissant un "framework" pour l'échange de messages sur lequel les Services Web peuvent se baser.

Un message SOAP (cf. Fig 2.3) est un document XML composé d'un élément enveloppe. L'élément enveloppe contient à son tour deux éléments fils, une entête (facultative)et un corps.

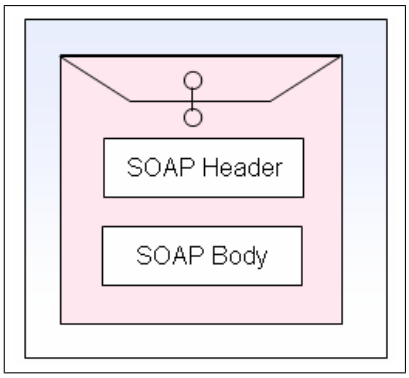


Fig 2.3 – Structure d'un message SOAP.

L'entête du message obéit à un schéma spécifique et elle est destinée à contenir des informations nécessaires aux applications pour interpréter la charge utile. La charge utile est contenue dans un élément "body". Elle obéit à une grammaire extensible. L'élément body peut contenir n'importe quelle structuration XML à condition qu'elle obéisse à une grammaire dont l'espace de noms est référencé.

Le protocole SOAP définit un modèle de liaison qui assure l'interopérabilité au niveau applicatif entre des applications hétérogènes. La vision des Services Web s'est concrétisée avec l'apparition de SOAP, en permettant à deux applications appartenant à des environnements technologiques hétérogènes d'échanger des données structurées.

2.3.2 WSDL

Le Web Service Description Language (WSDL) [Chinnici *et al.*, 2007] est devenu le standard actuel pour la définition des services. Par le biais d'un document XML, il décrit un service à travers une interface présentant un ensemble d'opérations et leurs paramètres d'entrée et de sortie respectifs. L'interface WSDL décrit la fonctionnalité accomplie par le service (ce que fait le service) mais il ne décrit pas les modalités d'accomplissement de cette fonctionnalité (comment le service le fait).

Un document WSDL est un fichier XML constitué de cinq éléments principaux : <types>, <message>, <portType>, <binding> et <service>. La figure 2.4 illustre la structure d'une interface décrite en WSDL avec les éléments suivants :

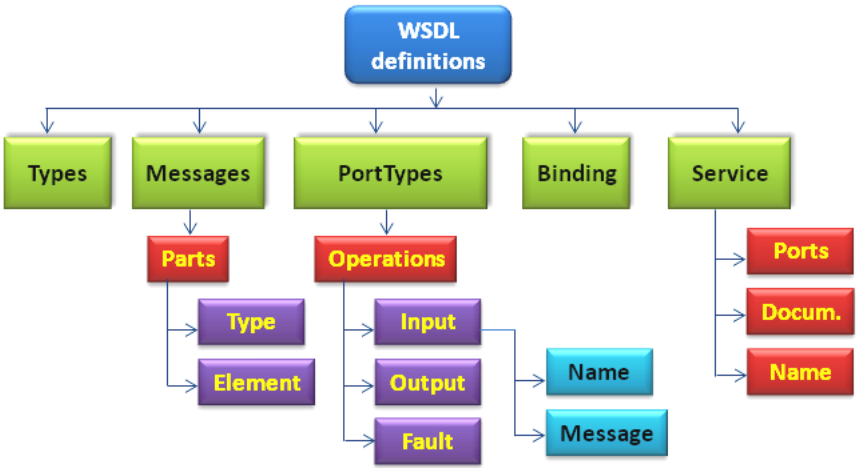


Fig 2.4 – Structure d'un document WSDL.

- <types> : est un conteneur définissant les données figurant dans les messages échangés par le service. Il définit, à travers un schéma XML, les types de ces données. Les données peuvent être simples ou complexes.
- <message> : définit les messages échangés par le service. Il est composé d'un ensemble d'éléments "<part>". Un "<part>" est associé à une donnée décrite dans "<types>".
- <portType> : désigne un ensemble d'opérations. Une balise "<operation>" est une description abstraite d'une action accomplie par le service. Elle contient un sous-élément "<input>" et un sous-élément "<output>". Ils décrivent les paramètres d'entrée et de sortie de l'opération. Un "<input>" ou un "<output>" possède un attribut message qui référence un élément "<message>".
- <binding> : reprend les opérations de l'élément "<portType>" et leurs associe un protocole de transfert de message. Concrètement, le "<binding>" décrit comment les messages seront échangés. Il spécifie un seul protocole d'échange mais il ne précise pas les adresses entre lesquelles les messages seront échangés. La spécification WSDL décrit le "<binding>" avec les méthodes "GET" et "POST" du protocole HTTP et présente un protocole d'échange plus élaboré pour HTTP, le protocole SOAP.
- <service> : précise l'adresse ou les adresses auxquelles se trouve le service. Un <service> est un ensemble de <port>. Un <port> spécifie une adresse pour un <binding> donné.

En résumé, les éléments <portType> décrivent les opérations du service en terme d'entrées et de sorties. Les <binding> décrivent concrètement les messages échangés par les opérations à travers un protocole choisi. Le <service> associe à chaque <binding> une adresse à laquelle se trouve l'implémentation.

Chronologiquement IBM, Microsoft et Ariba ont développé WSDL 1.0 en 2000. WSDL 1.1

est publié en mars 2001 comme une recommndation du W3C. WSDL 1.2 est une nouvelle version simplifiée, mais n'a pas été à ce jour validée par le W3C. En juin 2007, le WSDL 2.0 a été publié comme une recommandation du W3C. WSDL 2.0 n'est autre que la spécification WSDL 1.2 dont le nom a changé vu les différences majeures qui le démarque de WSDL 1.1.

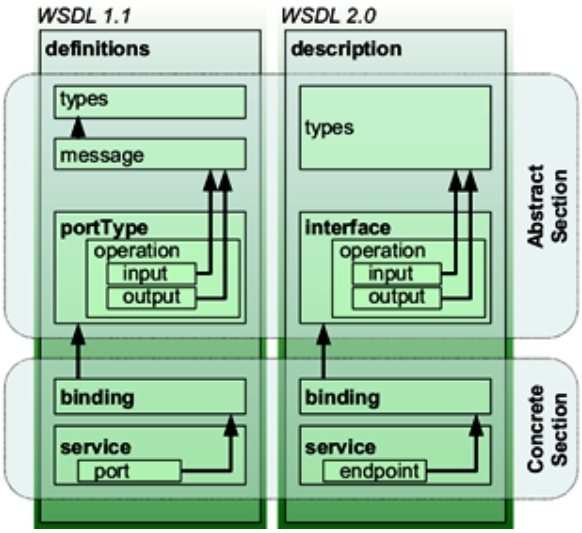


Fig 2.5 – Comparaison de WSDL 1.1 et WSDL 2.0.

Nous retenons les changements suivants (Cf. Fig 2.5) : (1) <portType> est devenu <interface>, (2) <port> est renommé en <endpoints>, (3) <message> n'est plus utilisé, un <xs :element> suffit pour définir les données échangées.

2.3.3 UDDI

Les deux standards présentés précédemment définissent ensemble l'aspect le plus basique de développement de l'infrastructure des Services Web. Toutefois, dans un environnement ouvert comme l'internet, le modèle de description des Services Web n'est d'aucune utilité s'il n'existe pas un moyen de localiser aussi bien les services que leurs descriptions WSDL.

Un troisième standard a été conçu pour réduire l'écart entre les applications clientes et les Services Web (Cf. Fig 2.6), appelé UDDI [Bellwood *et al.*, 2005].

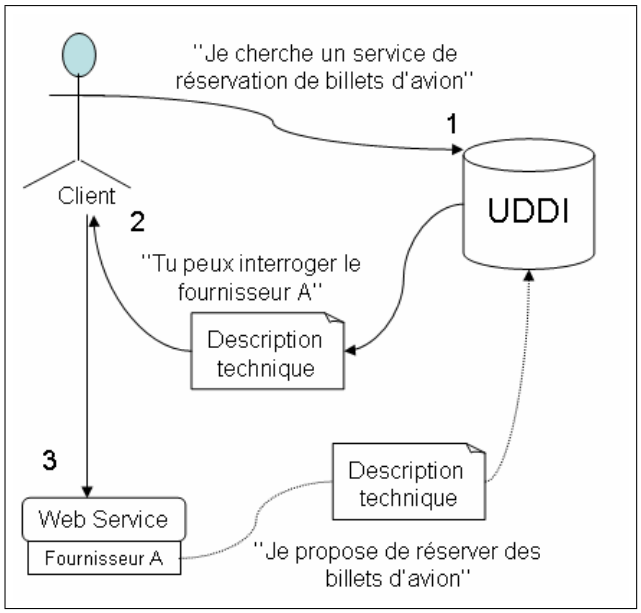


Fig 2.6 – Emploi de l'UDDI pour la recherche d'un Service Web.

Les registres UDDI offrent deux fonctionnalités de base : la publication des différents types d'information sur les services et leurs fournisseurs selon un schéma de description, et la consultation du contenu des registres. La mise en place d'un registre UDDI suit un processus uniforme imposé par la spécification. Chaque organisation qui veut mettre en place un registre UDDI doit suivre ce processus pour devenir un opérateur UDDI. Les registres UDDI créés sont organisés en réseaux. Ils partagent les différentes informations publiées. La publication d'un service chez un opérateur, donne lieu automatiquement à un processus de propagation des informations aux différents registres UDDI. L'accès à l'ensemble d'informations des registres peut se faire de n'importe quel opérateur UDDI.

Les informations sur un service publié dans un annuaire UDDI se présentent sous forme de trois (3) facettes :

- Les pages blanches comprennent la liste des entreprises ainsi que des informations associées à ces dernières (coordonnées, description de l'entreprise, identifiants...) ;
- Les pages jaunes recensent les Services Web de chacune des entreprises sous le standard WSDL ;
- Les pages vertes fournissent des informations techniques précises sur les services fournis.

Un annuaire UDDI est conçu pour être interrogé par des messages SOAP et afin de pouvoir stocker et fournir des informations permettant l'accès aux documents WSDL. Les outils de recherche disponibles sont basés sur des mots-clés et ne prennent pas en considération les relations entre les Services Web et les caractéristiques sémantiques de chaque Service Web,

forçant l'utilisateur à recommencer la recherche depuis le début en utilisant de nouveaux mots-clés.

SOAP, WSDL et UDDI sont les trois standards qui constituent l'infrastructure des Services Web. Ensemble, ils résolvent les problèmes de l'hétérogénéité des systèmes pour l'intégration d'applications en ligne.

2.4. Le Web Sémantique

Le Web Sémantique (WS), qui est une extension du Web actuel, a comme objectif majeur d'apporter de la sémantique à l'information manipulée afin de faciliter la communication entre agents (hommes et machines). Tim Berners-Lee, auteur du premier article sur ce thème, et al. prenant acte du fait que le Web actuel n'est "compréhensible" que par les humains, indique que l'objectif du WS est de fournir aux machines un meilleur accès (automatisé) à l'information [Berners-Lee *et al.*, 2001] grâce à des métadonnées qui décrivent les informations disponibles. Ces métadonnées sont fournies par le composant pivot du Web Sémantique, les ontologies.

Les ontologies représentent une technologie dont le but est d'améliorer l'organisation, la gestion et la compréhension de l'information électronique. Elles servent de vocabulaire standardisé pour le partage de connaissances. Les Ontologies ont été utilisées dans une large gamme d'applications telles que ; l'intelligence artificielle et les systèmes à base de connaissances [Janev and Vranes, 2009].

2.5. Les ontologies

2.5.1 Définitions

Dans l'univers du traitement automatique de l'information, le terme "ontologie" a été introduit dans les années 1990 avec les travaux de Grüber et son équipe à Stanford [Gruber, 1993]. Il est intéressant de noter que la référence la plus consultée sur le web, obtenue lors d'une recherche par le moteur de recherche Google, est l'article de Grüber "What is an Ontology" où il écrit : "An ontology is a explicit specification of a conceptualization". "Conceptualization" fait référence à un modèle abstrait de certains phénomènes du monde, ce modèle identifie les concepts pertinents de ce phénomène. "Explicit" signifie que le type des concepts utilisés et les contraintes sur leur utilisation sont explicitement définis.

Guarino et Giaretta [Guarino and Giaretta, 1995] affinent la définition de Grüber en considérant les ontologies comme des spécifications "partielles" et "formelles" d'une conceptualisation. Les ontologies sont "formelles" car elles sont exprimées sous un formalisme doté d'une sémantique

formelle compréhensible par les machines, et "partielles" car une conceptualisation ne peut pas toujours être entièrement formalisée dans un tel cadre, du fait d'ambiguïtés ou du fait qu'aucune représentation de leur sémantique n'existe dans le langage de représentation choisi.

En 1997, Borst [Borst, 1997] a modifié légèrement la définition proposée par Grüber en énonçant : "Une ontologie est définit comme étant une spécification formelle d'une conceptualisation partagée". L'adjectif "formel" précise que l'ontologie construite doit être lisible par un ordinateur et, enfin, le terme "partagée" montre qu'une ontologie fournit un vocabulaire conceptuel commun et une compréhension partagée par la communauté visée.

En 1998, Mizoguchi [Mizoguchi, 1998] a rajouté une définition qui provient du point de vue des systèmes à base de connaissance (SBC) : "Une ontologie est une théorie de concepts/vocabulaire, utilisée comme module des systèmes de traitement de l'information".

2.5.2 Types d'ontologies

En fonction de leur usage, on distingue classiquement cinq catégories d'ontologies [Heijst *et al.*, 1997] : génériques, de domaine, d'application, de représentation et de méthodes (Cf. Fig 2.7).

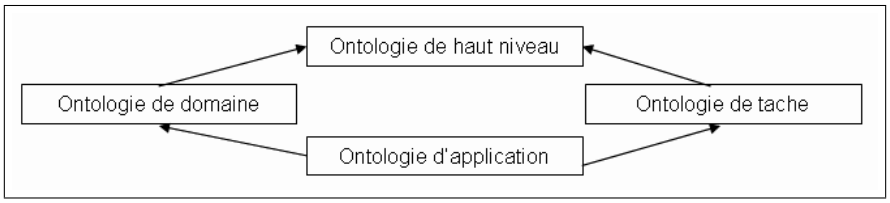


Fig 2.7 – Typologies d'ontologies selon Van Heijst [Heijst *et al.*, 1997]

- **Ontologies de domaine** : Ces ontologies expriment des conceptualisations spécifiques à un domaine. Elles sont réutilisables pour plusieurs applications de ce domaine. L'ontologie du domaine caractérise la connaissance du domaine où la tâche est réalisée. Par exemple, dans le contexte du e-learning, le domaine peut être celui de la formation.
- **Ontologies applicatives** : Ces ontologies contiennent des connaissances du domaine nécessaires à une application donnée, elles sont spécifiques et non réutilisables. Par exemple, dans le contexte du e-learning, une application peut être : la formation du module "statistiques et probabilités".
- **Ontologies génériques ou ontologies de haut niveau (top-ontologies)** : Ces ontologies expriment des conceptualisations valables dans différents domaines. Ce type d'ontologies est fondé sur la théorie de l'identité. Son sujet est l'étude des catégories des choses qui existent dans le monde. Comme les concepts de haute abstraction tels

que les entités, les évènements, les états, les processus, les actions, le temps, l'espace, les relations, les propriétés, etc. [Lenat and Guha, 1990] [Sowa, 1995].

2.6. Conclusion

Dans ce chapitre, nous avons présenté les principes des Services Web et du Web Sémantique qui constituent une sorte de révolution de fond du Web actuel. 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 préoccupation première l'interopérabilité entre applications via le Web en vue de rendre le Web plus dynamique. Ainsi, nous avons présenté une des bases du succés du Web Sémantique à savoir, les ontologies.

L'application des principes du Web Sémantique au domaine des Services Web permet d'utiliser le sens et la signification des données échangées pour améliorer la faisabilité des différentes tâches : découverte, composition, etc. En effet, cet aspect offre aux Services Web, d'une part,une description sémantique explicite de leurs fonctionnalités aussi compréhensible par les agents logiciels que par les utilisateurs humains, et d'autre part, une interprétation correcte des informations envoyées et reçues dans le cadre de découverte, d'invocation et surtout de composition et d'orchestration.

Dans le chapitre suivant, nous allons présenter la technologie des Services Web Sémantiques avec un état de l'art des différentes approches et outils qui ont été proposés dans ce domaine.

Chapitre

3 Les Services Web Sémantiques : Revue de la littérature

Sommaire

3.1. Introduction

La technologie des Services Web Sémantiques a emergé comme une solution pour résoudre les problèmes de découverte et de composition relatifs aux Services Web syntaxiques courants [Vitvar *et al.*, 2007].

L'idée des Services Web Sémantiques est basée sur l'utilisation des ontologies pour fournir des descriptions sémantiques aux éléments du service. Actuellement les Services Web sont décrit par le langage WSDL qui permet de définir les opérations et les paramètres autorisés par le Service Web. Cela est réalisé en attribuant des noms aux opérations et aux paramètres, puis associer ces paramètres à des types abstraits de données (exemple : string, char,...).

Prenons un exemple d'un Service Web qui fait la traduction d'un mot de l'anglais vers le

Français, "Translation_En_Fr". Ce service accepte un paramètre en entrée "word_en" de type "string", et un paramètre de sortie "word_fr" de type "string" aussi.

Cependant, le problème avec ce type de description est que lorsqu'on veut automatiser les divers aspects liés aux Services Web (découverte, composition,...), l'agent manipulant les paramètres du Service Web ne peut pas comprendre la signification de ces données. Pour l'agent logiciel c'est juste des variables dénotées contenant de l'information. La seule chose que l'agent logiciel puisse inférer de la description précédente est que les deux paramètres "word_en" et "word_fr" soient respectivement un paramètre d'entrée et un paramètre de sortie de type "String". A ce stade, les Services Web ne sont décrits qu'au niveau syntaxique. Un développeur voulant programmer une application cliente interagissant avec un Service Web doit, tout d'abord, avoir connaissance de la syntaxe de sa description, l'interpréter, puis écrire le code client conforme aux paramètres de la description du Service Web. Cependant, un agent logiciel ne peut lire la description d'un Service Web comme un humain, il peut prendre connaissance de la structure syntaxique de la description mais pas de sa sémantique.

Les Services Web Sémantiques sont des Services Web décrits de telle sorte qu'un agent logiciel puisse interpréter les fonctionnalités offertes par le Service Web. Un agent logiciel doit être capable de lire la description d'un Service Web pour déterminer si le Service Web fournit les fonctionnalités désirées, et s'il est lui-même capable d'utiliser ce service. Pour permettre cela, la description du Service Web doit être supplementée en information sémantique interprétable par la machine. Les paramètres du Service Web doivent être décrits de façon qu'un agent logiciel puisse avoir connaissance de leur signification. Cela est fait par la définition de vocabulaires organisés en ontologies.

Dans le domaine des Services Web, les ontologies sont utilisées pour annoter, avec une sémantique, les descriptions des fonctionnalités des Services Web. Ainsi pour qu'un agent logiciel puisse avoir connaissance de la sémantique d'une description d'un Service Web, il lui suffit juste d'accéder à une ontologie du domaine.

Un agent essayant d'interpréter la description du Service Web aura juste à utiliser l'ontologie où les annotations sémantiques sont définies. En utilisant un moteur d'inférence, l'agent peut inférer les similitudes entre la sémantique employée pour décrire le service et la sémantique connue par l'agent.

3.2. Infrastructure des Services Web Sémantiques

Selon [Cabral *et al.*, 2004], le développement des Services Web Sémantiques (SWS) est effectué en trois dimensions : les activités d'utilisation, l'architecture et l'ontologie de service (Cf. Fig 3.1).

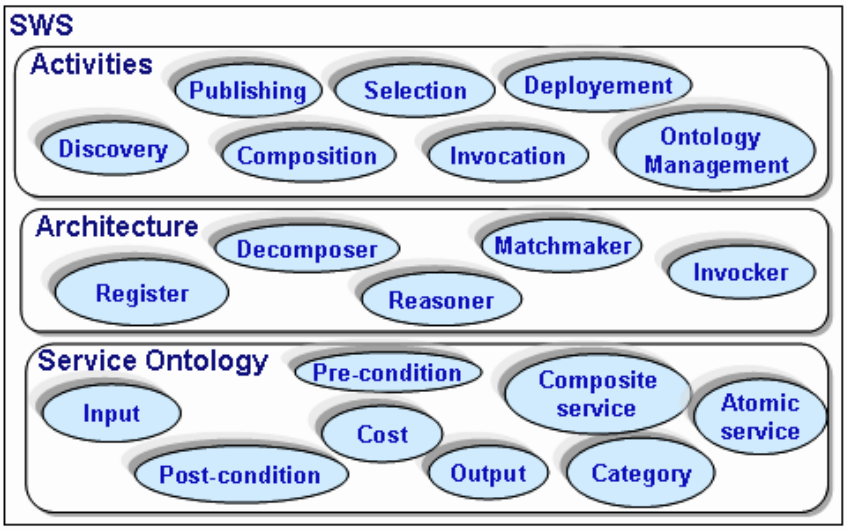


Fig 3.1 – Infrastructure des Services Web Sémantiques selon [Cabral *et al.*, 2004].

Ces trois dimensions couvrent la portée des présrequis à l'implémentation des SWS au niveau métier, matériel et conceptuel. La première dimension couvre les besoins fonctionnels qu'un cadre de travail implémentant les SWS se doit d'apporter une réponse. L'architecture des SWS est une agrégation de tous les modèles conceptuels liés à la description d'un SWS, et la troisième dimension constitue ainsi les métadonnées des informations décrivant et supportant l'utilisation du service, ainsi l'ontologie de service spécifie l'information utilisée pour le matching durant la découverte et l'invocation du service. En outre les connaissances du domaine devraient être publiées ou liées à l'ontologie de service.

Du point de vue de la première dimension, les SWS sont vus comme des objets actifs au sein d'une application métier. Les activités requises à l'exécution d'une telle application utilisant les SWS sont : la publication, la découverte, la sélection, la composition, l'invocation, le déploiement et la gestion des ontologies.

- La publication d'un SWS permettra à des agents ou à des applications de découvrir les services sur des buts et capabilités. Un registre sémantique est employé pour enregistrer les instances de l'ontologie de service pour différents services.
- La découverte des services consiste en un matching sémantique entre la description du service demandé et la description du service publié. Les requêtes impliquant le nom du

service, l'entrée, la sortie, les préconditions et autres attributs, peuvent être construites et employées pour la recherche du registre sémantique. Le matching peut également être fait au niveau des tâches ou des objectifs à réaliser, suivi par la sélection des services qui exécutent la tâche. Par exemple, on peut dire qu'une entrée pour le type "Professor" fourni par un service correspond à une entrée de type "Academic" d'un service demandeur.

- La sélection des services est requise s'il y a plus qu'un service correspondant à la requête. Les attributs non-fonctionnels tels que le coût ou la qualité sont utilisés pour choisir un service. En général, un "broker" se charge du contrôle de la satisfaction des préconditions des tâches et des services et montre que les postconditions et les effets impliquent l'accomplissement de l'objectif.
- La composition permet au SWS d'être défini en termes de services plus simples. Un "workflow" exprimant la composition des services atomiques peut être défini dans l'ontologie de service en employant les constructeurs appropriés. Cette description serait fondue sur une description syntaxique telle que BEPL4WS(Business Process Execution Language for Web Services) [1]. La composition dynamique est également vue comme une approche pendant la demande de service dans laquelle les services atomiques nécessaires pour satisfaire une demande sont localisés et composés sur le champ. Cela nécessite un "Invoker" qui fait correspondre les sorties des services atomiques aux entrées du service demandé.
- L'invocation d'un SWS se fait en plusieurs étapes, une fois que les données d'entrée ("inputs") ont été fournies par le consommateur du service. Premièrement, les ontologies de domaine correspondant à un service doivent être instanciées. Ensuite, les "inputs" doivent concorder avec les classes de l'ontologie du domaine du service. Enfin, le service peut être invoqué ou bien un "workflow", un enchaînement de processus métier, peut être exécuté à travers les informations fournies par le service. Il est aussi important de surveiller l'état du processus de décomposition (dimension architecture) et d'informer ; en cas d'une exception, le consommateur de service.
- Le déploiement d'un Service Web par un fournisseur se fait indépendamment de la publication de sa description sémantique, dés lors un Service Web peut offrir plusieurs fonctionnalités distinctes. Cependant, les SWS peuvent fournir des mécanismes de déploiement à la demande d'un code auto généré pour une description sémantique donnée.
- La gestion des ontologies de service est une activité cruciale pour les SWS puisqu'elle garantit la création, la consultation, la réutilisation des descriptions sémantiques du

1. http ://www.ibm.com/developerworks/library/ws-bpel.

service dans le Web Sémantique.

De la perspective d'architecture, les SWS sont définis par un ensemble de composants qui réalisent les activités citées précédemment, avec les mécanismes fondamentaux de sécurité et de confiance. Ces composants incluent : un "Register", un "Reasoner", un "Matchmaker", un "Decomposer" et un "Invoker".

- "Reasoner" est utilisé pendant toutes les activités et fournit le support de raisonnement pour interpréter les descriptions sémantiques et les requêtes ;
- "Register" fournit des mécanismes pour la publication et la localisation des services dans un registre sémantique, ainsi que des fonctionnalités pour créer et publier les descriptions du service ;
- "Matchmaker" joue le rôle de médiateur entre le demandeur et le "Register" durant la découverte et la sélection des services ;
- "Decomposer" est le composant utilisé pour exécuter le modèle de composition des services composés ;
- "Invoker" joue le rôle de médiateur entre le demandeur et le fournisseur, ou le "Decomposer" et le fournisseur lors de l'invocation des services.

L'ontologie de service constitue l'élément le plus important dans l'architecture des SWS. Dans la section suivante, nous allons décrire en détail cet élément.

3.3. Approches relatives aux Services Web Sémantiques

Plusieurs travaux de recherche considérables ont été élaborés dans le domaine des Services Web Sémantiques, tels que, OWL-S [Martin *et al.,* 2004], WSMO [Roman *et al.*, 2005], WSDL-S [Miller *et al.,* 2005] et SAWSDL [Farrell and Lausen,2007]. Chaque initiative a acquis un succès considérable et a traité certains aspects pragmatiques. Ces approches de réalisation des Services Web Sémantiques se divisent en deux catégories, la première catégorie consiste à développer un langage complet qui décrit le Service Web et sa description sémantique dans un seul bloc, on peut citer dans cette catégorie, les approches OWL-S et WSMO. La deuxième catégorie consiste à annoter les langages existants avec une description sémantique, parmi les approches qui s'inscrivent dans cette catégorie, nous citons, WSDL-S et SAWSDL.

Nous présentons dans ce qui suit, les principales approches ainsi que les différents outils proposés dans le domaine des SWS.

3.3.1 Approches basées sur des langages sémantiques

Une ontologie de services saisit les differents aspects liés à la description des services et leur utilisation à travers un ensemble de concepts, de propriétés et de relations entre eux. Deux modèles d'ontologies de services sont décrits ci-après : OWL-S et WSMO.

a) OWL-S

OWL-S (OWL-based Web Service Ontology) [Martin *et al.*, 2004] a pour objectif d'ajouter des descriptions sémantiques aux Services Web (en plus de leur description syntaxique WSDL). OWL-S permet une plus grande expressivité en permettant la description des caractéristiques des services afin de pouvoir raisonner dessus dans le but de découvrir, invoquer, composer et gérer les Services Web d'une façon plus automatisée que possible.

Modèle conceptuel

Le OWL-S définit un ensemble d'ontologies de haut niveau se basant sur le OWL et qui sont utilisées pour décrire les différents aspects d'un Service Web Sémantique. Cet ensemble est organisé en trois zones conceptuelles : le profil ("Profile"), le modèle de processus ("Process-Model") et les liaisons avec le service ("Grounding"). La figure 3.2 représente ces trois concepts qui permettent de répondre aux questions suivantes :

- Quelles sont les informations renvoyées par le service ? La réponse à cette question est donnée par le profil du service : la classe "ServiceProfile" ;
- Comment fonctionne le service ? La réponse est donnée par le modèle de processus du service : la classe "ServiceModel" ;
- De quelle façon le service doit-il être utilisé ? La réponse à cette dernière question est donnée par la classe "ServiceGrounding".

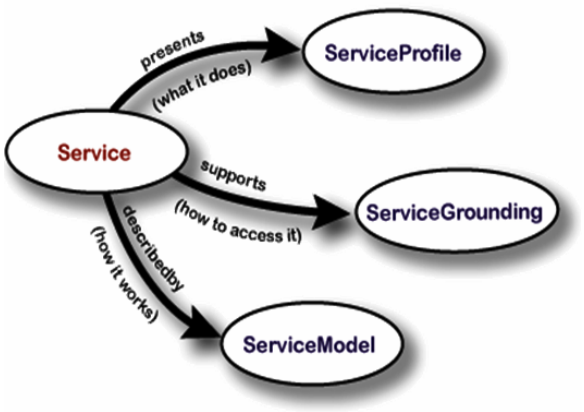


Fig 3.2 – Les éléments de base du OWL-S selon [Martin *et al.*, 2004].

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

La classe "ServiceModel" donne les conditions nécessaires à l'exécution d'un service ainsi que ses conséquences. Elle inclut des informations concernant les entrées, sorties, préconditions et effets d'un service. Dans le cas d'un service complexe (i.e.,composé de plusieurs étapes dans le temps), le modèle de processus détaille la décomposition du service en composants plus simples et en explicitant les liaisons et structures de contrôle permettant l'enchaînement de ces composants. La classe "ServiceGrounding" spécifie la manière, pour un agent, d'accéder au service concerné.

Pour chaque processus, OWL-S permet donc de modéliser un ensemble d'entrées et de sorties. Il permet également de spécifier les conditions sous lesquelles les données de sortie et les effets seront produits. Les entrées, sorties, préconditions et effets jouent un rôle crucial dans la caractérisation des processus.

Langage

Par conception, le langage primaire utilisé pour la description des services est le OWL. Par la réutilisation du OWL comme standard recommandé, OWL-S a obtenu un succès considérable. Cependant il est clairement apparu bientôt qu'il n'est pas suffisamment expressif pour tous les aspects d'un service. OWL-S a pour inconvénient, le besoin de langages plus expressifs, ce qui ouvre alors de nouvelles issues de recherches sur la façon dont il devrait interagir.

b) WSMO

WSMO (Web Service Modelling Ontology) [Roman *et al.*, 2005] est un modèle conceptuel qui vise à décrire tous les aspects liés aux services et qui sont accessibles par une interface de Service Web. WSMO fournit des spécifications ontologiques pour les éléments du noyau des Services Web Sémantiques, son but final est de permettre l'automatisation (totale ou partielle) des activités telles que la découverte, la sélection, la composition, la médiation, l'exécution et la surveillance impliquées dans l'intégration intra et inter-entreprise des Services Web.

WSMO a un modèle conceptuel basé sur le WSMF (Web Services Modeling Framework) [Fensel and Bussler, 2002]. WSMO représente à la fois une ontologie formelle et un langage.

Le WSMO est basé sur des ontologies, les ontologies sont utilisées comme modèles de données à travers le WSMO ; cela signifie que les descriptions des ressources et des données échangées durant l'usage du service sont basées sur des ontologies.

WSMO fournit trois principales catégories pour structurer les descriptions sémantiques. Premièrement, il fournit des moyens pour décrire les Services Web ; en second lieu, il fournit des moyens pour décrire les buts (”Goals”) des utilisateurs se rapportant à la résolution des problèmes. Troisièmement, il fournit des moyens pour assurer l'interopérabilité entre les diverses descriptions sémantiques des environnements hétérogènes : ontologies et médiateurs.

WSMO consiste en quatres éléments (Cf. Fig 3.3) : Ontologies, Services Web , Buts et Mediateurs.

Fig 3.3 – Les éléments du WSMO [Roman *et al.*, 2005].

- **Les Services Web :** Sont définis comme des entités qui fournissent une fonctionnalité. Une description est associée à chaque service, dans le but de décrire sa fonctionnalité, son interface, et ses détails internes ;
- **Les Buts(”Goals”) :** Servent à décrire les souhaits des utilisateurs en termes de fonctionnalités requises. Les buts représentent une vue orientée utilisateur du processus d'utilisation des Services Web, ils consistent en une entité à part entière dans le modèle WSMO. Un but décrit la fonctionnalité, les entrées, les sorties, les pré-conditions et les postconditions d'un Service Web ;
- **Les Médiateurs :** Sont utilisés pour résoudre de nombreuses incompatibilités, telles que les incompatibilités de données dans le cas où les Services Web utilisent différentes terminologies, les incompatibilités de processus dans le cas de la combinaison de Services Web, et les incompatibilités de protocoles lors de l'établissement des communications.
- **Les Ontologies :** Fournissent la terminologie de référence aux autres éléments WSMO, afin de spécifier le vocabulaire du domaine de connaissance d'une manière interopérable par les machines.

Dans WSMO, nous distinguons également deux composants principaux qui sont le WSML (Web Service Modeling Language) et WSMX (Web Service Execution Environment) :

- **WSML** [2] : Fournit un langage formel pour la description des éléments définis dans

2. http ://www.w3.org/Submission/WSML.

l'architecture WSMO. Il est basé sur différents langages logiques qui sont la logique de description, la logique du premier ordre et la logique de programmation ;

– **WSMX**[3] : Est un environnement d'exécution qui offre un certain nombre de modules utilisables en temps d'exécution permettant de prendre en charge la découverte, la sélection, la médiation et l'invocation des services sémantiques.

3.3.2 Approches de description à base d'annotation

L'annotation sémantique consiste à enrichir et à compléter la description d'un service. Elle établit des correspondances entre des éléments de la description et des concepts d'un ensemble d'ontologies de référence. Une ontologie de référence permet de représenter un domaine par des structures interprétables par une machine. Deux modèles principaux suivent l'approche d'annotation sémantique, à savoir WSDL-S et SAWSDL. Ces deux modèles permettent d'annoter manuellement une description WSDL avec des éléments faisant référence à des ontologies.

a) WSDL-S

WSDL-S (Web Service Description Language-Semantic) [Miller *et al.*, 2005] est un langage de description sémantique des Services Web. Une description WSDL-S de Service Web est une description WSDL augmentée de sémantique (Cf. Fig 3.4), cette sémantique est ajoutée en deux étapes : La première étape consiste à faire référence, dans la partie définition de WSDL, à une ontologie dédiée au service à publier. La seconde étape consiste à annoter les opérations de la définition WSDL de sémantique en ajoutant deux nouvelles balises ; la balise "Action" et la balise "Contrainte".

– La balise "Action" permet de représenter l'action de l'opération ;

– La balise "Contrainte" représente les pré et post-conditions d'une opération.

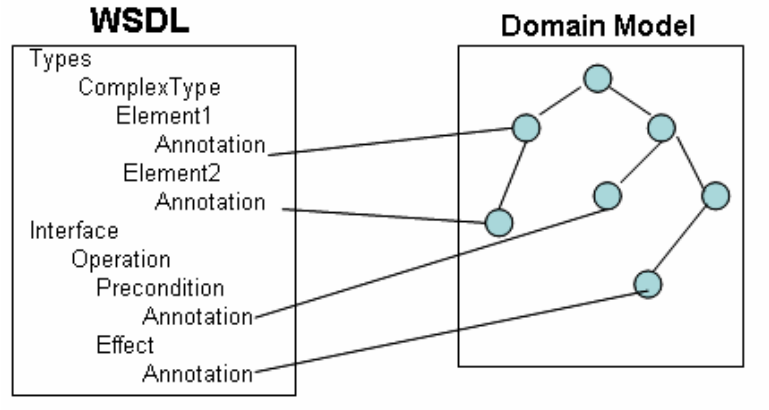


Fig 3.4 – Processus d'annotation du WSDL-S [Miller *et al.*, 2005].

WSDL-S distingue quatre modèles sémantiques à savoir :

3. Implémentation de référence de WSMO disponible dans http ://www.wsmx.org.

- **InputSemantic :** Le sens des paramètres d'entrée ;
- **OutputSemantic :** 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 réalise son exécution.

b) SAWSDL

SAWSDL (Semantic Annotations for Web Services Description Language and XML Schema) [Farrell and Lausen,2007] recommandé par le W3C en avril 2007, est une des approches les plus récentes initiée par la communauté des Services Web Sémantiques. Elle présente un mécanisme permettant d'annoter sémantiquement les services décrits avec des documents WSDL et leurs schémas XML associés. Les auteurs affirment que SAWSDL ne spécifie pas un langage pour représenter les modèles sémantiques, mais plutôt, il fournit un mécanisme à travers lequel des concepts appartenant à des modèles sémantiques existants peuvent être référés à partir d'un document "WSDL 2.0".

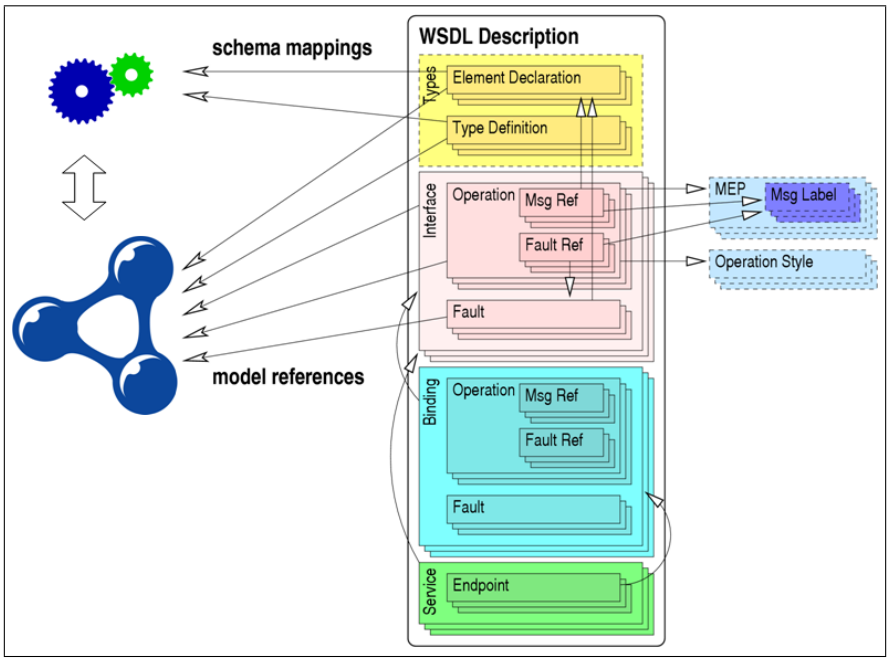


Fig 3.5 – L'approhe SAWSDL [Farrell and Lausen,2007].

SAWSDL (Cf. Fig 3.5) propose des extensions à WSDL 2.0 similaires à celles proposées par WSDL-S. Sa particularité réside dans l'annotation supplémentaire des schémas XML. SAWSDL fournit des mécanismes pour référencer des concepts de modèles définis à l'extérieur du document WSDL. Cela se fait grâce à l'attribut "sawsdl". Il existe trois extensions de cet attribut.

La première est "modelReference" et permet d'associer un composant WSDL ou XML Schema à un concept d'une ontologie. Les deux autres sont "liftingSchemaMapping" et "loweringSchemaMapping" et permettent de spécifier la correspondance (dit Mapping) entre les données sémantiques et les éléments XML. Les "SchemaMapping" sont utilisés pour établir la correspondance entre les structures des entrées et des sorties, et sont utiles lorsque les structures XML demandées par le client et celles fournies par le service sont différentes.

3.4. Outils des Services Web Sémantiques

3.4.1 METEOR-S

Le projet METEOR-S [Sheth *et al.*, 2005] initié par le laboratoire LSDIS vise l'intégration des technologies de services avec celles du Web Sémantique. Il ne fournit pas un nouveau modèle de représentation des services mais propose de générer des annotations sémantiques pour les fichiers de type WSDL. METEOR-S est basé sur le projet METEOR qui signifie "Managing End To End OpeRations". Ce dernier traite de la gestion des "workflows" de processus à grande échelle dans des environnements hétérogènes. L'idée principale de METEOR-S est de générer une annotation sémantique de l'interface d'un service à travers des références à une ontologie ajoutées manuellement dans le code source de l'implémentation du service. Les auteurs présentent un prototype permettant de générer des interfaces de services en (1) WSDL 1.1 où les liens avec les concepts d'une ontologie de référence sont décrits avec les éléments XML d'extensibilité autorisée par la spécification du langage ou en (2) WSDL-S à travers les attributs "modelReference", "schemaMapping" et l'élément "<category>".

3.4.2 OWL-S Editor

L'éditeur OWL-S [4] est un plug-in de protégé [5] pour élaborer les services OWL-S. L'éditeur fournit des moyens pour décrire les processus atomiques et composés (i.e. définition du process model). L'éditeur OWL-S fournit en outre certains supports limités de grounding, en produisant une structure des services OWL-S à partir des fichiers WSDL. Cependant, les options pour intégrer de nouvelles fonctionnalités ou étendre les fonctionnalités existantes sont limitées.

3.4.3 WSMT

Le WSMT [6](Web Service Modeling Toolkit) est un environnement intégré de développement (IDE) pour les Services Web Sémantiques, se basant sur la technologie des descriptions

4. http ://owlseditor.semwebcentral.org/
5. http ://protege.stanford.edu/
6. http ://sourceforge.net/projects/wsmt

de WSMO. WSMT représente une collection d'outils pour les Services Web Sémantiques destinés à l'utilisation avec le WSMO, le WSML et le WSMX. WSMO et les spécifications de WSML ainsi que l'environnement d'exécution WSMX ont été développés ensembles par DERI International [7].

WSMX est l'implémentation de référence du WSMO. C'est un environnement d'exécution où les Services Web sont intégrés pour différentes applications. Le but est de renforcer l'automatisation des processus du e-business d'une façon très flexible tout en fournissant des solutions d'intégration scalable. il est à noter que le language interne de WSMX est le WSML.

3.4.4 WSMO Studio

Le but du WSMO studio, est de créer une collection d'outils pour assister les utilisateurs potentiels à la création d'ontologie, la description, la découverte et la composition de services. Ces outils sont implémentés comme des "plug-ins" dans la plateforme Eclipse. Ces outils incluent un "WSMO Navigator" pour l'affichage des entités dans la description de WSMO avec différentes formes d'éditeurs pour chacune des entités de WSMO. Un éditeur de texte avec une syntaxe accentuante est également disponible pour éditer le format fondamental de WSML pour un utilisateur plus expérimenté.

Le WSMO studio fournit également des interfaces pour interagir avec des consignes WSMO afin de stocker et rechercher les descriptions WSMO.

3.5. Etude comparative des approches relatives aux SWS

Dans cette section, avons choisi d'abord de résumer, dans le tableau 3.1 quelques critères d'évaluation et ensuite comparer les cadres généraux relatifs aux Services Web Sémantiques sus-mentionnés selon ces critères.

Les caractéristiques de ces initiatives sont synthétisées à l'aide de six critères que nous avons retenu car ils représentent, à notre sens, les critères les plus pertinents. Il s'agit, en l'occurrence, des critères suivants :

- Critère "Portée" qui permet de préciser le niveau d'abstraction (conceptuel,technique) associé à l'approche considérée ;
- Le critère "Principaux éléments" qui permet de représenter les éléments constituant l'approche en question ;

7. http ://www.deri.org/

- Le critère "Supporter directement les propriétés non-fonctionnelles" indique si l'approche supporte les propriétés non-fonctionnelles. L'utlisation des propriétés non-fonctionnelles en plus des propriétés fonctionnelles (entrées, sorties, les pré, post-conditions), permet une découverte plus précise des services ;
- Le critère "Langage de description" qui permet de préciser le langage avec lequel les services sont décrits sémantiquement par l'approche étudiée ;
- Le critère "Médiation de services" qui permet de préciser la prise en compte des mécanismes de médiation par l'approche étudiée ;
- Le critère "Maturité" qui désigne en fait, le degré de maturité de l'approche vis à vis de la problématique d'intégration.

Tab 3.1 – Comparaison des principales approches relatives aux SWS.

Approche / **Critère**	**OWL-S**	**WSMO**	**WSDL-S**	**SAWSDL**
Portée	Modèle pour la description des SWS	Modèle + Langage pour la description des SWS	Annotation Sémantique du WSDL	Annotation Sémantique du WSDL et XML Schema
Principaux éléments	Service Profile , Service Process, Service Grounding	Ontologies, Goals, Services Web, Médiateurs	Opérations Description WSDL	Opérations Description WSDL
Supporter directement les propriétés non-fonctionnelles	Service Profile	Tous les éléments	Aucun élément	Aucun élément
Langage de description	OWL	WSML	Choix de l'utilisateur	Choix de l'utilisateur
Médiation de Services	Non Définie	Définie	Non Définie	Non Définie
Maturité	Forte	Forte	Moyenne	Moyenne

L'analyse du tableau 3.1, nous a permis de retenir un certain nombre de points importants qui sont :

- L'approche OWL-S est basée sur le OWL. OWL n'a pas été conçu pour la définition des sémantiques des processus qui exigent des définitions riches de leurs fonctionnalités, ainsi OWL-S a considérablement une expressivité assez limitée.

- OWL-S constitue une ontologie générique de services permettant principalement de décrire les Services Web dans un cadre général. Une des caractéristiques majeures de l'approche OWL-S est le fait qu'elle soit compatible avec les standards industriels notamment le WSDL, et , ce grâce, au "Service Grounding" qui permet de définir des mappings entre OWL-S et WSDL. Certains travaux existent quant à la découverte et la composition de services en utilisant OWL-S.
- WSMO constitue une approche basée sur des fondements conceptuels issus du WSMF pour la réalisation des Services Web Sémantiques. Elle est beaucoup plus orientée utilisateur dans le sens où elle permet de spécifier les désirs des utilisateurs en fonction de l'élément "Goal" et d'offrir un langage très proche des utilisateurs. WSMO/WSML était conçu pour remédier à certaines limitations en fournissant plusieurs couches d'expressivité permettant ainsi des définitions plus riches des Services Web.
- WSML fournit un langage intégré pour la description à la fois des Ontologies et des Services. En outre, le langage logique utilisé pour la spécification des pré-conditions et les postconditions des Services Web est une partie intégrale du langage. Ainsi la description globale des Services Web et les expressions logiques qui spécifient les pré-conditions et post-conditions sont connectées automatiquement.
- Le concept de médiateur du WSMO est très important pour résoudre les problèmes d'interopérabilité. La définition des goals est nécessaire pour l'automatisation de la chorégraphie et de l'orchestration. WSMO sépare le fournisseur et le point de vue du demandeur de service en définissant les buts ("Goals") et les capabilités des services ce qui le différencie du OWL-S qui ne modélise pas les "Goals".
- Les propriétés non-fonctionnelles dans OWL-S sont limitées au "Service Profile". D'autre part, WSDL-S et SAWSDL ne supportent pas ces propriétés directement. Toutefois, WSMO offre entre autre la possibilité d'ajouter des propriétés non-fonctionnelles, elles peuvent être utilisées pour décrire les aspects tels que le temps moyen de réponse ou le protocole de chiffrage utilisé pour la transaction. WSMO fournit une liste extensible de telles propriétés qui sont basées sur des normes communes de métadonnées telles que "Dublin Core"[8]. WSMO n'est pas limité uniquement à la composition des Services Web mais à modéliser l'orchestration aussi.
- WSDL-S et SAWSDL constituent une approche basée sur l'extension des standards. Le principe d'annotation sémantique est basé principalement sur l'utilisation des fichiers de description des services, WSDL et une ontologie de domaine. Bien que ces approches

8. http ://www.dublincore.org/

reposent sur des standards, il n'en demeure pas moins qu'elles manquent d'abstraction dans le sens où elles constituent des approches plus proches du niveau d'implémentation que du niveau conceptuel, en plus ces approches souffrent du manque d'ontologie de domaine appropriée.

– En gardant le modèle sémantique en dehors du fichier WSDL, WSDL-S et SAWSDL sont impartiales à n'importe quel langage de représentation de l'ontologie. L'avantage de ces approches est qu'elles s'appuient sur les standards actuels de l'industrie. Mais d'autre part, sans un certain degré de dépendance à un langage spécifique, ou une définition de la façon dont les différentes sémantiques des langages utilisables se rapportent les unes des autres, il est impossible de définir formellement le matching entre les offres et les demandes.

3.6. Conclusion

Dans ce chapitre, nous avons présenté un aperçu sur les cadres généraux relatifs aux Services Web Sémantiques. Toutes ces approches, à savoir OWL-S, WSMO, WSDL-S, et SAWSDL ont été soumissionnées au W3C. Cependant, il est à noter que l'approche SAWSDL est actuellement le seul standard officiel, pour les Services Web Sémantiques, recommandé par le W3C. Après l'introduction d'un ensemble de critères, les approches examinées ont été évaluées et comparées en fonction de ces critères.

Généralement, il y a deux catégories d'approches pour définir un Services Web Sémantique. Une approche révolutionnaire qui consiste à créer une ontologie de service où tous les aspects des Services Sémantiques sont reconsidérés, OWL-S et WSMO suivent cette approche. L'autre catégorie est plus évolutive, elle consiste à annoter des Services Web existants, cette approche reste compatible avec les standards et pratiques industrielles existantes. Des exemples concrets, le WSDL-S et SAWSDL.

En outre, les résultats de cette étude ont montré que OWL-S et WSMO fournissent des formalismes sémantiques beaucoup plus riches pour les Services Web, c'est la raison pour laquelle, nos contributions se focalisent sur l'ontologie WSMO. En revanche, WSDL-S et SAWSDL fournissent une approche assez légère.

Tous ces efforts de recherche sur les Services Web Sémantiques ont essayé de surmonter l'inconvénient du manque de sémantique des standards des Services Web actuels et de réaliser d'une manière automatique les tâches, telles que, la découverte, la composition et l'invocation des Services Web en fournissant des moyens de descriptions appropriées permettant l'efficacité d'exploitation des annotations sémantiques.

Cependant, la conception d'un Service Web Sémantique est un processus long et coûteux. Pour profiter pleinement des avantages de la technologie des SWS, un processus de ré-ingénierie des Services Web vers les Services Web Sémantiques est nécessaire. La ré-ingénierie permet de réutiliser les fonctionnalités des Services Web sous forme de Services Web Sémantiques sans reprendre l'écriture du code de ces applications.

Aussi, le nombre croissant de Services Web disponibles sur le Web soulève un nouveau et difficile problème de la recherche : "comment trouver les services appropriés, de manière efficace, dans un grand nombre de Services Web, qui répondent aux attentes des clients" devient de plus en plus difficile.

Dans le chapitre suivant, nous passons en revue les principales approches liées à la ré-ingénierie et à la découverte des Services Web et nous présentons dans le chapitre qui suit, nos contributions basées sur l'ontologie de service WSMO.

Une étude de cas portant sur les cadres proposés sera présentée en détail par la suite, qui va montrer l'impact des propositions en réduisant le temps et les efforts du processus de la ré-ingénierie et de la découverte dans son ensemble.

Chapitre

4 Etat de l'art

Sommaire

4.1. Introduction

Beaucoup d'entreprises cherchent à moderniser leurs applications Web en les introduisant dans un contexte SOA au moyen des services générés. La recherche dans le domaine du modèle orienté service a accordé une attention significative à divers aspects principaux relatifs aux Services Web, telles que la description sémantique des Services Web et la découverte des Services Web pertinents que nous présentons dans ce chapitre.

Cependant, le développement des Services Web Sémantiques à partir de zéro est souvent une tâche coûteuse, notamment lorsque le service est principalement publié et traité automatiquement. Afin de répondre aux exigences minimales de cohérence et de sécurité, la ré-ingénierie des Services Web Sémantiques à partir des Services Web existants peut être une solution raisonnable et efficace.

En outre, avec l'augmentation exponentielle des Services Web sur le net, les tâches typiques

comme la découverte de Services Web pertinents est devenu un défi majeur. C'est la raison pour laquelle, la technologie des Services Web Sémantiques a été proposée comme un effort de recherche vers l'automatisation de l'utilisation des Services Web en les améliorant avec des descriptions sémantiques appréhendables par la machine.

Dans ce chapitre, d'abord dans la section 4.2, nous étudions le domaine de la ré-ingénierie des Services Web vers les Services Web Sémantiques . Nous mettons l'accent particulièrement sur les principaux travaux relatifs à ce domaine.

Ensuite, dans la section 4.3, nous présentons un aperçu sur le domaine de la découverte des Services Web et nous passons en revue les principales approches relatives à ce domaine.

4.2. Ré-Ingénierie des Services Web

La besoin de changer les logiciels existants nous a accompagné depuis que les premiers programmes ont été écrits. Après les travaux pionniers de Lehman [Lehman and Belady, 1985], nous sommes convaincus que les systèmes logiciels du monde réel ont besoin de changement et d'amélioration continus pour répondre aux nouvelles exigences et attentes des utilisateurs, s'adapter aux nouveaux modèles commerciales et émergents et s'adhérer à l'évolution de la législation, pour faire face à l'innovation technologique et préserver le système de la détérioration.

Une partie importante des coûts consacrés au cycle de vie des logiciels est dédiée à la maintenance et l'évolution, bien que les chiffres varient, la communauté du génie logiciel est unanime sur le fait que la maintenance dépasse 50% du coût global des logiciels.

La nécessité d'une modification du logiciel induit la nécessité de le comprendre. La compréhension du logiciel est un processus humain intensif, où les développeurs doivent acquérir des connaissances suffisantes et nécessaires sur un logiciel, ou un système entier, de manière à être capable de mener à bien une tâche donnée, au moyen d'analyse, de découverte et de formulation d'hypothèses. Dans la plupart des cas, la compréhension du logiciel est contestée par le manque de documentation adéquate et mise à jour du logiciel en question, souvent dûe à des compétences limitées ou des ressources et contraintes de temps pendant les activités de développement. La rétro-ingénierie est un terme général qui englobe un ensemble de méthodes et d'outils pour obtenir des informations et des connaissances à partir des artefacts logiciels existants et en tirer parti dans les processus d'ingénierie logicielle.

Il y a quelques années, les recherches s'orientent de plus en plus vers la technologie des Services Web sémantiques. Pour être en phase avec cette nouvelle technologie sans pour autant laisser

tomber le patrimoine applicatif hérité du passé, un processus de ré-ingénierie des Services Web vers les Services Web Sémantiques est nécessaire.

Ce processus n'est pas aussi simple qu'il paraît. Cela est dû à la complexité du code des systèmes hérités et la diversité des sources de données, etc. Ainsi, le problème de la ré-ingénierie vers les SWS a fait l'objet de plusieurs travaux de recherche.

Dans ce qui suit, nous allons tout d'abord expliquer la notion de ré-ingénierie ainsi que les problèmes liés à ce processus, ensuite nous présentons un état de l'art sur la ré-ingénierie des Services Web vers les Services Web Sémantiques.

4.2.1 Définitions

Selon Chikofsky et al. [Chikofsky *et al.,* 1990], la ré-ingénierie est le processus d'examen et de modification d'un système dans le but de le reconstituer dans une nouvelle forme puis l'implémenter. Le processus de ré-ingénierie s'emploie, à partir d'un existant, plus ou moins bien documenté. Il consiste à remonter dans les niveaux d'abstraction afin d'approfondir les connaissances.

La ré-ingénierie permet de bénéficier des avantages des nouvelles technologies sans pour autant abandonner un énorme patrimoine applicatif informatique hérité après des années d'investissement dans le domaine industriel. Elle permet donc de réutiliser les fonctionnalités des systèmes hérités, à travers de nouvelles technologies, sans réécrire à zéro le code de ces systèmes.

La ré-ingénierie utilise des techniques de rétro-ingénierie et d'ingénierie directe. Nous allons, par la suite, expliciter ces deux concepts à travers des défintions.

- Rétro-ingénierie (Reverse engineering) : La rétro-ingénierie (ou ingénierie inverse) est le processus d'analyse d'un système permettant l'identification des entités et leurs corrélations en vue de passer d'une forme de représentation à une autre de niveau d'abstraction identique ou plus élevé. Cette activité consiste donc à examiner un existant, suivi éventuellement d'une remontée dans les niveaux d'abstraction, permettant ainsi une vue plus large du domaine et une meilleure compréhension du système d'information grâce à une redocumentation.

  Le processus de rétro-ingénierie des applications Web s'avère souvent difficile à cause de la non-disponibilité de la documentation associée à ces applications. En plus, la plupart des concepteurs n'ont pas suivi une méthodologie formelle pour le développement de ces applications. D'où la difficulté de comprendre le code et les relations entre les composants d'une application Web.

– Ingénierie directe (Forward engineering) : L'ingénierie directe (Forward engineering) correspond au processus de développement traditionnel d'un système. On part d'un niveau de représentation conceptuelle élevé et on cherche à obtenir un système opérationnel par une série de développements/modifications progressifs descendants. Cette activité a pour objectif soit la mise au point d'un nouveau système (ingénierie de développement) soit l'évolution d'un système existant (ingénierie de maintenance).

4.2.2 Problèmes de la ré-ingénierie

Le processus de ré-ingénierie consiste à analyser un système existant puis remonter dans le niveau d'abstraction (conceptuel, logique, etc.) et produire un nouveau système. En particulier, la ré-ingénierie des Services Web vers les Services Web Sémantiques consiste à analyser un Service Web pour le reconstituer sous une nouvelle forme, à savoir les Services Web Sémantiques.

Pour appliquer ce processus, plusieurs problèmes se posent, parmi lesquels on cite :

– Il est difficile de rencontrer les concepteurs des applications Services Web pour rendre facile la tâche de développement des Services Web sémantiques avec les mêmes fonctionnalités ;

– Le code des applications Web est souvent protégé par un serveur d'application et n'est pas accessible aux utilisateurs ;

– Le développement des Services Web Sémantiques nécessite à la fois une connaissance dans le domaine des Services Web (langages, plateformes, architecture, etc.), et une connaissance dans le domaine du Web Sémantique (Ontologies de domaine, Ontologies de service, etc.) ;

– La construction des ontologies de domaine pour les services n'est pas aussi une tâche facile, puisque le domaine de recherche du Web Sémantique et des SWS est en cours de développement, les outils dédiés aux ontologies sont en cours de développement.

Cependant, le processus de ré-ingénierie demeurre une solution plus au moins difficile et ceci est dû aux points suivants :

– La difficulté du processus de ré-ingénierie : Ce même processus est composé de plusieurs phases, à savoir la rétro-ingénierie et l'ingénierie directe. Chaque phase implique un ensemble d'étapes, telles que l'analyse, la conception et l'implémentation ;

– La complexité des applications Web : A la différence des applications traditionnelles, le

code d'une même application Web est écrit en différents langages, tels que HTML, JSP, PHP, ASP, etc, il faut noter aussi que ce même code peut parfois être crypté ;

– Le domaine des SWS est encore récent. Il n'existe pas de méthodologies parfaites pour l'ingénierie des SWS.

4.2.3 Approches de ré-ingénierie des Services Web vers les Services Web Sémantiques

Le développement des Services Web Sémantiques n'est pas un processus simple car il nécessite à la fois le développement des Services Web ainsi que la couche sémantique associée. Cette dernière dépend de l'ontologie de service choisie. Quelques travaux ont été consacrés au processus de développement des SWS. Ces travaux se distinguent par les modèles utilisés pour la conception et de l'ontologie de service concernée par le processus d'ingénierie.

Dans ce qui suit, nous allons présenter les principaux travaux relatifs à l'ingénierie des Services Web Sémantiques à partir des Services Web.

a) Approches à base de modèles

Ces approches sont fondées sur la description sémantique des Services Web dirigée par les modèles (MDA) ou à partir d'un ensemble de diagrammes UML fournis. En effet, ces approches maintiennent l'élaboration des descriptions sémantiques séparées de l'évolution réelle des services sous-jacents et fournissent une solution méthodologique pour créer des descriptions sémantiques pendant le développement du service. Ces approches basées sur la méthodologie logicielle sont en mesure de combiner les meilleures pratiques dans la spécification sémantique et le développement de Services Web.

Paolucci et al. [Paolucci *et al.*, 2003] ont développé un outil appelé WSDL2OWL-S. Cet outil fournit une transition entre WSDL et OWL-S. Les résultats de cette transformation sont une spécification complète de l'élément "Service Grounding" et des spécifications partielles (incomplètes) des éléments "Service profile" et "Service model". L'incomplétude des spécifications est dûe principalement au fait que OWL-S et WSDL diffèrent dans l'information contenue. La sortie de l'outil WSDL2OWL-S fournit la structure de base de la description OWL-S des Services Web et permet d'économiser beaucoup d'effort humain.

Gronmo et al. [Gronmo *et al.*, 2005] ont développé une approche supportée par un outil pour le passage d'un diagramme d'activités UML vers une description OWL-S. Cette approche utilise l'architecture dirigée par les modèles (MDA)pour générer une description OWL-S d'un Service Web à partir d'un modèle UML. L'outil permet la conversion de la représentation XML

du modèle UML dans des descriptions OWL-S.

Timm et Gannod [Timm and Gannod, 2005] ont introduit une approche ascendante pour le développement des Services Web Sémantiques basée sur le OWL-S à l'aide de diagrammes de classes UML et des diagrammes d'activités avec leurs profils UML. Cette approche utilise le WSDL pour exécuter les tâches de spécification, du "grounding" et d'exécution. L'approche consiste en quatre principales étapes : la modélisation, la conversion, le "grounding" et l'exécution. Le diagramme de classes UML est utilisé pour modéliser la structure du service et le diagramme d'activités UML facilite la modélisation de la composition.

MIDAS [Acuna and Marcos, 2006] est un framework à base MDA pour la modélisation et le développement des SWS. Le framework se concentre sur la création des SWS et les descriptions WSML associées en utilisant un modèle UML selon l'approche MDA. MIDAS ne fournit pas encore une méthode compréhensive couvrant toutes les phases de conception des SWS.

Brambilla et al. [Brambilla *et al.,* 2007] ont proposé une méthodologie à base de modèles pour la conception et le développement des SWS WSMO. Leur méthode est basée sur la spécification des processus d'affaires (BPMN) et les modèles d'ingénierie des applications Web (WebML) pour la génération semi-automatique des descriptions sémantiques WSMO.

Une approche de rétro-ingénierie pour la construction des SWS a été proposée par [Amar Bensaber and Malki, 2008]. Cette approche adopte une solution dirigée par les modèles (MDA). Les descriptions syntaxiques existantes du WSDL sont converties en diagrammes de classes UML et d'activités. Les diagrammes générés sont ensuite annoté à la main et transformés en descriptions OWL-S. Comme toutes les approches discutées, cette solution est étroitement couplé aux descriptions OWL-S. Une critique de cette approche est qu'elle s'appuie entièrement sur des descriptions syntaxiques pour générer des descriptions sémantiques par rétro-ingénierie, ce qui pourrait limiter l'expressivité des SWS générés.

Un autre projet de recherche [Kuropka *et al.,* 2008] qui est un environnement pratique de développement intégré (IDE) connu sous le nom CMU's [1] OWL-S pour le developpement, le déploiement et la consommation des SWS. L'IDE OWL-S adopte et étend les outils existants des Services Web, comme l'éditeur OWL-S, afin d'assister les développeurs dans le processus de développement, de déploiement et de consommation des Services sémantiques [Srinivasan *et al.,* 2006]. Ce projet est intégré dans l'environnement Eclipse, purement basé sur le langage de programmation Java et l'outil OWL-S et suit les deux paradigmes : dirigé

1. Carnegie Mellon University.

par le code et dirigé par les modèles pour le développement des SWS. Cette approche prend également en charge diverses étapes de développement des SWS, telles que la découverte, l'invocation, et l'exécution. Toutefois, elle ne répond pas à d'autres spécifications, telles que le WSMO.

b) Approches basées sur le matching ontologique

Le but de cette catégorie d'approches est d'appliquer les algorithmes de matching ontologique pour le développement des Services Web Sémantiques. Certaines approches s'appuient sur l'annotation de Services Web afin de simplifier la gestion des Services Web par l'automatisation partielle des tâches telles que la découverte, la composition ou l'invocation. Le défi majeur pour l'automatisation de la gestion des Services Web est que la plupart des informations concernant les Services Web sont en format compréhensible uniquement par les humains comme la documentation. Ainsi, une description interprétable par les machines est nécessaire pour résoudre l'intéropérabilité sémantique.

Les descriptions sémantiques représentent le moyen de remédier à cette lacune sous forme de métadonnées sémantiques décrivant la sémantique d'une information dans un format interprétable par la machine. Comme mentionné précédemment, une description des Services Web spécifie les propriétés d'un Service Web par des métadonnées. Une description sémantique à travers des métadonnées est basée sur une ontologie, ainsi, une propriété du Service Web est reliée à un concept correspondant défini dans une ontologie.

Dans ce qui suit, nous allons présenter les principales contributions utilisant le matching ontologique et relatives à la ré-ingénierie des Services Web Sémantiques.

MWSAF(METEOR-S Web Services Annotation Framework) est un framework d'annotation de Services Web introduit dans [Patil *et al.,* 2004] et modifié dans [Oldham *et al.,* 2004]. Il annote les descriptions WSDL des services avec des métadonnées émanant des ontologies. En raison de la différence dans l'expressivité de OWL et WSDL, il est difficile de faire correspondre directement les deux formats. MWSAF propose de réduire l'écart en transformant chacun d'eux à une représentation commune appelée "schemaGraph". Un "schemaGraph" est simplement une représentation graphique sous forme d'arbre d'un document XML ou OWL-S. Une fois les "schemaGraphs" ont été créés, les algorithmes de matching sont exécutés sur les graphes afin de déterminer les similarités.

Un autre framework pour la génération des descriptions OWL-S à partir d'un fichier WSDL a été développé par [Duo *et al.,* 2005]. Le processus de génération commence manuellement par la conversion du schéma XML du fichier WSDL dans une ontologie OWL intermédiaire.

Cette transformation utilise des règles telles que : (1) les éléments XSD complexes, simples et globales sont convertis en "classes : Owl", 2) tout élément local du type simple est convertit en "DatatypeProperty : Owl". Ensuite, l'ontologie intermédiaire résultante est mise en correspondances avec des ontologies de domaine existantes en utilisant des mesures de similarité basées sur le nom et la structure. Les résultats du mapping sont utilisés pour générer les descriptions OWL-S désirées à partir du fichier WSDL.

Jaeger et al. [Jaeger *et al.,* 2005] ont développé un outil pour le développement des SWS OWL-S. Leur prcessus consiste en trois étapes. En premier lieu, un modèle abstrait est généré à partir des descriptions existantes comme le fichier WSDL. Ensuite, les ontologies utiles sont identifiées pour la sémantique des éléments du service. Enfin une classification est effectuée avec ces ontologies. Leur méthodologie se focalise sur l'utilisation des algorithmes de matching pour l'identification des ontologies correspondantes.

INFRAWEBS est une autre approche proposée pour l'ingénierie des SWS [Agre *et al.,* 2007]. Elle se concentre sur la construction des descriptions sémantiques pour des Services Web nouveaux ou existants, et permet l'intégration de divers composants . L'approche INFRAWEBS est composée de différentes unités relatives au cycle de vie des SWS, comme la création, la sélection, la découverte et la composition, qui sont indispensables pour le développement réel et la mise en œuvre des Services Web Sémantiques. Cependant, INFRAWEBS souffre d'être lié à un langage ontologique spécifique (WSMO) et au style architectural de service (services basés sur le SOAP).

Dans [Heß *et al.,* 2008], les auteurs ont introduit ASSAM (Automated Semantic Service Annotation with Machine Learning) qui est un outil permettant de générer un fichier OWL-S à partir d'un document WSDL. ASSAM utilise les techniques du "machine learning" pour déterminer la catégorie du Service Web et classer les fichiers WSDL dans des hiérarchies définies manuellement. ASSAM tient compte de la documentation du Service Web ainsi que les descriptions exprimées en langage naturel.

[Lei *et al.,* 2008] ont proposé une approche pour l'annotation des fichiers WSDL. Ils affirment que leur approche peut améliorer l'efficacité du processus d'annotation en effectuant une étape initiale de matching basée sur le nom pour créer un ensemble de concepts ontologiques qui sont les meilleurs correspondant à un élément XSD donné. Ayant eu l'ensemble, un matching struturel est mis en œuvre pour trouver les meilleurs scores entre les éléments de l'ensemble. Néanmoins, cette approche est non exhaustive parce qu'elle ne fournie pas une indication claire sur les tranformations des XSD en une ontologie temporaire. Cependant, l'approche utilise des mécanismes d'adaptation trés simples qui ne peuvent pas fournir une

correspondance précise.

Une autre approche d'annotation des Services Web a été développée par [Zhang *et al.*, 2008]. Cette approche est similaire à celle proposée par [Duo *et al.*, 2005] car elle utilise les mêmes règles de transformation du schéma XML en une ontologie. Cette approche génère des services SAWSDL. Dans ce mécanisme d'annotation, l'outil de matching H-MATCH [Castano *et al.*, 2006] est utilisé pour trouver les correspondances entre l'ontologie intermédiaire qui représente le service et les ontologies partagées.

Bouchiha et al. ont proposé dans [Bouchiha *et al.*, 2012] une approche supportée par un outil pour l'annotation semi-automatique des descriptions des Services Web. Leur approche d'annotation se compose de deux principaux processus : la catégorisation et l'appariement. La catégorisation consiste à classifier les descriptions des services WSDL selon leurs domaines d'intérêt respectifs. L'appariement consiste à faire correspondare les entités du fichier WSDL aux concepts de l'ontologie de domaine. Les deux processus reposent sur des algorithmes de calcul de similarité sémantique à base du WordNet.

4.2.4 Comparaison et discussion

Nous présentons dans le tableau 4.1 une synthèse des principales caractéristiques des approches de ré-ingénierie des SW vers les SWS. Le tableau se compse de 4 colonnes où chaque colonne indique un critère de comparaison comme suit :

- La colonne "Approche" indique l'approche en question ;
- La colonne "Nature" indique si l'approche suit un processus descendant (D), ascendant (A) ou mixte (M) ;
- La colonne "Modèles" indique la source d'information ou les modèles utilisés pour la conception du système étudié ;
- La colonne "Framework visé" indique quel type de SWS sera le produit du processus de ré-ingénierie : WSMO, OWL-S, WSDL-S, etc. ;
- La colonne "Outil" indique si l'approche en question est supportée par un outil logiciel.

Tab 4.1 – Synthèse des approches d'ingénierie des SWS.

Catégorie	**Approche**	**Nature**	**Modèles**	**Framework visé**	**Outil**
A base de modèles	[Paolucci et al., 2003]	A	WSDL	OWL-S	Oui
	[Gronmo et al., 2005]	D	Modèles UML	OWL-S	Oui
	[Timm and Gannod, 2005]	A	WSDL	OWL-S	Oui
	[Acuna and Marcos, 2006]	D	Modèles UML	WSMO	Non
	[Brambilla et al., 2007]	D	BPMN et WebML	WSMO	Oui
	[Amar Bensaber and Malki, 2008]	A	WSDL	OWL-S	Oui
	[Kuropka et al., 2008]	A	WSDL et d'autres	OWL-S	Oui
A base de matching	[Patil et al., 2004]	A	WSDL et ontologies de domaine	OWL-S	Oui
	[Duo et al., 2005]	M	WSDL et ontologies de domaine	OWL-S	Oui
	[Jaeger et al., 2005]	M	WSDL et d'autres	OWL-S	Non
	[Agre et al., 2007]	M	WSDL	WSMO	Oui
	[Heß et al., 2008]	A	WSDL et d'autres	OWL-S	Oui
	[Lei et al., 2008]	A	WSDL	OWL-S	Oui
	[Zhang et al., 2008]	A	WSDL et ontologies de domaine	SAWSDL	Oui
	[Bouchiha et al., 2012]	A	WSDL	SAWSDL	Oui

La plupart des travaux présentés ci-dessus se basent sur le langage UML. UML fournit une notation graphique pour la conception et la compréhension des systèmes. Cependant, dans sa forme standard, UML ne fait pas de distinction claire entre un logiciel traditionnel et un Service Web. En plus, ces travaux n'ont pas étudié la possibilité d'étendre UML pour s'adapter aux particularités de l'architecture "SOA".

L'utilisation des diagrammes d'activité UML pour décrire le comportement des services composites ne semble pas la décision appropriée. En particulier, l'exécution du Service Web composite suit les lignes directrices de son fichier BPEL qui reflète son comportement. Cependant, UML ne définit aucun métamodèle d'exécution pour les processus métier modélisés avec ce dernier.

En outre, en dépit de son expressivité et sa pupularité, OWL-S ne s'est pas aligné avec les standards des Services Web existants. Il propose également l'utilisation de OWL pour représenter

les ontologies.

L'utilisation du processus d'annotation des Services Web permet l'annotation de certains paramètres des services existants, comme les entrées, les sorties et les contraintes mais ne permet pas d'annoter les propriétés non-fonctionnelles telles que la qualité de service ou le niveau de sécurité.

Nous pouvons donc conclure qu'il n'y a pas de standards communs pour les descriptions sémantiques et que les platesformes actuelles de développement des SWS sont limitées et dans divers cas étroitement couplées à des styles architecturaux de services spécifiques et des modèles de description sémantiques, il est essentiel donc que les nouvelles approches relatives au développement des SWS doivent répondre aux divers langages et différents modèles de description sémantique.

Durant la recherche bibliographique, nous avons remarqué que peu d'effort ont été consacré à la transformation du WSDL au WSMO. En générale, nous avons constaté que ce domaine de recherche n'a pas reçu une considération suffisante de la part de la communauté des chercheurs.

Parmi les outils d'interface graphique orienté utilisateur pour la construction et la gestion des composants conformes à l'ontologie de service WSMO, nous distinguons le WSMO-Studio et le WSMT [2]. WSMO Studio est un éditeur WSMO pour construire les aspects sémantiques des Services Web, disponible en "plug-in" Eclipse. Ses principales fonctionnalités incluent la définition d'ontologies, des Services Web, des buts("Goals") et des médiateurs.

WSMO studio spécifie les éléments WSMO dans le WSML, le langage de représentation de l'ontologie de service WSMO. L'outil prend en charge l'interaction avec l'utilisateur mais il exige que des référentiels pour les éléments à spécifier, aussi la construction est faite sur des copies locales sur la machine de l'utilisateur. À l'heure actuelle, aucune extension n'est offerte pour permettre un accès simultané et l'édition d'un référentiel partagé.

WSMT est un environnement de développement intégré (IDE) pour le développement des Services Web Sémantiques. Le WSMT vise l'assistance des développeurs des Services Web sémantiques à travers le paradigme WSMO en fournissant un ensemble homogène d'outils pour améliorer leur productivité.

Ces outils produisent des éléments WSMO à travers un processus semi-automatique où la participation de l'utilisateur est fondamentale à toutes les étapes. Ces deux outils sont basés sur une conception et compréhension préalables de l'utilisateur, aussi ces outils ne fournissent aucune indication sur le passage du fichier de description WSDL aux éléments WSMO.

2. http ://www.wsmo.org/TR/d9/d9.1/v0.2/20050425/

Par conséquent, ce processus est relativement lent, en plus du coût relatif à la complexité du Service Web. En outre, aucun outil automatique ou semi-automatique n'a été proposé pour la création de spécifications WSMO à partir du fichier de description WSDL.

Afin de remédier à ces limites, nous proposons une nouvelle approche de ré-ingénierie des Services Web à partir du fichier de description WSDL. L'approche a été proposée dans El Bouhissi et Malki [El Bouhissi *et al.,* 2009a] [El Bouhissi *et al.,* 2009b] [**?**] et se base sur l'ontologie WSMO-Lite. Notre contribution est une amélioration de cette approche [El Bouhissi and Malki, 2014a] [El Bouhissi and Malki, 2014b], et consiste en une spécification partielle des éléments de l'ontologie de service WSMO-Lite [Fensel *et al.,* 2010] en utilisant les techniques de la ré-ingénierie et le "matching" ontologique.

L'ontologie WSMO-Lite [Fensel *et al.,* 2010] a été créée en raison d'un besoin d'ontologie de service légère qui s'appuye, sur les standards les plus récentes et permet la modélisation de services selon un processus ascendant. WSMO-Lite adopte le modèle conceptuel de WSMO. WSMO utilise le langage WSML pour décrire les modèles spécifiques de domaine, cette approche permet l'utilisation de tout langage ontologique avec la syntaxe RDF. Cependant, WSMO-Lite ne permet de spécifier que les éléments "Ontologie" et "Service Web", les autres éléments "Goal" et "Mediateurs" représentent un projet en cours d'étude.

Dans le chapitre suivant, nous allons présenter notre approche de ré-ingénierie en détail.

4.3. Découverte des Services Web

Comme suite logique au succès des architectures orientées services au cours de ces dernières années, un nombre important de Services Web est disponible en ligne [Bentahar *et al.,* 2007]. Avec cette augmentation, plusieurs problèmes ont été soulevés, notamment liés à la découverte, la disponibilité, la qualité de services, etc.

Conformément au modèle orienté services, la découverte d'un service par un demandeur de services se fait comme suit : d'un côté, un fournisseur de services publie les descriptions des fonctionnalités de ses services dans un annuaire de services. De l'autre côté, un demandeur de services utilise cet annuaire pour localiser un service et par la suite l'invoquer.

Les techniques de découverte de services traditionnelles peuvent être inadéquates dans des contextes réels (un grand nombre de services et d'annuaires, manque d'expressivité des descriptions de services, etc.).

La découverte des Services Web représente un axe de recherche émergent. Au début, la découverte se faisait au niveau du registre UDDI, elle était basée essentiellement sur la recherche syntaxique des descriptions WSDL des Services Web. Mais avec le développement des technologies du Web sémantique, les techniques de découverte sont devenues essentiellement sémantiques.

Cette sémantique est apportée grâce aux ontologies, une des technologies importantes du Web sémantique. Ainsi, des agents logiciels peuvent être développés afin de raisonner sur ces ontologies rendant la découverte des Services Web, dynamique et automatique.

Peu importe le client que ce soit un humain ou une machine, il présente son besoin sous forme d'une requête pour découvrir les services publiés dans un annuaire. On désigne par la découverte automatique le fait qu'un agent logiciel peut juger qu'un Service Web est utile dans une situation et que ce jugement soit vrai.

4.3.1 Définitions

La découverte de Services Web est abordée dans un grand nombre de projets et travaux de recherche. Elle est définie par Keller et al. comme "la localisation automatique des services répondant à une requête utilisateur" [Keller *et al.,* 2004]. Booth et al., décrivent le processus de découverte comme étant "la localisation d'une description compréhensible par la machine d'un service éventuellement inconnu au préalable et correspondant à certains critères fonctionnels" [Booth *et al.,* 2004]. Toma et al. définissent la découverte comme "le processus qui prend en entrée une requête utilisateur et retourne une liste de ressources ou services pouvant combler éventuellement le besoin décrit" [Toma *et al.,* 2005]. De Paoli et al. [De Paoli *et al.,* 2008] définissent la découverte comme étant "Le processus de recherche des Services Web demandés et leur Services Web concrets pour atteindre le but des utilisateurs". Trois aspects majeurs de la découverte ressortent de ces définitions. Premièrement, l'aspect localisation des services qui consiste à trouver l'adresse à laquelle est fournie la description d'un service. Deuxièmement, l'aspect traitement des données qui indique le degré d'automatisation de la découverte. Troisièmement, l'aspect mise en correspondance ou "matching" qui compare la requête utilisateur aux services répertoriés.

4.3.2 Approches de découverte des Services Web

Avec la popularité croissante des Services Web, la découverte de Services Web pertinents est devenu un défit majeur. L'approche actuelle de découverte consiste à effectuer une recherche en fonction de mots-clés de l'UDDI. Comme d'autres techniques de recherche fondées sur les mots-clés, cette technique souffre de problèmes tels que l'ambiguïté, la synonymie, etc .. Une

solution à ce problème est d'établir des descriptions de Services Web plus significatives en annotant les descriptions de service avec des métadonnées appréhendables par la machine à partir d'ontologies partagées.

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. Pour atteindre ces buts, les Services Web doivent agir automatiquement avec le minimum d'intervention humaine, ils doivent être capables de découvrir d'autres services qui ont des fonctionnalités particulières et réalisent des tâches précises. A cause de ces contraintes, la découverte des Services Web est conçue comme un problème critique depuis la naissance de ce paradigme.

Plusieurs approches ont été proposées dans la littérature pour améliorer la découverte des Services Web . La catégorisation des approches de découverte peut être formulée selon beaucoup de critères. Dans ce qui suit, nous allons scinder ces approches en approches lexicales, approches sémantiques et approches hybrides. Les différences majeures entre ces approches sont présentées dans le tableau 4.2.

a) Approches lexicales

Ces approches se basent généralement sur la description du Service Web (WSDL) publiée dans l'annuaire UDDI. Le processus de découverte se focalise sur les aspects d'implémentation d'un service ainsi adaptée pour satisfaire les exigences des programmeurs. Ces approches consistent à rechercher des similitudes entre les différentes descriptions de service, permettant la recherche de Services Web substituables et composables comme les opérations et services similaires peuvent être découvertes sur la base de paramètres de fonctionnement.

Paolucci et al. [Paolucci *et al.,* 2002] ont proposé un algorithme d'appariement des services avec des requêtes décrites en DAML-S (Prédécesseur de OWL-S). L'algorithme reconnaît différents degrés de correspondance qui sont déterminés par la distance minimale entre les concepts dans la taxonomie de concepts. Les auteurs abordent la problématique du calcul de correspondance entre offre et demande de service du point de vue de leur description sémantique, mais donnent peu d'information quant à leur implémentation.

Dong et al. [Dong *et al.,* 2004] présentent "Woogle" un moteur de recherche de services basé sur la similarité lexicale. Woogle permet à l'utilisateur de rechercher des services similaires à un service donné, des services ayant les mêmes entrées qu'un service donné (respectivement sorties). L'algorithme proposé par cette approche, étudie la similarité entre les descriptions textuelles des opérations des services et les identifiants des paramètres d'entrée - sortie. Il

est basé sur une technique de clustering qui, en fonction de leurs entrées et sorties, relie les services à des concepts. L'algorithme de clustering est basé sur l'heuristique suivante : "les paramètres ont tendance à indiquer le même concept s'ils apparaissent souvent ensemble". Des règles d'association sont définies entre les identifiants des paramètres.

Gomez et al. [Gomez *et al.,* 2004] proposent une approche utilisant le traitement du langage naturel et permettant de chercher des Services Web WSMO correspondants à un but donné, formulé sous forme textuelle en langue anglaise. Cette approche consiste en un référentiel de Goals prédéfinis et supportée par un outil nommé "GODO". La recherche se fait dans la liste des buts prédéfinis. Le but WSMO le plus proche sera envoyé au WSMX, qui est un environnement d'exécution pour la découverte et composition de services WSMO et par conséquent le Service Web WSMO retrouvé sera renvoyé à l'utilisateur. Cette approche fait le bon usage des capacités du Framework WSMO, mais elle ne peut pas être appliquée pour d'autres langages sémantiques comme le OWL-S, qui n'ont pas de tels éléments de représentation de but.

L'approche de Zachos et al. [Zachos *et al.,* 2008] a comme objectif d'aligner les exigences des utilisateurs aux services opérationnels disponibles. L'idée-clé de ce travail est de créer un cahier des charges spécifiant les exigences des utilisateurs, transformer ces exigences en des requêtes pour l'interrogation de l'annuaire de services et modifier le cahier des charges élaboré initialement en fonction des services retournés. La spécification des exigences est faite en langage naturel en utilisant le modèle de spécification d'exigences VOLERE. Les spécifications obtenues sont analysées par l'outil lexical "WordNet" pour enlever les ambigüités sémantiques. Pour découvrir les services logiciels qui réalisent les exigences des utilisateurs, ces spécifications sont transformées en des requêtes en utilisant le langage "XQuery". Ces requêtes sont utilisées pour extraire les spécifications WSDL qui sont sémantiquement similaires aux exigences formulées à partir de l'annuaire de services. A la fin, Zachos et al. proposent de réaligner les exigences initialement formulées aux Services Web trouvés.

Seekda [3] [Seekda, 2009] est un outil qui emploie les techniques de clustering des paramètres de fonctionnement. Cette approche extrait la sémantique des services à partir des fichiers WSDL, permettant l'échange d'exécution des services similaires et composables. Le portail Seekda propose une recherche basique de services par mots-clés et une recherche avancée qui permet de découvrir les services à partir d'un ensemble de balises, découvrir les fournisseurs par pays, découvrir les Services Web les plus utilisés ou faire une naviguation à travers les Services Web récemment découverts et sélectionnés. Les services peuvent être marqués pour

3. Le site de Seekda a été retiré de l'internet.

une réutilisation/consultation ultérieure et ils peuvent être invoqués directement à partir du portail (pour tester directement le Service Web). Pendant la phase de découverte, un appariement est effectué entre des mots-clés définis par l'utilisateur et des mots-clés associés aux services publiés dans l'annuaire. Les Services Web peuvent être découverts suite à un appariement lexicale avec les mots-clés qui sont leurs associés. Les résultats peuvent être classés par pertinence, disponibilité, pays, fournisseur ou date de publication.

L'approche de Driss et al. [Driss *et al.,* 2010] s'appuie sur le modèle des cartes pour capturer les exigences fonctionnelles et non-fonctionnelles des utilisateurs. Des mots-clés sont extraits et sont utilisés pour rechercher directement les descriptions WSDL dans des annuaires UDDI en utilisant le moteur de recherche "Service-Finder". Les résultats sont classés en fonction des propriétés non-fonctionnelles. La Carte [Rolland and Prakash, 2000] est un formalisme qui s'inscrit dans le domaine d'ingénierie des méthodes, une discipline de conceptualisation, de construction et d'adaptation de méthodes, de techniques et d'outils pour le développement des systèmes

Dans [Karray *et al.,* 2013], Karray et al. proposent une approche dont l'architecture est basée sur les registres des Services Web appelés communautés. Un registre représente un groupe de Services Web ayant le même domaine d'intérêt et ils différent les uns des autres en fonction de leur propriétés non-fonctionnelles afin de limiter l'espace de recherche. Enfin pour trouver le service optimal correspondant à la requête du client, ils utilisent les algorithmes génétiques pour permettre une recherche intelligente.

Enfin, Sangers et al. [Sangers *et al.,* 2013] proposent un Framework pour la recherche des Services Web Sémantiques en utilisant les techniques du traitement du langage naturel. Leur approche permet de rechercher, à partir d'un ensemble de Services Web Sémantiques, une correspondance avec une requête de l'utilisateur constituée de mots-clés. En précisant le but de la recherche avec des mots-clés, les utilisateurs finaux n'ont pas besoin d'avoir des connaissances sur les langages sémantiques.

b) Approches Sémantiques

Ces approches utilisent les descriptions sémantiques des Services Web et les techniques du Web Sémantique pour automatiser le processus de découverte. Ces approches utilisent des ontologies prédéfinies, en identifiant les similarités sémantiques entre ces ontologies, les Services Web Sémantiques relatifs seront découverts. Le processus se focalise sur les aspects conceptuels d'un service.

Sycara et al. [Sycara *et al.,* 2002] proposent une approche pour la description des capabilités et requêtes d'un agent et leur matchmaking. Le moteur du matchmaking est basé sur plusieurs filtres de différentes complexités et précisions que les utilisateurs peuvent choisir. Cette approche est supportée par un outil nommé "LARKS".

Dans [Oundhakar *et al.,* 2005], les auteurs proposent un Framework qui aborde le problème de la découverte des services dans un scénario où les fournisseurs de services et les demandeurs peuvent utiliser des termes de différentes ontologies. Leur approche repose sur l'annotation des registres des services (pour un domaine particulier) et exploiter de telles annotations durant la découverte. Cette approche est supportée par un outil nommé "METEOR-S".

L'approche de [Stollberg and Kerrigan, 2007] est basée sur le Framework WSMO, le concept but est directement exploité pour la découverte de service. Une interface utilisateur graphique, visualise une hiérarchie de buts structurés sous forme de graphiques par rapport à leur similarité sémantique.

L'approche de [da Silva *et al.,* 2009] combine deux concepts orientés vers l'utilisateur : les objectifs et les tâches. En s'appuyant sur le Framework WSMO, des experts définissent les ontologies de domaine et de tâche. Les tâches fournissent des connaissances sur les descriptions de comportement, et sont mises en correspondance avec les services. Des requêtes peuvent être exprimées directement comme des buts ; puis elles sont sémantiquement comparées avec les buts de l'ontologie WSMO.

[Bener *et al.,* 2009] propose une architecture qui réalise l'appariement sémantique des Services Web basée sur les descriptions d'entrée et de sortie, ainsi que les conditions préalables et les effets. Dans le processus de matchmaking des Services Web Sémantiques, les descriptions OWL-S sont supposées et la mise en relation se fait au niveau conceptuel en utilisant les règles du langage SWRL (A Semantic Web Rule Language combining OWL and RuleML).

Dans [Chabeb and Tata 2010], Chabeb et al. proposent une approche qui combine SAWSDL et OWL-S. Afin de définir la catégorie des informations aux annotations sémantiques, une ontologie technique est construite en utilisant le OWL-S. Les annotations sont enrichies avec des liens vers l'ontologie technique et les requêtes des utilisateurs sont comparées à cette description enrichie.

c) Approches hybrides

L'approche de [Klusch and Kapahnke, 2009] combine les techniques lexicales avec sémantiques correspondantes afin d'améliorer considérablement les performances de la découverte.

Les entrées des utilisateurs sont exprimées en utilisant des modèles OWL-S qui sont directement comparées avec des descripteurs de service OWL-S. Une question ouverte ici est la mesure dans laquelle l'appariement sémantique est précis et efficace pour un but assorti.

Dans [Aljoumaa *et al.*, 2010], les auteurs se sont intéressés à étendre les approches existantes pour la description des services intentionnels. Cette approche est consacrée à la spécification et la mise en œuvre des descripteurs intentionnels de service, où l'annotation est utilisée comme une technique d'extension sémantique. Leur approche propose une extension du SAWSDL et utilise deux types d'ontologies : "ontologie verbe" représentant les concepts syntaxiques et sémantiques liés aux verbes, et "l'ontologie produit", qui est une ontologie de domaine contenant les concepts définissant un vocabulaire commun pour tous les objets manipulés dans le domaine des affaires.

4.3.3 Comparaison et discussion

Dans cette section, nous avons choisi d'abord de résumer, dans le tableau 4.2 quelques critères d'évaluation et ensuite comparer les principales approches relatives à la découverte des Services Web sus-mentionnées selon ces critères.

Les caractéristiques de ces initiatives sont synthétisées à l'aide de quatre critères que nous avons retenu car ils représentent, à notre sens, les critères les plus pertinents. Il s'agit, en l'occurrence, des critères suivants :

- La colonne "Framework utilisé" indique quel cadre de travail des SWS sera utlisé pour le processus de découverte : WSMO, OWL-S, WSDL-S, etc ;
- La colonne "Technique du Matchmaking " indique quelles sont les techniques utilisées pendant le processus de découverte ;
- La colonne "Outil" indique si l'approche en question est supportée par un outil implémenté. La mention "Oui"/"non" indique la présence ou l'absence d'un outil logiciel. Les noms de quelques outils sont portés sur le tableau.
- La colonne "Goal" indique si l'approche en question se base sur le concept de "Goal".

Le tableau présente aussi les avantages et les inconvénients des différentes catégories d'approches.

Tab 4.2 – Synthèses des différentes approches relatives à la découverte des Services Web.

	Approche	Framework	Technique de matchmaking	Outil	Goal	Avantages	Inconvénients
Lexicales	[Paolucci et al., 2002]	OWL-S	Technique d'appariement	WOOGLE	Non	Simple et largement utilisée	Ne permet pas la récupération des SW avec des fonctionnalités similaires
	[Dong et al., 2004]	/	Technique du clustering	WOOGLE	Non	Simple et largement utilisée	Ne permet pas la récupération des SW avec des fonctionnalités similaires
	[Gomez et al., 2004]	WSMO	Analyser le goal décrit en langage naturel	GODO	Oui		
	[Zachos et al., 2008]	WSMO	Expansion de requêtes + WordNet	Oui	Non	Basée sur des standards comme WSDL et UDDI	Nécessitent l'interaction humaine
	[Seekda,2009]	/	Techniques du clustering	Non	Non		
	[Driss et al. 2010]	/	Techniques de classification	Non	Non		
	[Karray et al. 2013]	/	Clustering + algorithmes génétiques	Non	Non	Recherche par mots-clés est plus familière à l'utilisateur	Ne convient pas pour le traitement automatique
	[Sangers et al., 2013]	WSMO	Traitement du langage naturel	Oui	Non		
Sémantique	[Sycara et al. ,2002]	/	Description des capabilités et requêtes des agents et leur matchmaking	LARKS	Non	Réduit l'effort manuel de découverte	Technique complexe, Tagging sémantique des SW peut être nécessaire
	[Oundhakar et al., 2005]	WSDL-S	Annotation avec une ontologie de domaine	METEOR-S	Non		
	[Stollberg et al., 2007]	WSMO	Similarité sémantique	Oui	Goal Template		
	[Da Silva Santos et al., 2009]	WSMO	Mapping des tâches aux services	Oui	Oui	Technique efficace et fiable	Mots-clés ne suffisent pas pour spécifier avec précision les besoins d'information de l'utilisateur
	[Bener et al., 2009]	OWL-S	Matching sémantique des SW	/	Non		
	[Chabeb and Tata, 2010]	SAWSDL et OWL-S	Catégorisation + Similarité Semantique	Oui	Non		
Hybride	[Klush and Kapahnke, 2009]	OWL-S	Léxicale et Sémantique	Oui	Non	Plus efficace	L'utilisateur doit avoir connaissance du OWL-S/SAWSDL
	[Aljoumouaa et al., 2010]	SAWSDL	Annotation	Oui	Non		

Nous avons brièvement présenté les principales approches proposées dans la littérature , en fournissant une description de base pour le lecteur intéressé. Toutefois , elles présentent plusieurs limites importantes. Tout d'abord, certains cadres de découverte proposés sont basés sur une requête de l'utilisateur exprimée dans une description sémantique en modèle spécifique comme le OWL- S , WSMO ou WSDL-S. En conséquence, ces approches exigent que l'utilisateur final doit avoir une connaissance préalable des Services Web Sémantiques et des outils de description et les détails de mise en oeuvre qui rendent leur utilisation difficile pour les utilisateurs finaux.

Certaines de ces approches adoptent souvent des technologies de mots-clés correspondants à localiser les Services Web publiés. Bien que la recherche par mots-clés est une technique largement utilisée pour la récupération de l'information, elle n'utilise pas de sémantique explicite et bien définie. Les mots-clés utilisés pour récupérer des informations pertinentes n'ont pas une formalisation explicite et, par conséquent , ne permettent pas d'améliorer les résultats de recherche. La description des capabilités des Services Web est essentielle pour la classification, la découverte et l'utilisation d'un service. En effet, les demandeurs de services invoquent les services sur la base des descriptions de service découverts. Ce procédé lexicale renvoie des résultats de découverte qui ne correspondent pas avec exactitude à la demande d'un service donné. En conséquence, seulement quelques services qui ont une correspondance lexicale exacte de la demande de service peuvent être récupérés et considérés pour la sélection. Ainsi, le processus de découverte est également limité par sa dépendance à l'intervention humaine pour choisir le service approprié en fonction de sa sémantique. Les mots-clés utilisés pour récupérer les informations pertinentes ne possèdent pas de formalisation explicite et, par conséquent, ne permettent pas d'améliorer les résultats de la recherche.

D'autres propositions consistent en leurs approches sémantiques correspondantes. En effet, le fournisseur de service, ainsi que le demandeur, utilisent des ontologies de domaine pour construire les descriptions sémantiques du service. Le matchmaker sémantique utilise les ontologies de domaine afin de déterminer leur degré de correspondance sémantique. La plupart des approches proposées supposent que tous les deux, fournisseur de services et demandeur de services utilisent la même ontologie de domaine pour décrire les capabilités du service, ce qui n'est pas applicable dans les scénarios du monde réel. Pour surmonter l'hétérogénéité de ces ontologies, il est nécessaire d'utiliser des techniques du mapping ontologique afin de résoudre l'interopérabilité sémantique .

La proposition de [Klusch and Kapahnke, 2009] est une illustration claire de la nécessité de

combiner les deux techniques de recherche lexicale et sémantique afin d'améliorer considérablement les performances de découverte. Une question ouverte ici est la mesure dans laquelle l'appariement sémantique est précis et efficace pour le but assorti.

Toutes les approches présentées n'emploient pas le concept de traçabilité. En outre, ces approches n'utilisent pas explicitement le concept de but ("Goal") à l'exception des approches de [Gomez *et al.,* 2004], [Stollberg and Kerrigan, 2007] et [da Silva *et al.,* 2009]. Cependant, [Gomez *et al.,* 2004] font usage des mots-clés qui ne permettent pas la récupération des Services Web avec des fonctionnalités similaires. [Stollberg and Kerrigan, 2007] se focalise sur un modèle de but ("Goal template"). Cependant, il existe plusieurs catégories de buts et différentes significations peuvent être associées à la notion de but. [da Silva *et al.,* 2009] s'appuyent sur les représentations ontologiques des Services Web Semantiques en termes de tâches et buts où les experts définissent les ontologies de domaine et de tâches, seulement, cette proposition souffre du manque d'ontologies pertinentes.

En résumé, aucune des approches décrites dans le tableau 4.2 , ne fournit une solution complète selon les dimensions indiquées. Ainsi, un mécanisme efficace pour la découverte des Services Web est nécessaire pour améliorer la capacité de trouver les Services Web, les plus pertinents, en comparant les demandes de service avec les descriptions des services offerts.

Les motivations à mener une approche de découverte plus efficace des Services Web Sémantiques nous guide à éviter tout besoin d'entremise dans la procédure de découverte. L'intervention des utilisateurs, à un stade du processus, présente toujours une limite dans une approche de découverte. L'exigence d'une assistance humaine est de plus en plus inacceptable dans un tel domaine (de plus en plus automatique). De plus, une interprétation humaine peut ne pas être assez pertinente, cela nécessite un bon niveau d'expertise. Une intervention humaine, même experte, peut coûter plus chère par exemple en termes de temps, qu'une approche automatique bien élaborée.

Afin de répondre aux limites des approches existantes, nous proposons une nouvelle approche de découverte basée sur les préférences des utilisateurs et la traçabilité. Cette approche a été proposée en version préléminaire dans [El Bouhissi and Malki, 2011] et améliorée dans [El Bouhissi *et al.,* 2014c] [El Bouhissi *et al.,* 2014d]. L'approche procède par une capture d'un but de l'utilisateur à partir d'une page Web, puis, à la recherche des Services Web pertinents correspondant à ce but avec l'utilisation des algorithmes de calcul de similarité sémantique. Les services découverts peuvent satisfaire les demandeurs de services par l'intermédiaire de l'application des préférences de l'utilisateur. La prise en compte des préférences de l'utilisateur permet d'affiner la découverte de Services Web récupérés .

Notre proposition utilise huit (8) algorithmes de calcul de similarité sémantique durant tout le processus de découverte. Ainsi, les Services Web correspondant le mieux au profil et au contexte de l'utilisateur final sont récupérés. Dans notre approche, nous indiquons également l'intérêt du concept de traçabilité qui représente une base de faits, une mémoire renfermant à tout instant les traces d'exécution, cette base est enrichie au fur et à mesure par les réponses retournées à l'utilisateur, ainsi, cette base, consiste à sauvegarder l'historique d'exécution du processus de découverte en vue de récupéper les exécutions ultérieures similaires pour fournir un résultat rapidement à l'utilisateur.

4.4. Conclusion

Dans ce chapitre, nous avons commencé par introduire la terminologie de la ré-ingénierie. Ensuite, nous avons présenté un état de l'art de la ré-ingénierie et de la découverte des Services Web et nous avons examiné les différents travaux présentés selon leurs approches.

Pour examiner ces approches, nous avons commencé par identifier les critères les caractérisant. Finalement, nous avons utilisé ces critères pour évaluer les approches étudiées.

La discussion autour de l'état de l'art de la ré-ingénierie et de la découverte des Services Web a démontré le besoin, aussi bien d'une ré-ingénierie plus efficace et d'une découverte plus précise basées sur une expressivité sémantique plus explicite.

Dans le chapitre suivant, nous allons présenter en détail nos contributions relatives à la ré-ingénierie des Services Web vers les Services Web Sémantiques, ainsi, que la découverte des Services Web basée sur les préférences des utilisateurs.

Chapitre

5 Contributions à la Réingénierie et à la Découverte des Services Web Sémantiques

Sommaire

5.1. Introduction

Les chapitres précédents nous ont permis d'étudier les Services Web Sémantiques et les principales approches relatives à leur description et leur découverte. L'analyse des solutions existantes a montré un certain nombre de limites qui concernent principalement les architectures des ontologies actuelles, le manque de méthodologie pour définir les Services Sémantiques et le manque d'outils permettant de prendre en compte la complexité inhérente à l'utilisation des technologies aussi variées que diverses permettant de supporter les architectures des Services Sémantiques.

Les Services Web sont basés sur des standards tels que WSDL pour la description de leurs propriétés fonctionnelles (interfaces, opérations, inputs, outputs, etc). Le manque de sémantique dans le fichier de description WSDL empêche la découverte ainsi que l'invocation et la composition automatiques. Certains efforts tels que SAWSDL proposent une annotation sémantique aux descriptions des standards des Services Web comme le WSDL afin de remédier à ce manque. Mais, comme il est indiqué dans le chapitre 3, les approches étudiées dans l'état de l'art, y compris SAWSDL, présentent encore soit des lacunes à combler soit des inconvénients à éviter.

Partant de ce constat, nous avons investigué le domaine d'intégration des Services Sémantiques à la recherche d'une méthodologie et d'une solution flexible d'intégration.

Dans nos travaux, nous nous intéressons à étendre les approches existantes en termes de description et de découverte de Services Web Sémantiques. Ce chapitre présente en détail nos contributions relatives à la ré-ingénierie et à la découverte des Services Web. Ainsi, la première contribution, consiste à générer des Services Web Sémantiques WSMO à partir du fichier de description WSDL. Quant à la deuxième, elle consiste à proposer une approche pour la découverte des Services Web Sémantiques basée sur les préférences des utilisateurs et la traçabilité (Historique de l'exécution). Les principes généraux de nos contributions sont l'intégration de la sémantique dans la description ainsi que la découverte des Services Web en utilisant le matching ontologique.

La section 5.2 présente un rappel de la problématique. Les sections suivantes présentent une explication détaillée de nos contributions et nous clôturons ce chapitre par une conclusion.

5.2. Rappel de la problématique

Rappelons que notre travail consiste en la ré-ingénierie et la découverte des applications Services Web dans le contexte spécifique des projets du laboratoire EEDIS de l'université de Sidi-Bel-Abbés (Algérie).

Dans l'état actuel des choses, les Services Web souffrent d'un certain nombre de limites et de lacunes. Il s'agit fondamentalement du manque de méthodologies à mettre en œuvre pour définir et de découvrir les Services Sémantiques. Nous pouvons surmonter cette insuffisance en utilisant la technologie du Web Sémantique. Nous pouvons enrichir sémantiquement les Services Web avec des descriptions, compréhensibles par la machine, décrivant ce que fait l'application et comment peut-on l'utiliser.

En outre, avec des ontologies capables de décrire les services fournis, nous pouvons apporter

les services dans une manière qui est à la fois riche et compréhensible par la machine. Ceci permet une découverte de services plus efficace et rapide que celle utilisée par l'UDDI.

Comme nous avons vu au chapitre 3, plusieurs approches ont été proposées dans la littérature pour la conception des Services Web Sémantiques tels que OWL-S, WSMO, WSDL-S et SAWSDL. Cependant, les technologies proposées n'exploitent pas les informations contenues dans les interfaces du service telles que le WSDL et exigent l'intervention d'un utilisateur humain et des experts du domaine pour la conception et la validation.

En ce moment, l'application pratique des technologies des SWS est toujours plutôt limitée et ceci est dûe à plusieurs raisons :

- La complexité élevée des deux approches OWL-S et WSMO ;
- L'insuffisance d'ontologies de domaine et l'indisponibilité des outils avancés supportant WSMO ;
- L'absence d'applications pilotes se concentrant sur les besoins journaliers des consommateurs, des citoyens, de l'industrie etc., qui peuvent démontrer les avantages d'employer la sémantique.

Afin de produire de nouvelles technologies Sémantiques de Services Web, accessibles aux utilisateurs, le niveau des outils soutenant ces technologies devrait être sensiblement élevé.

Dans le cadre de notre travail, nous nous focalisons sur l'approche WSMO-Lite qui est une approche relativement récente et toujours en cours de développement.

L'approche WSMO-Lite [Fensel *et al.,* 2010] a été proposée comme une extension légère de l'approche WSMO, elle est adoptée au cours des dernières années dans plusieurs projets par des consortiums. Cependant, en investiguant l'approche WSMO-Lite, nous avons soulevé certaines insuffisances vis-à-vis de sa conception qui se résument dans les points suivants :

- La procédure de conception des ontologies WSMO (ou WSMO-Lite) se fait manuellement en grande partie, ceci représente une charge pénible aux experts du domaine surtout avec des Services Web assez consistants, ce qui explique l'insuffisance des modèles et l'indisponibilité des Data-Sets, notamment pour les développeurs du projet eux-mêmes ;
- L'indisponibilité des outils de conception de l'ontologie WSMO (WSMO-Lite) qui exploitent les informations des interfaces du service, le WSDL, ni d'ailleurs les registres UDDI ou les messages SOAP. Durant notre phase de recherche préliminaire, nous avons remarqué qu'il existait deux outils de conception et d'exécution des Services WSMO

appelé WSMO Studio et WSMT qui permettent à l'utilisateur d'introduire toutes les données pour les différents éléments du WSMO d'une manière assistée, l'utilisateur était guidé par des menus et des boites de dialogue, l'utilisateur était donc amené à introduire toutes les informations nécessaires et par conséquent, l'utilisateur doit avoir des connaissances approfondies sur le domaine et le service lui même.

Cette façon de procéder représente à notre avis une charge lourde et consommant beaucoup de temps d'un côté et d'un autre côté, elle nécessite des experts du domaine pour sa mise en œuvre et sa validation. A cette étape, nous nous sommes posés la question suivante : "Ne serait-il pas intéressant dans un premier temps d'automatiser certaines tâches de la conception du WSMO afin d'éviter aux utilisateurs certaines manipulations inutiles et minimiser le temps d'exécution".

Notre travail répond en grande partie à cette question, et met l'accent sur l'exploitation de l'interface du Service Web, à savoir, le fichier WSDL pour la conception de l'ontologie de service WSMO-Lite.

5.3. Approche de ré-ingénierie des Services Web vers les Services Web Sémantiques

La proposition [El Bouhissi and Malki, 2014a] [El Bouhissi and Malki, 2014b] que nous allons détailler par la suite, est une amélioration de [El Bouhissi *et al.,* 2009a], [El Bouhissi *et al.,* 2009b] et [El Bouhissi and Malki, 2009c], elle est supportée par un outil nommé WSDL2WSMO-Lite (Cf. Fig 5.1), un système permettant d'exploiter les informations de la description d'un Service Web existant (WSDL) en vue d'établir une présentation partielle des éléments "Ontologie" et "Service Web" de l'ontologie de service WSMO-Lite à travers un processus de ré-ingénierie.

WSDL2WSMO-Lite est un environnement de développement graphique pour créer des descriptions sémantiques des Services Web selon le modèle conceptuel du WSMO-Lite. L'outil est dédié aux utilisateurs – fournisseurs des Services Web et des applications sémantiques de Services Web, qui voudraient convertir leurs services en Services Web Sémantiques conformément aux spécifications du modèle WSMO-Lite.

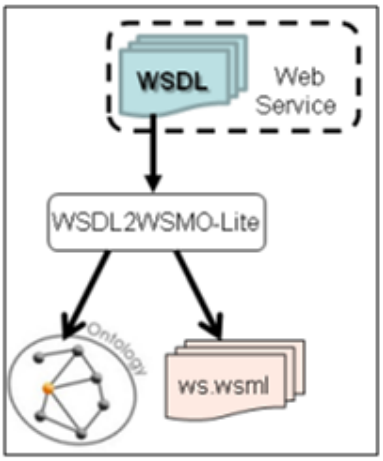


Fig 5.1 – Approche de ré-ingénierie des Services Web vers les Services Web Sémantiques

Notre approche de ré-ingénierie des Services Web se déroule principalement en deux grandes étapes :

- Une étape de rétro-ingénierie pour l'extraction et le traitement des informations utiles encapsulées dans le fichier WSDL ;
- Une étape d'ingénierie pour la construction des éléments du WSMO-Lite selon la syntaxe du langage WSML.

Dans le cadre de notre contribution relative à la ré-ingénierie, nous nous sommes limités à la spécification uniquement des éléments "Ontologie" et "Service Web" de l'ontologie WSMO-Lite.

5.3.1 Pourquoi le WSDL ?

Le fichier WSDL est un document XML pour la description des Services Web qui peut être stocké dans le registre UDDI ou servi d'un fournisseur de service. Les descriptions du WSDL sont prévues pour décrire le Service Web à un client de sorte que le client puisse se servir du service.

Nous avons choisi pour notre contribution le fichier WSDL (Cf. Fig 5.2) comme source d'information en considérant que le code des applications Web est souvent, protégé par un serveur d'application, n'est pas accessible aux utilisateurs et même, il peut être crypté. Ainsi, ce choix se justifie par les considérations suivantes :

```
...
<types>
      Définition des types de données utilisées par le service web
</types>
<message>
      Définition des messages utilisés par le service web
</message>
<portType>
      Définition des opérations exécutées par le service web
</portType>
<binding>
      Définition des protocoles de communication utilisés par le service web
</binding>
...
```

Fig 5.2 – Structure de base d'un document WSDL.

- WSDL est indépendant de toute plateforme et de tout langage de programmation, il est exprimé en XML, un format d'échange de données standard qui contribue à la résolution de l'interopérabilité syntaxique. Le fait que le WSDL est exprimé en XML, ceci permettra l'exploitation de son contenu en particulier les schémas XML ;
- WSDL répond à 3 questions essentielles pour pouvoir utiliser un Service Web :

  a. Comment peut-on y accéder ? (adresse, protocole) ?

  b. Quelles sont les opérations mises à disposition ?

  c. Quels sont les messages échangés ? (paramètres) ?

  Toutes ces informations nécessitent d'être décrites sémantiquement puis seront très utiles pour la conception du WSMO-Lite.
- Le fichier WSDL est généré à partir du Service Web lui-même, les informations utilisées pour la description d'un service et les moyens pour pouvoir l'utiliser, reflètent les données et les actions du Service Web ;
- Les schémas XML présentent les paramètres du Service Web et leurs types de données respectifs ;
- Nous avons remarqué aussi dans les petits fragments d'exemples que nous avons trouvé, qu'il y avait une liaison forte entre les informations du fichier WSDL et les éléments du WSMO-Lite, ce qui nous a poussé à faire la recherche d'un cadre formel pour le passage du WSDL vers le WSMO-Lite.

Le standard WSDL présente des lacunes de précision dans la description d'un service. Ces lacunes sont liées au niveau d'expressivité faible de la description syntaxique, car elle est

limitée à l'énumération des opérations et à la description des types des paramètres d'entrée et de sortie associés.

Cette description ne caractérise pas la sémantique de la fonctionnalité accomplie par le service. Pour pallier le manque de sémantique de WSDL, plusieurs approches proposent de rajouter une couche au dessus de WSDL complétant la description syntaxique par des précisions sémantiques.

Par exemple le Service Web d'Amazon.com, décrit en WSDL [1], a pour objectif la vente de livre. Il permet à un utilisateur de parcourir les catalogues d'Amazon, pour rechercher des livres et les acheter. Cependant, un utilisateur pourra s'en servir pour parcourir les catalogues en guise de recherche d'information, sans acheter.

Syntaxiquement, le terme "rechercher" est différent du terme "acheter", cependant sémantiquement ces deux concepts sont liés, le premier étant un pré-requis du second. Une recherche par mots-clés d'un service permettant de rechercher un livre ne retournera pas les services décrits permettant d'acheter un livre, par la suite l'utilisateur ne sera pas informé de tous les services potentiellement pertinents. Une subtilité de ce genre sera contournée si le service fourni une description sémantique plus élaborée qu'un fichier WSDL.

5.3.2 Le système WSDL2WSMO-Lite

Notre proposition vise d'un côté la conception de l'ontologie WSMO-Lite elle-même et d'un autre côté la conception de l'élément "Services Web".

La figure 5.3 illustre notre approche en détail. Le processus de ré-ingénierie passe principalement par deux (2) grandes étapes, une étape de rétro-ingénierie, et une étape d'ingénierie, ces deux étapes sont réparties en (5) phases, depuis le chargement du fichier WSDL jusqu'à la validation des éléments du WSMO-Lite produits.

Il est à noter que notre contribution s'appuie sur le concept de calcul de la similarité sémantique qui permet d'enrichir sémantiquement les éléments produits et résoudre le probléme de standardisation.

1. http ://webservices.amazon.com/AWSECommerceService/AWSECommerceService.wsdl

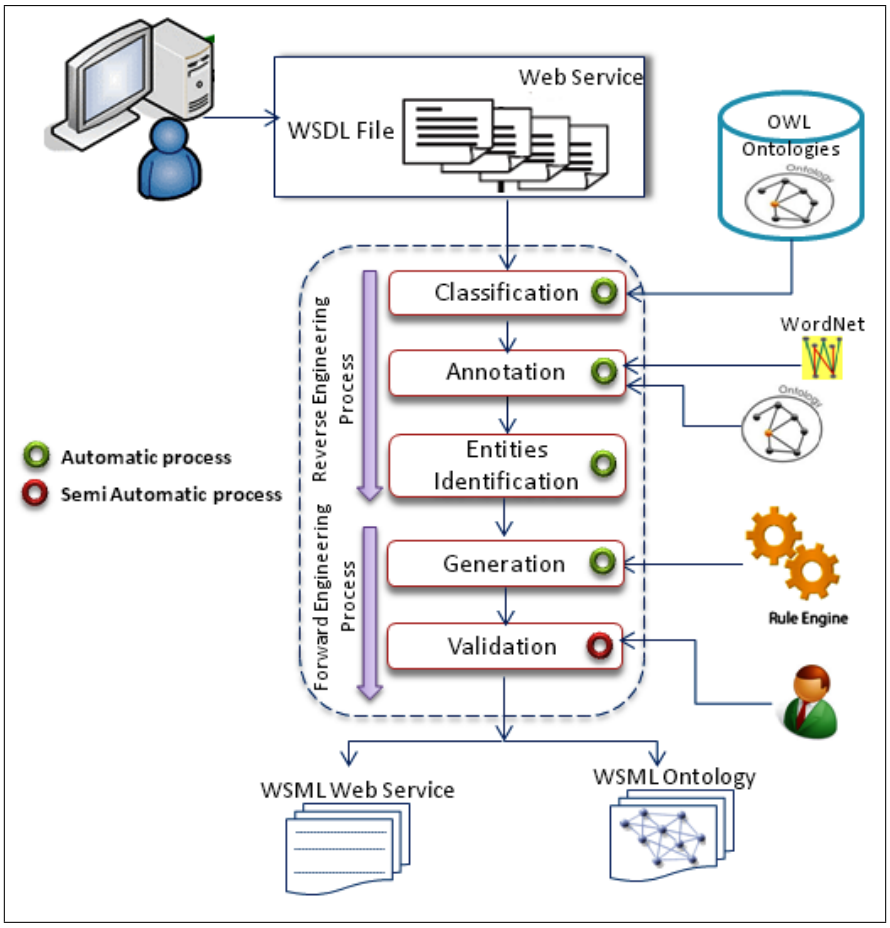


Fig 5.3 – Processus de ré-ingénierie proposé.

L'identification de la similarité dans les ontologies est un concept fondamental et adopté par plusieurs techniques telles que le regroupement, la fouille de données (data mining), le Web Sémantique et en particulier, le domaine de la recherche de l'information. Cette dernière repose largement sur des mesures pour l'identification de la similarité entre les documents. Les rapports ontologiques entre les mots peuvent être détectés par un processus de calcul de similarité entre des paires d'objets contenus dans l'ontologie.

Dans la littérature, plusieurs travaux sur la mesure de similarité sémantique entre les objets d'une ontologie ont été développés dans différents contextes. Les méthodes qui utilisent WordNet [2] peuvent fournir des résultats excellents [Meng *et al.*, 2013].

Le calcul de similarité sémantique joue un rôle central dans notre processus de ré-ingénierie, cette manipulation porte sur les informations retenues du Service Web et les ontologies du domaine. Elle renvoie une valeur numérique (de 0 à 1) indiquant si les deux éléments ont un degré, élevé ou bas, de similitude.

2. WordNet est une base de données lexicale qui organise les noms et les verbes dans des concepts (synset) en hiérarchies de relations "is-a". Chaque concept est décrit par une brève close.

Pendant le processus de ré-ingénierie, nous utilisons la mesure de Wu et Palmer [Wu and Palmer, 1994] qui est largement utilisée dans la littérature et propose une mesure qui calcule la similarité entre deux concepts en regardant la longueur du chemin entre eux dans la taxonomie WordNet. Cette mesure suppose que la similitude entre les deux concepts est la fonction de la longueur du chemin et de la profondeur dans les mesures basées sur le chemin. La mesure de Wu et Palmer [Wu and Palmer, 1994] a l'avantage d'être simple à implémenter et d'avoir aussi de bonnes performances comparativement à d'autres mesures de similarité.

La mesure de similarité de [Wu and Palmer, 1994] est basée sur le principe suivant : Etant donnée une ontologie formée par un ensemble de nœuds et un nœud racine R. Soit X et Y deux éléments de l'ontologie dont nous allons calculer la similarité. Le principe de calcul de similarité est basé sur les distances (N1 et N2) qui séparent les noeuds X et Y du noeud racine et la distance qui sépare le concept subsumant [3] (CS) de X et de Y du noeud R.

Nous avons calculé les similarités sémantiques pour identifier les ontologies de domaine indispensables à notre processus de ré-ingénierie et enrichir les éléments du WSMO-Lite. Aussi, nous avons fixé au préalable un seuil pour la distance sémantique qui représente un degré de confiance. Le seuil est une valeur comprise entre 0 et 1 ; la valeur 1 indique que les deux entités sont totalement similaires.

Enfin, le lecteur trouvera une étude détaillée des différents algorithmes de calcul de similarité sémantique utilisés dans nos contributions en Annexe B.

Le processus de ré-ingénierie dans son ensemble passe par deux grandes étapes, depuis le chargement du fichier WSDL jusqu'à la validation des éléments du WSMO-Lite produits, (1) Une étape de rétro-ingénierie qui comprend trois (3) phases et (2) une étape étape d'ingénierie qui consiste en deux (2) phases.

Comme le fichier WSDL est notre principale source d'information exprimée en langage XML, dans ce qui suit, nous allons donner quelques explications sur les schémas XML, les notions à notre sens indispensables à notre démarche. Les étapes du processus seront présentées par la suite.

Le XML a révolutionné la manière dont les données peuvent être échangées et représentées à travers le web. Quand deux systèmes désirent échanger des données, une rupture de document XML est souvent le mécanisme le plus simple pour atteindre les résultats désirés.

3. Le concept subsumant c'est le concept commun le plus spécifique.

Les schémas XML fournissent un moyen de définir les contraintes sur la syntaxe et la structure d'un document XML. La spécification du schéma XML est substantielle. Nous avons défini une approche pour le développement d'un mapping entre le schéma XML et l'élément ontologie du WSMO-Lite en se basant sur un sous ensemble de types utilisés communément et définitions d'éléments utilisés par le schéma XML. Nous allons voir dans ce qui suit comment un sous ensemble de composants de schéma XML est apparié aux différents composants de l'ontologie WSMO. Les composants que nous avons pris en considération sont : les types simples, les types complexes, les attributs et les restrictions sur les éléments.

On distingue deux familles de types :

- Les types simples qui caractérisent le contenu d'un noeud ou d'un attribut.
- Les types complexes sont utilisés pour décrire les autres formes de contenu.

Ceci nous amène à distinguer différents modèles de contenu pour un élément selon la nature de ses noeuds fils autorisés :

- Vide : aucun nœud fils.
- Simple : ne contient que des nœuds textuels.
- Complexe : que des sous éléments ; ou des sous éléments avec des nœuds textuels.

Dés qu'un élément possède un attribut, il est considéré comme étant de type complexe même si son contenu est vide ou simple.

Les types simples. Le schéma XML définit une large gamme de datatypes intégrés, par exemple, "string", "integer", "boolean", "float", "decimal", etc. Ces types représentent des exemples d'une forme de la définition des types. Une autre forme des définitions des types simples peut être utilisée pour fournir les contraintes sur les valeurs des types intégrés, par exemple, il est peut être nécessaire de limiter les valeurs acceptées du type de données positiveInteger pour une valeur maximale particulière.

Le listing suivant donne un exemple d'un type simple :

```
<xsd :element name="age">
   <xsd :simpleType>
        <xsd :restriction base="xsd:positiveInteger">
             <xsd :maxExclusive value="35">
        </xsd :restriction>
   </xsd :simpleType>
</xsd :element>
```

Listing 5.1 – XML : Exemple de type simple

Comme le montre le listing 5.1 , l'élément âge du type entier est limité à la valeur 35.

Les types complexes. Ils peuvent êtres utilisés pour :

- Définir un type de données composé de sous-éléments d'autres datatypes.
- Etendre ou limiter la définition d'un type complexe existant.

En plus, les valeurs des éléments peuvent être accompagnées des contraintes sur leurs valeurs. Le listing 5.2 illustre un exemple explicatif.

```
<xsd:complexType name="employe">
    <xsd:element name="name" type="xsd:string">
        <xsd:element name="age">
            <xsd:simpleType>
                <xsd:restriction base="xsd:positiveInteger">
                    <xsd:maxExclusive value="35">
                </xsd:restriction>
            </xsd:simpleType>
        </xsd:element>
</xsd complexType>
```

Listing 5.2 – XML : Exemple de type complexe

L'exemple suivant (listing 5.3) présente un type complexe "job" qui possède un attribut "jobid" de type "string".

```
<xsd:complexType name="job" type="jobDesc">
    <xsd:attribute name="jobid" type="xsd:string"/>
</xsd:complexType>
```

Listing 5.3 – XML : Exemple d'un type complexe avec attribut

5.3.3 L'étape de rétro-ingénierie

Elle représente l'étape la plus importante du processus de ré-ingénierie, car elle permet d'identifier les éléments indispensables à la génération des éléments de l'ontologie WSMO-Lite. Cette étape comporte trois (3) principales phases.

a) Phase de catégorisation

Etant donné un ficher de description WSDL et un ensemble d'ontologies de domaine en entrée, il est intéressant de faire la catégorisation de ce service pour connaitre son domaine d'intérêt.

La classification peut être bénéfique pour plusieurs activités du cycle de vie des Services Web. Elle peut permettre de faciliter, d'optimiser, d'automatiser l'efficacité et l'efficience du processus d'annotation des Services Web. Ainsi, dans notre processus, cette phase est indispensable

car elle permet de décider sur quelle ontologie utiliser pour la deuxième phase (Phase d'annotation).

Lorsque plusieurs ontologies sont disponibles, les similitudes entre les ensembles doivent être calculées en comparant l'ensemble des entités du fichier WSDL avec tous les concepts de chaque ontologie. Sur la base de ces similitudes, le système peut choisir parmi les ontologies disponibles, celle pour exécuter l'algorithme d'annotation. L'ontologie de domaine choisie détermine la catégorie du document WSDL.

La phase de catégorisation consiste à extraire une ontologie à partir de la partie schéma XML du fichier WSDL et la comparer avec l'ensemble des ontologies de domaine existantes dans le référentiel à travers un calcul de la mesure de similarité sémantique. les résultats de l'appariement sont affichés à l'utilisateur pour qu'il puisse choisir l'ontologie de domaine dont le score est le plus élevé, qui veut dire l'ontologie la plus proche du domaine du Service Web.

b) Phase d'annotation

Cette phase permet d'annoter le fichier de description WSDL avec l'ontologie de domaine selectionnée durant la phase précédente. Annoter, c'est accompagner un texte des notes, des remarques, des explications, des commentaires pour aider à le comprendre. Actuellement et avec ce grand volume d'information, il est difficile d'annoter manuellement des millions de ressources mises à la disposition des utilisateurs.

Il existe trois types d'annotation, (1) Manuelle : lorsque le document est analysé par un spécialiste du domaine ou un documentaliste, (2) Automatique : lorsque cette tâche est réalisée complètement par la machine, et (3) Semi automatique lorsqu'une partie de l'annotation est réalisée automatiquement et l'intervention d'un spécialiste de domaine est nécessaire pour l'autre partie.

En considérant la description d'annotation, maintenant nous allons nous intéresser aux outils d'annotation existants qui peuvent nous servir à annoter les fichiers de description WSDL. Le mécanisme d'annotation utilisé est automatique et il représente celui proposé par l'approche SAWSDL. SAWSDL définit comment on peut ajouter des annotations sémantiques dans les différentes parties d'un document WSDL comme les entrées, les sorties, les structures des messages, les interfaces et les opérations. L'ajout est réalisé en mettant des attributs "modelReference", "liftingSchemaMapping" ou bien "loweringSchemaMapping".

Dans ce qui suit, nous allons nous intéresser uniquement à l'annotation des informations contenues dans le fichier WSDL et utiles pour notre approche, celles utilisant l'attribut "modelReference". Les quatre ajouts essentiels du modelReference sont :

– Annotation d'interfaces : Cet ajout sert à se référer à un concept d'une ontologie pour décrire une interface. Dans l'exemple du listing 5.4, l'interface "Order" est associée à un concept "electronics" d'une ontologie quelconque.

```
<wsdl:interface name="order" sawsdl:ModelReference="http://exemple.org/
categorization/products/electronics">
</wsdl:interface>
```

Listing 5.4 – Annotation d'interface

– Annotation des opérations : Cet ajout sert à se référer à un concept d'une ontologie pour décrire une opération. Dans l'exemple du listing 5.5, l'opération "Order" est associée à un concept "RequestPurchaseOrder" d'une ontologie quelconque.

```
<wsdl:operation name="order" sawsdl:ModelReference="http://www.w3.org/
2002/w/sawsdl/spec/ontology/Purchaseorder#RequestPurchaseOrder>
</wsdl:operation>
```

Listing 5.5 – Annotation d'opérations

– Annotation des schémas XML : L'annotation du schéma implique plusieurs annotations :

1. Annotation des Types simples : Les types simples peuvent être annotés en incluant un attribut "modelReference" sur l'élément xs : simpleType. Dans l'exemple du listing 5.6, un élément ou un attribut dont le type est la confirmation et décrit par la notion de confirmation de commande sémantique dans le modèle référencé.

```
<xs:simpleType name="Confirmation"
  sawsdl:modelReference="http://www.w3.org/2002/ws/
␣␣sawsdl/spec/ontology/purchaseorder#OrderConfirmation">
</xs:simpleType>
```

Listing 5.6 – Annotation d'un type simple

2. Annotation des types complexes : Il existe deux principales techniques pour annoter les types complexes qui peuvent être résumées comme suit : l'annotation à bas niveau qui consiste à annoter un élément membre ou un attribut du type complexe. L'annotation de haut niveau consiste en une annotation du conteneur du type complexe (le type complexe lui-même). Un type complexe peut être annoté à la fois selon les deux techniques haut et bas niveau. Ces annotations sont indépendantes les unes des autres :

   - En ce qui concerne l'annotation à bas niveau, tous les éléments membres et attributs d'un type complexe peuvent être annotés. Dans certains cas, les membres d'un type complexe correspondent aux concepts dans un modèle sémantique. Dans ce cas, les éléments membres et attributs peuvent être annotés par l'ajout d'un attribut "modelReference" à l'élément du schéma ou attribut.

Dans la séquence du listing 5.7, chaque élément de niveau inférieur aura une annotation depuis le modèle sémantique contenant des concepts, à savoir "Quantity" et "ProductCode", qui décrivent chacun des composants du type complexe.

```
<xs:complexType>
  <xs:sequence minOccurs="1" maxOccurs="unbounded">
    <xs:element name="quantity" type="xs:integer"
      sawsdl:modelReference="http://www.w3.org/2002/ws/
␣␣␣␣␣␣sawsdl/spec/ontology/purchaseorder#Quantity"/>
    <xs:element name="UPC" type="xs:string"
      sawsdl:modelReference="http://www.w3.org/2002/ws/
␣␣␣␣␣␣sawsdl/spec/ontology/purchaseorder#ProductCode"/>
  </xs:sequence>
</xs:complexType>
```

Listing 5.7 – Annotation de bas niveau d'un type complexe

- En mode d'annotation à haut niveau, les types complexes eux-mêmes sont annotées avec le "ModelReference".

  Dans l'exemple du listing 5.8, le type complexe dans son ensemble a été annoté avec une référence à la notion "OrderRequest". OrderRequest décrit un concept composé par le groupe d'éléments "quantity" et "UPC".

```
<xs:complexType sawsdl:modelReference="http://www.w3.org/2002/ws/
sawsdl/spec/ontology/purchaseorder#OrderRequest">
  <xs:sequence minOccurs="1" maxOccurs="unbounded">
    <xs:element name="quantity" type="xs:integer"
    <xs:element name="UPC" type="xs:string"
  </xs:sequence>
</xs:complexType>
```

Listing 5.8 – Annotation de haut niveau d'un type complexe

3. Annotation des éléments avec "ModelReference" : Une déclaration d'élément peut être annotée en incluant un "modelReference" sur l'élément xs : élément. Un exemple d'annotation d'un élément global en utilisant l'attribut "modelReference" est illustré dans le listing 5.9.

```
<xs:element name="OrderRequest"
sawsdl:modelReference="http://www.w3.org/2002/ws/
sawsdl/spec/ontology/purchaseorder#OrderRequest">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="customerNo" type="xs:integer" />
<xs:element name="orderItem" type="item" minOccurs="1" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

Listing 5.9 – Annotation des éléments avec ModelReference

Dans cet exemple, l'annotation indique que l'élément "OrderRequest" est décrit par le concept "OrderRequest" qui représente la référence dans le modèle sémantique. Cet exemple est très similaire à l'annotation de haut niveau du type complexe.

4. Annotation d'attribut avec "ModelReference" : Un attribut peut être annoté en incluant un modelReference sur les xs : élément d'attribut. Si l'élément "quantity" dans l'exemple du listing 5.10 a été défini comme un attribut, un modelReference pourrait être appliqué comme suit :

```
<xs:attribute name="quantity" type="xs:integer"
    sawsdl:modelReference="http://www.w3.org/2002/ws/
sawsdl/spec/ontology/purchaseorder#Quantity"/>
```

Listing 5.10 – Annotation d'un attribut avec ModelReference

c) Phase d'extraction

La phase d'extraction constitue une partie importante de notre processus de ré-ingénierie, elle permet d'identifier les éléments indispensables à la conception de l'ontologie WSMO-Lite.

L'information pertinente extraite est stockée dans un fichier XML (info.xml) qui sera utilisé pour les étapes suivantes. Les informations considérées sont :

- Les informations extraites à partir du schéma XML du fichier WSDL qui seront utilisées pour la conception des éléments "Ontologie" et "Service Web" de l'ontologie WSMO-Lite.
- Les informations qui se trouvent entre les balises <opération>...</opération> et <message>...</message> qui contribuent à la conception d'une partie de l'élément "Service Web" de l'ontologie WSMO-Lite.

Rappelons que l'extraction des informations se fait à partir du fichier WSDL annoté, les informations à extraire du fichier WSDL ne constituent pas les informations de base mais plutôt les entités ajoutées avec l'attribut "modelReference", c'est à dire celles provenant de l'ontologie de domaine.

5.3.4 L'étape d'ingénierie

Cette étape permet l'ingénierie des éléments WSMO-Lite à partir des informations identifiées et analysées pendant l'étape de rétro-ingénierie.

a) Phase de génération

Cette phase du processus d'ingénierie consiste à traduire les informations annotées et extraites à partir du fichier WSDL et raffinées par la suite en des spécifications WSML.

Comme déjà mentionné au début, nous nous sommes contentés dans notre approche de la spécification partielle des éléments "Ontologie" et "Service Web" du WSMO-Lite, ce qui fourni des spécifications incomplètes, nécessitant l'intervention d'un expert humain pour compléter les informations utiles.

Nous nous sommes intéressés durant cette phase de trouver une méthodologie formelle pour traduire les informations extraites du fichier XML intermédiaire (info.xml) en spécifications WSML[4]. Nous avons considéré dans cette section uniquement la spécification partielle des éléments "Ontologie" et "Service Web", nous laissons donc en perspective la continuité du projet car un tel projet -en cours d'étude- nécessite à notre sens plusieurs années et groupes de recherche.

Afin de passer du modèle WSDL à l'ontologie WSMO-Lite, nous proposons des règles de mapping simples (Cf. Fig 5.4) pour quelques informations du fichier WSDL, et nous fournissons entre autre, des exemples de fragments de fichier WSDL explicatifs pour montrer l'application des différentes règles du mapping. Il est indispensable de rappeler que l'extraction se fait à partir du fichier WSDL annoté.

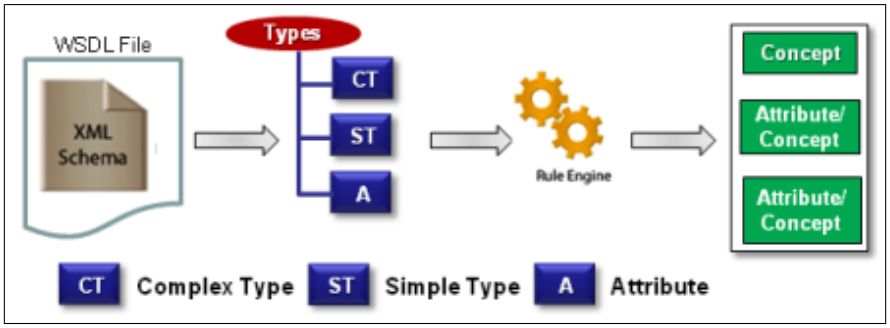


Fig 5.4 – Mapping des éléments du schéma XML.

Règle 1 : Les types simples

Quand une définition de type simple est utilisée pour la création d'un nouveau type basé sur une restriction d'un type intégré, nous créons un nouveau concept avec le type intégré

4. Pour plus de détail sur le langage WSML, consulter l'annexe A

correspondant et l'axiome qui définit la restriction.

Pour un type simple, nous considérons deux cas :

a. Si le type simple est lié directement à la racine du schéma XML, il se transforme en un Concept.

b. Par contre, si le type est simple et contenu dans un type complexe, il se transforme en un attribut de ce type complexe.

Par exemple, la définition du type simple du listing 5.1 présenté au-dessus et utilisée pour définir l'élément "âge" se transforme en un concept de type "integer".

Règle 2 : Les types complexes

Les définitions des types complexes peuvent contenir des sous composants qui peuvent être une fusion d'éléments, d'attributs et d'autres types complexes.

Les types complexes sont représentés comme des concepts dans WSMO-Lite, leurs sous composants sont définis comme suit :

a. Les sous composants avec un type simple intégré deviennent des attributs du concept créé avec le même type intégré.
b. Les sous composants avec un simple type qui n'est pas intégré, deviennent des attributs avec le même type du type simple transformé.
c. Les types complexes peuvent à leur tour contenir d'autres types complexes. Les règles ainsi, peuvent être appliquées de façon récursive sur tous les composants complexes du type complexe défini.

En se référant au listing 5.2 concernant "Les types complexes" et présenté précédemment, le type complexe "employe" se transforme en un concept avec deux attributs : "Name" de type "string" et "âge" de type "integer".

Règle 3 : Les attributs

Un élément Attribut d'un schéma XML peut avoir comme parent, le schéma lui-même ou un type complexe. Quand le parent est le schéma, l'élément attribut se transforme en une définition de concept dans l'ontologie WSMO-lite. Par contre, si le parent est un type complexe, cet attribut deviendra un attribut du type complexe transformé.

En se référant au listing 5.3 relatif aux attributs, l'attribut "jobid" devient un attribut pour le concept "jobdesc".

Règle4 : Les opérations

Chaque opération est attribuée à un Service Web atomique particulier qui prendra le même nom que celui de l'opération et possédera ses paramètres d'entrée ainsi que ses paramètres de sortie.

Nous allons considérer un exemple concret de fragment de fichier WSDL.

```
<message name="getTempRequest">
    <part name="zipcode" type="xsd:string"/>
</message>
<message name="getTempResponse">
    <part name="return" type="xsd:float"/>
</message>
<portType name="TemperaturePortType">
<operation name="getTemp">
 <input message="tns:getTempRequest"/>
<output message="tns:getTempResponse"/>
 </operation>
</portType>
```

Listing 5.11 – Fragment d'un fichier WSDL - Opération

Dans notre cas, nous aurons le Service Web nommé "getTemp", nous avons utilisé donc la balise <opération> et nous avons récupéré l'élément de l'attribut "name".

Règle 5 : Les paramètres d'entrée/sortie

Les attributs <message> des balises <opération> permettent de récupérer les paramètres d'entrée et de sortie du service, dans notre cas, "getTempRequest" fait référence au paramètre d'entrée : "zipcode", et "getTempResponse" fait référence au paramètre de sortie : "return".

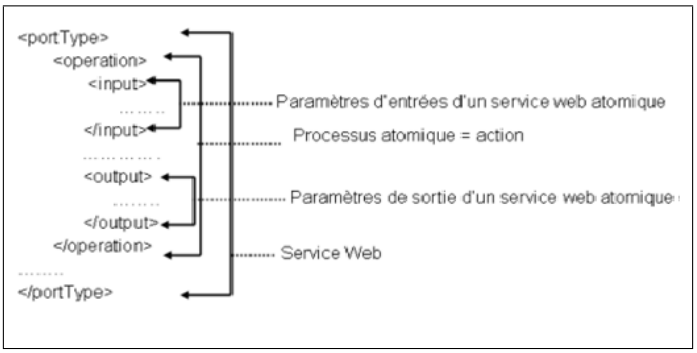


Fig 5.5 – Transformation d'une opération et ses paramètres d'entrée/sortie

Ces paramètres sont récupérés en faisant la correspondance des attributs des balises inputs (respectivement output) des balises <opération> avec les balises <part> (des balises <message>).

Il en résulte, un seul Service Web nommé "getTemp" qui fait le calcul, reçoit en entrée une variable "zipcode" de type string et renvoie en sortie une variable "return" de type float.

Règle 6 : Les variables partagées

L'ensemble des variables d'entrée et de sortie, représente les variables partagées (shared variables) que nous allons intégrer dans la définition du Service Web, dans notre cas, nous aurons deux variables, "zipcode" et "return". Nous pouvons établir la capabilité du service en fonction de ces variables.

La démarche que nous avons adoptée pour l'établissement des différents "mapping" procède par un parcours du fichier WSDL. L'algorithme 5.1 illustre le processus d'extraction des types de données à partir du schéma XML et les "mappings" correspondants et l'algorithme 5.2 illustre le processus d'extraction des opérations.

Algorithme 5.1 Extraction des types de données à partir du fichier WSDL

Entrée : <S> Un schéma XML du fichier WSDL.
Sortie : Fichier info.XML
Début
1 : Tant qu'il y a des définitions de schéma XML
2 : Pour tout $s \in S$
3 : Si s est un attribut alors s devient un concept
4 : Sinon Si s est un type simple alors s devient un concept
5 : Sinon Si s est un type complexe alors s devient un concept
6 : FinSi
7 : FinSi
8 : FinPour

Fin

Complexité. L'algorithme d'extraction des informations (Cf. Algorithme 5.1) à partir du schéma XML s'éxécute tant qu'il y a une définition de schéma XML dans WSDL et pour chaque entrée de type complexe il s'exécute une autre fois de manière recursive . Ainsi la complexité de cet algorithme est de l'ordre *O(m*n)* (où m est le nombre d'éléments et n le nombre d'éléments de type complexe).

Algorithme 5.2 Extraction des opérations du fichier WSDL

Entrée : Fichier WSDL
Sortie : Fichier info.XML
Début
1 : Pour toute entrée faire
2 : Si a est une <operation> de l'entrée <PorType> alors
3 : Récupérér les input et output
4 : Utiliser les input/output pour récupérer les paramètres des éléments <message>
5 : FinSi
6 : FinPour

Fin

Complexité. L'algorithme d'extraction des opérations à partir du fichier WSDL (Cf. Algorithme 5.2) s'éxécute tant qu'il y a une balise <operation> de la balise <portype> dans WSDL. Ainsi la complexité de cet algorithme est de l'ordre *O(m*n)*(où m est le nombre de balises <PortType> et n le nombre de balises <operation> contenues dans la balise <PortType>).

Règle 7 : Les restrictions

Toute restriction se transforme en un axiome. En se référant au listing 5.2 relatif aux types simples : l'élément simple "âge" qui est du type PositiveInteger ne doit pas dépasser la valeur "35". L'attribut maxExclusive se transforme en opérateur arithmétique ">=".

Nous présentons le tableau (Cf. Tableau 5.1) récapitulant quelques attributs de restriction et leurs transformations respectives en opérateurs arithmétiques (Cet échantillon de restriction est choisi selon notre expérimentation, par contre cette démarche peut être extensible, ce qui ouvre d'autres perspectives à l'avenir).

Tab 5.1 – Mapping des restrictions - sur les types - en opérateurs arithmétiques.

Restriction/données	**Types de données**	**Opérateur correspondant**
minInclusive	Integer	$<$
maxInclusive	Integer	$>$
minExclusive	Integer	$\leq$
maxExclusive	Integer	$\geq$
minlength	String	$<$
maxlength	String	$>$
length	String	$=$

Règle 8 : Capabilité du Service Web – les pré-conditions

Afin que le Service Web puisse s'exécuter, nous devrions lui fournir toutes les informations nécessaires, ce qui veut dire que tous les paramètres d'entrée du Service Web doivent avoir des valeurs.

Pour l'exemple de la règle 4, pour que le service "getTemp" puisse s'exécuter, le paramètre "zipcode" doit avoir une valeur.

Règle 9 : Capabilité du Service Web – les assumptions

Comme déjà mentionné dans le chapitre 3, les assumptions représentent l'état du monde désiré avant l'exécution du service, nous proposons de mettre comme assumption les restrictions sur les éléments, plus simplement les axiomes.

Pour l'exemple de la règle 4, pour que le service "getTemp" puisse s'exécuter, le paramètre "zipcode" doit avoir une valeur.

b) Phase de validation

Cette phase constitue la dernière phase de notre processus de ré-ingénierie. Les étapes précédentes peuvent introduire des concepts et/ou des relations erronées, donc une phase de validation automatique ou semi-automatique du résultat obtenu est nécessaire. Cette étape est souvent réalisée manuellement, mais dans certains cas, la validation peut être automatique ou semi-automatique.

Après la génération des éléments "Ontologie" et "Service Web", ils doivent être validés afin de s'assurer de leur compatibilité à la syntaxe du langage WSML.

Si ces éléments ne sont pas compatibles avec la syntaxe du langage WSML, un expert du domaine pourrait apporter quelques modifications pour corriger la situation.

Le processus de ré-ingénierie produit des descriptions incomplètes des deux éléments "Ontologie" et "Service Web", d'autres informations telles que les propriétés non-fonctionnelles sont introduites par un expert de domaine d'une manière assistée.

5.4. Découverte des Services Web : Approche basée sur les préférences des utilisateurs

La deuxième approche proposée [El Bouhissi and Malki, 2014b] dans cette thèse vise à permettre une recherche efficace des Services Web correspondants à une requête formulée par l'utilisateur.

Le travail présenté ici est une amélioration du travail proposé dans [El Bouhissi and Malki, 2011] [El Bouhissi and Malki, 2014a] qui vise à améliorer la recherche des Service Web selon les préférences de l'utilisateur.

La correspondance des besoins des utilisateurs avec des capabilités du service est une question fondamentale dans l'ingénierie orientée services. Notre analyse montre clairement que les préférences des utilisateurs deviennent un élément important dans les approches de découverte de service. Nous avons néanmoins souligné trois aspects importants lors du processus de découverte : (1) Formulation des besoins des utilisateurs, (2) Parcours des référentiels des buts(Goal) et des services et chercher les correspondances, et enfin, (3) Amélioration du processus de découverte en utilisant la notion de traçabilité.

5.4.1 Le système de découverte

Dans cette section, nous présentons en détail le cadre de découverte proposé. Nous donnerons des descriptions de notre approche de découverte fondée sur la sémantique pour trouver les Services Web appropriés selon ls préférences des utilisateurs.

Comme il s'agit de découverte, il est nécessaire d'avoir une infrastructure basée sur un référentiel de recherche de service (WSR : Web Service Repository) ; ce référentiel doit contenir toutes les informations relatives aux Services Web, à savoir leurs descriptions exprimées en langage WSML y compris leurs capabilités, leurs emplacements, leurs interfaces, etc. Ces descriptions sont analysées et lues par notre système pour extraire toutes les informations pertinentes et indispensables pendant le processus de découverte afin de trouver les Services Web Sémantiques en mesure de satisfaire les préférences des utilisateurs.

En outre, nous mettons en place un référentiel des ontologies (OR : Ontology Repository). "OR" contient les ontologies WSML relatives aux Services Web du référentiel WSR.

Notre approche est fondée sur la sauvegarde de l'historique d'exécution de l'application. Nous supposons que pendant le processus de découverte, des utilisateurs peuvent avoir les mêmes préférences, par conséquent, des mêmes goals peuvent être formulés préalablement et leur exécution est enregistrée dans un référentiel appelé DBH (DataBase History) ce qui rend la recherche plus rapide et plus efficace. L'idée de garder l'historique d'exécution, permet de garder la trace de tous les utilisateurs du système. D'un coté, elle permet de répertorier les exécutions répétées en vue d'accélérer la recherche et d'un autre coté, elle permet d'avoir des indicateurs en vue d'améliorer le système.

Initialement, ie avant l'exécution, la base DBH est vide, puis au fur et à mesure des réponses

données par l'utilisateur, elle se voit modidiée pour devenir une base de donnée dynamique au cours de l'exécution. Cette base représente une base de faits. Les faits consistent en des "goals". C'est en fin de cycle d'exécution que le système fait appel à cette base pour donner des explications "Pourquoi" de la déduction trouvée ie, le Service Web. Cette base est structurée en un fichier database nommé DBH.

L'idée de la découverte par rapport aux buts permet d'afficher une liste de tous les Services Web appropriés. Chaque Service Web trouvé est accompagné par le détail de ses informations, en particulier, sa capabilité, qui peuvent être consultées.

La figure 5.6 représente l'architecture principale du système de découverte proposé qui comprend trois principales étapes : (1) Une catégorisation des services basée sur la sémantique ; (2) Une Découverte sémantique et (3) Une sélection des meilleurs Services.

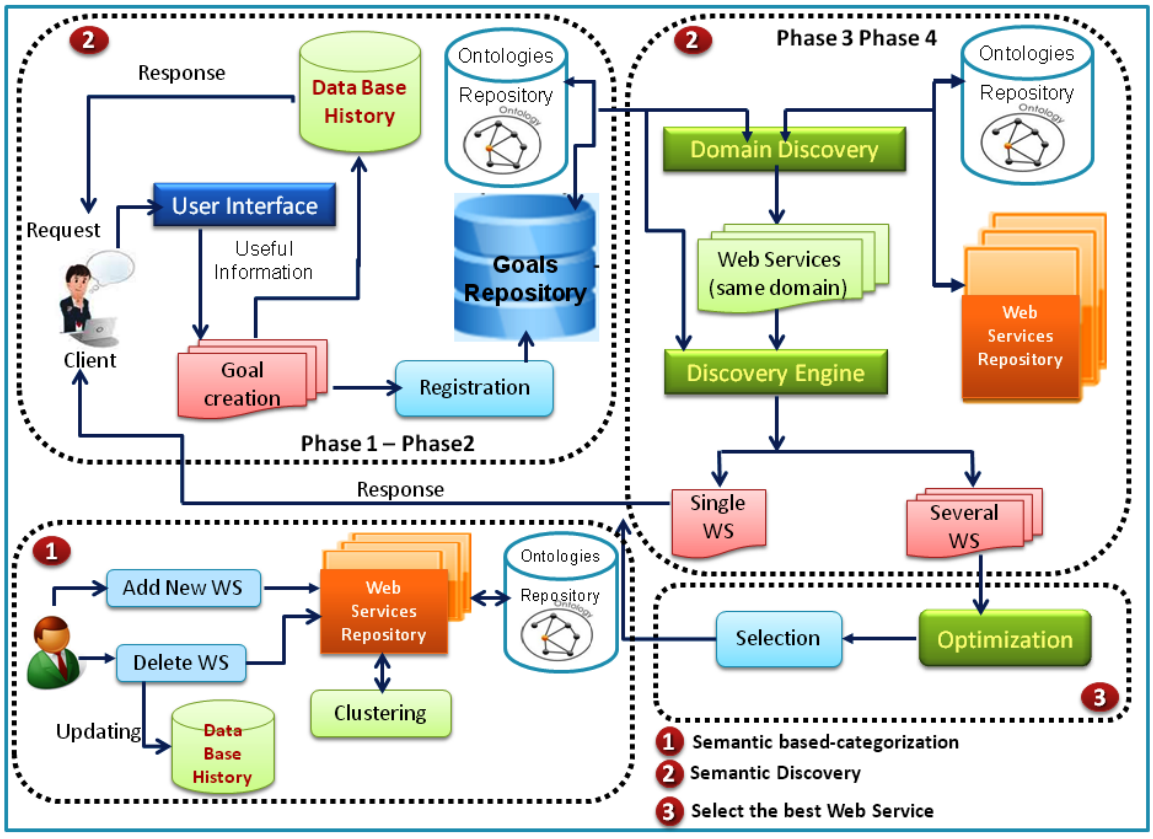


Fig 5.6 – Architecture générale du système de découverte proposé

La première étape concerne l'évolution et la maintenance du système et la mise à jour de l'ensemble des référentiels. Les deux dernières étapes concernent le processus de découverte dans son ensemble : découverte des Services Web pertinents et la sélection du meilleur Service susceptible de répondre aux besoins des utilisateurs.

Du point de vue demandeur de service, notre système offre une interface graphique conviviale simple pour faciliter le processus de découverte. Par conséquent, l'architecture globale et les technologies mises en œuvre sont transparentes pour l'utilisateur. Pour trouver les Services Web appropriés correspondants aux besoins de l'utilisateur, nous établissons un appariement pair à pair du but formulé avec les Services Web WSML. À la fin du processus, si plusieurs Services Web sont retournés, nous utilisons les propriétés non-fonctionnelles pour sélectionner le meilleur Service Web.

Une clé importante de notre processus de découverte est le calcul de la mesure de similarité sémantique qui vise à quantifier combien le "goal" et le Service Web se ressemblent. Pendant le processus de découverte, nous avons utilisé huit (8) algorithmes de calcul de similarité sémantique à base du WordNet [Meng *et al.,* 2013]. Plus de détail sur les algorithmes de calcul de similarité sémantique est disponble dans l'annexe B.

Nous avons également défini un seuil pour la mesure de similarité. Le seuil représente la précision requise ; ce seuil représente une valeur comprise entre 0 et 1 ; la valeur 1 indique qu'il y a une équivalence sémantique totale entre deux concepts. L'utilisation d'un seuil reflète la tolérance admissible pendant les calculs. Plus la valeur de seuil est plus grande, plus les résultats sont exactes. Si la similitude calculée est inférieure au seuil prédéfini, le Service Web ne correspond pas à la requête et ne devrait pas être considéré.

Etant donné une requête, la première série de Services relatifs au domaine du Goal est sélectionnée. Par la suite, un matching entre les éléments de cette série et le Goal formulé est établi avec un algorithme de calcul de la mesure de similarité sémantique en fixant un certain seuil. Si la mesure calculée est inférieure au seuil prédéfini, le Service Web est considéré comme non pertinent à la requête et il est éliminé donc de l'espace de recherche des Services.

Or, selon le but formulé, nous pouvons ne pas avoir des Services Web susceptibles de répondre aux besoins des utilisateurs. Dans ce cas, une réponse est tout de suite renvoyée à l'utilisateur pour l'informer qu'il n'y a pas de Services Web qui répondent à ses besoins.

A la fin du processus, si le système retourne plusieurs Services Web, nous utilisons les valeurs des propriétés non-fonctionnelles contenues dans chaque élément Service Web WSML pour choisir le meilleur Service selon des critères que nous avons fixés arbitrairement.

Le processus de découverte en question est modélisé par un diagramme de séquence exprimé en UML (Unified Modeling Language) et présenté à la figure 5.7. Le client soumet une requête au système, qui découvre les Services Web appropriés à partir du WSR. Les résultats sont classés avant qu'ils ne soient présentés à l'utilisateur final. A la fin, le meilleur Service Web en

fonction de certains critères est renvoyé à l'utilisateur.

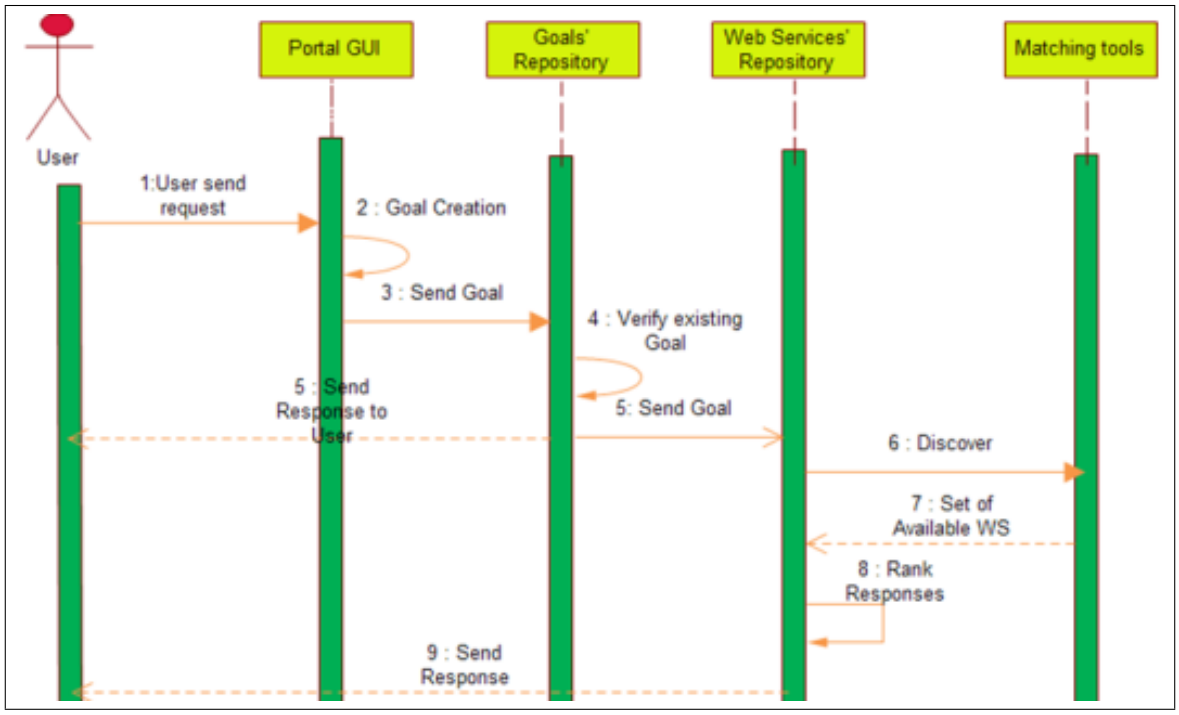


Fig 5.7 – Diagramme de séquence UML représentant le haut niveau d'interaction

5.4.2 Etapes du processus de découverte

L'approche de découverte des Services Web, proposée consiste principalement en trois étapes. Dans ce qui suit, nous allons voir ces étapes avec plus de détail. Une étude de cas corespondant à un scénario du monde réel relatif au domaine des analyses médicales est représentée dans le chapitre suivant.

a) Catégorisation Basée sur la sémantique

Cette étape est généralement exécutée en hors-ligne (off-line) et orientée - batch - , elle implique d'importants volumes de données en cours de traitement avec peu ou pas d'intervention, généralement un expert du domaine qui exécute ce traitement. Cette étape porte sur la mise à jour des différents référentiels, à savoir, le WSR (Référentiel des Services), le OR (Référentiel des ontologies) et la DBH (Base de données de l'historique d'exécution).

Cette étape est effectuée à chaque fois il y a de nouveaux Services Web à insérer dans le référentiel WSR ou à supprimer du référentiel WSR. Le processus d'insertion et/ou de suppression est un traitement qui peut être effectué par l'administrateur du domaine.

Une fois un nouveau Service Web WSML est ajouté au référentiel WSR, l'ontologie WSML correspondante est ajoutée aussi à l'OR. En outre, si un Service Web existant est supprimé du référentiel WSR, l'ontologie correspondante est également supprimée de l'OR et le référentiel

DBH est donc mis à jour.

Après un traitement d'ajout ou de suppression, les Services Web du référentiel WSR sont immédiatement regroupés(classés) en domaines. L'efficacité de la découverte peut être grandement améliorée après que les services ont été bien classés par domaine.

La méthode de clustering ci-dessous est une technologie qui réorganise un ensemble de données dans les différents groupes en fonction de certaines normes de similitude. L'objectif du regroupement des Services Web en catégories de domaine est d'organiser les Services Web du référentiel WSR en n catégories (Cf. Fig 5.8) et chaque catégorie est identifiée par une clé unique qui est le nom de la catégorie.

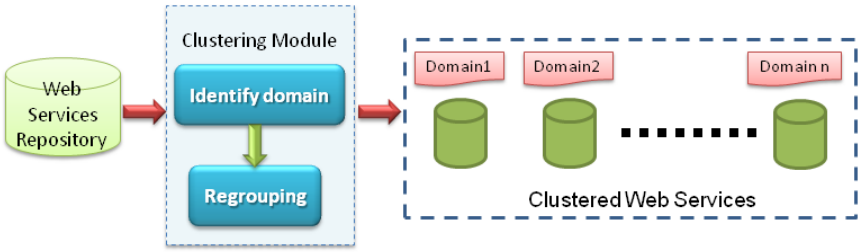


Fig 5.8 – Procédure du clustering des Services Web

Le mécanisme de regroupement est adapté de manière à faciliter la gestion et à améliorer l'efficacité de la découverte. Le principe du clustering comme le montre l'algorithme 5.3, consiste à parcourir tous les Services Web du référentiel WSR, extraire le paramètre de domaine à partir des propriétés non-fonctionnelles des Services Web ou des ontologies correspondantes, puis créer des groupes de domaine qui contiennent de tels Services Web où chaque Service Web est placé dans le groupe de domaine approprié. Le groupe de domaine est créé progressivement au cours du processus de classification. A la fin de cette phase, nous gardons une liste de catégorie de domaines où chaque domaine contient les Services Web appropriés. Il est à noter que pendant ce processus, nous considérons uniquement les domaines de haut niveau.

Algorithme 5.3 Clustering des Services Web

Entrée : WS ensemble de Services Web WSML du référentiel WSR
Sortie : Ensemble WSDi (WSDi un domaine, $i \geq 1$, i nombre de domaines)
Début
1 : Pour tout WSi de WS faire (i allant de 1 à k , K nombre de Services Web du WSR)
2 : Extraire Di
3 : Si WSDi existe alors
4 : ajouter WSi à WSDi
5 : Sinon créer WSDi
6 : ajouter WSi à WSDi
7 : FinSi
8 : FinPour

Fin

Complexité. L'algorithme de classification des Services Web (Cf. Algorithme 5.3) s'éxécute tant qu'il y a de Services Web dans le référentiel WSR. Ainsi la complexité de cet algorithme est de l'ordre *O(m)*(où m est le nombre de Services Web existants dans le WSR).

L'objectif de cette phase est de réduire le coût de calcul d'un grand ensemble de services et d'adapter les services au niveau de la notion sémantique pour réduire le nombre de Services recommandés menant à diminuer les efforts de recherche de services ce qui permettra donc une découverte plus rapide.

b) Découverte sémantique des Services Web

Cette étape est exécutée après la vérification du traitement hors-ligne (OFF-LINE STEP). Elle comporte cinq principales phases.

- **Phase 1 "Création du Goal"** : Durant cette phase, le Goal et son ontologie correspondante sont créés et spécifiés en langage WSML à partir des informations introduites par l'utilisateur.
- **Phase2 "Recherche des Goals similaires"** : Durant cette phase, les goals introduits sont étudiés et comparés avec des goals similaires enregistrés dans le GR (GR : Goal Repository) référentiel des Goals. L'objectif de cette phase est de fournir un résultat rapide à l'utilisateur. Il est à noter que que cette phase est ignorée si au moins un traitement hors ligne est effectué.
- **Phase3 "Matching du domaine du goal avec le domaine des Services Web"** : Considérant un domaine du Goal, durant cette phase, nous ne gardons que les Services Web correspondant au même domaine. Cette phase consiste à éliminer tous les Services Web non correspondants au domaine du goal.

– **Phase4 "Matching du Goal avec les Services Web du même domaine"** : Durant cette phase, nous effectuons un matching pair à pair de chaque Service Web retourné à la phase 3 et le Goal formulé. Nous ne gardons que les Services Web dont le degré de matching est le meilleur par rapport un seuil fixé à l'avance et par le biais d'un algorithme de calcul de similarité sémantique.

Dans ce qui suit, nous allons présenter ces phases avec plus de détail.

Phase 1 : Création des Goals

Une des étapes cruciales dans une recherche efficace de Services Web est de comprendre les attentes des utilisateurs à travers leurs demandes . Dans ce cas, nous utilisons une page Web HTML intégrée dans un portail qui fournit une interface utilisateur conviviale et permet à l'utilisateur de spécifier ses préférences en tant que valeurs d'entrée.

Les formulaires HTML représentent l'interface la plus populaire pour communiquer avec les utilisateurs à travers des données introduites en entrée et publiées sur le Web.

La phase de création du Goal utilise un formulaire HTML comme entrée. Ainsi, le processus de création du Goal passe par quatre étapes fondamentales (Figure 5.9) : (1) Extraction des informations utiles qui exprime la sémantique des formulaires HTML , (2) Analyse des informations extraites , (3) Traduction du modèle formel en éléments "Goal" et "Ontology" exprimés en langage WSML, et enfin (4)Validation des éléments produits.

– Extraction des informations utiles : Cette étape permet l'acquisition des informations nécessaires pour générer l'élément "Goal" et son ontologie correspondante (concepts, attributs, relations et axiomes) à partir des formulaires HTML. Le formulaire est conçu avec le langage HTML et il est intégré dans une balise FORM.

  Chaque formulaire HTML contient un ensemble de champs de sorte qu'à chaque fois, l'utilisateur peut sélectionner ses préférences et soumettre sa requête. On peut trouver aussi d'autres paramètres comme par exemple, les tableaux, menus déroulants, des boutons radio, cases à cocher,etc.

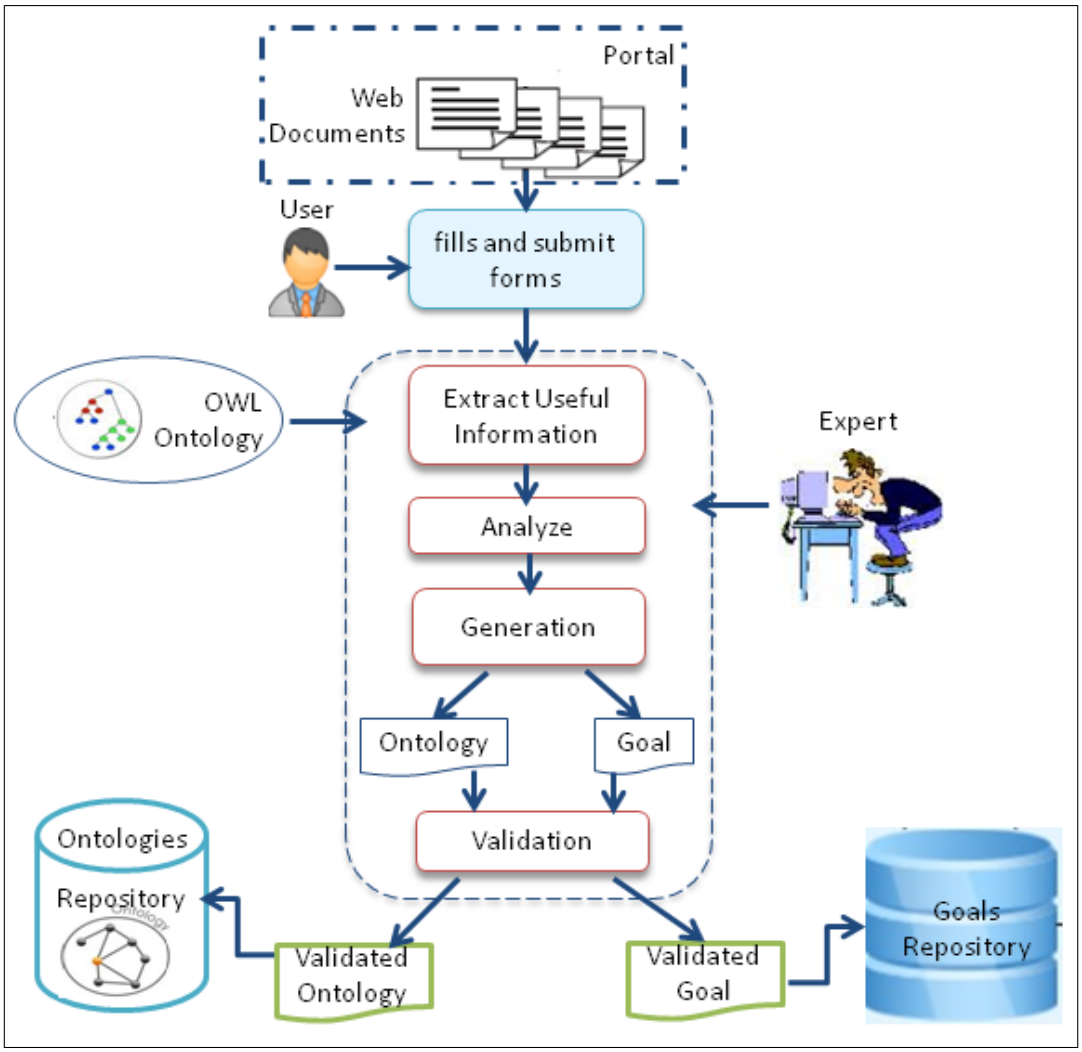


Fig 5.9 – Les étapes de création du Goal

Après avoir rempli toutes les informations requises pour le formulaire, l'utilisateur soumet sa requête au Serveur Web pour un traitement ultérieur. Les informations utiles sont capturées au moment de l'exécution de l'application.

L'information utile extraite est stockée ensuite dans un fichier XML qui sera utilisé pour les étapes suivantes. Notons que, dans les pages HTML, les éléments candidats peuvent être cachés également dans des tableaux, des listes, etc.

– Analyse : Cette étape se concentre sur le matching des informations extraites avec une ontologie de domaine existante. L'étape d'analyse consiste à identifier et récupérer les concepts de l'ontologie qui correspondent aux éléments candidats en utilisant un algorithme de mesure de calcul de similarité sémantique.

– Génération : Cette étape concerne la production des éléments "Goal" et "Ontologie". La méthode proposée pour construire une ontologie WSML à partir du fichier de données XML comprend deux parties. Tout d'abord, elle est basée sur un schéma XML pour

construire l'ontologie. Si le schéma n'existe pas, il peut être automatiquement généré à partir du fichier XML source. Nous allons utiliser les notations suivantes pour spécifier les mappings XML-to-WSML.

Ces trois types de correspondances sont distingués :

Concept : Un noeud XML correspond à un concept WSML ;

Type de données de la propriété : Correspond au type de données de la propriété du concept WSML ;

Propriété/Attribut : Concerne deux concepts WSML.

– Validation : Les étapes précédentes peuvent produire des concepts et/ou des relations erronés , donc une phase de validation du résultat est indispensable. Avant d'inscrire le Goal dans le référentiel des Goals (GR), il doit être validé afin d'assurer sa conformité au langage WSML. Rappelons que le GR est un référentiel qui contient tous les goals formulés précédemment par d'autres utilisateurs. Si le goal n'est pas conforme à la syntaxe du langage WSML, un expert du domaine pourrait apporter quelques modifications pour le corriger. Une fois le goal et l'ontologie sont validés, ils sont stockés respectivement dans les référentiels GR et OR.

  Notez que cette phase produit des descriptions incomplètes des éléments "Goal" et "Ontologie", nous nous sommes concentrés uniquement sur la capabilité qui est la partie indispensable de notre proposition. D'autres informations complémentaires sont hors de la portée du projet.

Phase 2 : Recherche des goals similaires

Au cours de cette phase, nous comparons le Goal ainsi créé durant la phase précédente avec tous les goals enregistrés dans le référentiel GR. Nous supposons que les préférences peuvent être les mêmes pour d'autres utilisateurs du système. L'utilisation du référentiel GR permet de réduire le temps d'exécution afin de faciliter la découverte. Un appariement pair à pair est effectué entre ce Goal et les goals déjà existants, le processus de matching dépend uniquement des paramètres fonctionnels, d'autres informations telles que les paramètres non-fonctionnels ne sont pas nécessaires.

L'existence d'un Goal similaire signifie qu'une même requête a été formulée par un utilisateur préalable, par conséquent, le résultat se trouve dans la base DBH. Cependant, si un traitement hors-ligne a été effectué, cette étape sera ignorée car le traitement hors-ligne supprime l'historique d'exécution (remise à zéro de la base DBH et du GR).

Si un Goal similaire existe, le système retourne directement le Service Web adéquat à partir

de la DBH et c'est la fin du processus de découverte. Dans le cas contraire, nous continuons le processus.

Phase 3 : Correspondance du domaine du Goal avec le domaine des Services Web

L'élément "Goal" doit être exprimé avec le langage WSML et il suit la même structure que celle du composant Service Web WSMO. Le référentiel WSR contient plusieurs Services Web relatifs à des différents domaines.

Durant cette phase, une correspondance du domaine du Goal avec les domaines des Services Web existants est nécessaire car elle permet de ne retenir que les Services Web compatibles au domaine du Goal.

La correspondance est réalisée de la façon suivante : d'abord une extraction du domaine du goal ainsi que les domaines des Services Web est établie, ensuite, un matching pair à pair est effectué avec l'algorithme de [Wu and Palmer, 1994] pour retenir uniquement les Services Web appartenant au même domaine du Goal, c'est à dire les matchings ayant des scores spécifiques par rapport au seuil défini.

Phase 4 : Matching du Goal et les Services Web

Deux types d'éléments WSMO sont considérés, le Goal (G), qui est l'élément créé à partir des besoins des utilisateurs et les Services Web (WS) qui sont les éléments publiés dans le WSR. Il est à noter que durant cette phase, le Goal et les Services Web ont le même domaine d'intérêt.

Nous présentons dans cette section un mécanisme de matching formel des Services Web et les Goals qui est basé sur les ontologies comme représentation de connaissance d'un domaine d'intérêt, à la fois, formelle, compréhensible par la machine. Selon une plage de seuils définie arbitrairement, le matching a été effectué avec huit (8) algorithmes de calcul de similarité sémantique à base du WordNet [Meng *et al.,* 2013]. Une étude comparative des résultats obtenus avec ces (8) huit algorithmes nous a permis de connaitre quel est l'algorithme le plus approprié à notre approche.

Chaque Service Web (respectivement "Goal") a une capabilité décrite dans la partie "Capability" (Fonctionnalité). En outre, la fonctionnalité d'un Service Web (respectivement du "Goal") est associée à des pré-conditions (inputs), des hypothèses (assumptions), des post-conditions (outputs) et des effets.

Les pré-conditions (inputs), les hypothèses (assumptions), les post-conditions (outputs) et les effets doivent se référer à des objets dans le monde où toutes les parties de la transaction ont besoin d'avoir une connaissance et une compréhension commune.

A la fin de cette phase du matching, nous retenons uniquement les Services Web dont le score de matching est le plus grand élevé par rapport au seuil prédéfini.

Nous définissons un Service Web (WS) par $S = <Prs, Pos, Ass, Efs, Dss>$ et $G = <Prg, Pog, Asg, Egf, Dsg>$, où Prs (respectivement Prg) représente les pré-conditions, un ensemble de paramètres (inputs) du WS (respectivement Goal). Pos (respectivement Pog) représente les Post-conditions, un ensemble de paramètres (Outputs) du WS (respectivement Goal). Ass (respectivement Asg) représente les assumptions , un ensemble d'hypothèses du WS (Goal). Efs (respectivement Efg) est un ensemble d'effets du WS (respectivement Goal) et Ds (respectivement Dg) est une description textuelle du WS (Goal) intégrée en qualité de propriété non-fonctionnelle dans le WS (respectivement le Goal).

Nous effectuons un matching pour chaque type des composants S et G (Cf. Fig 5.10).

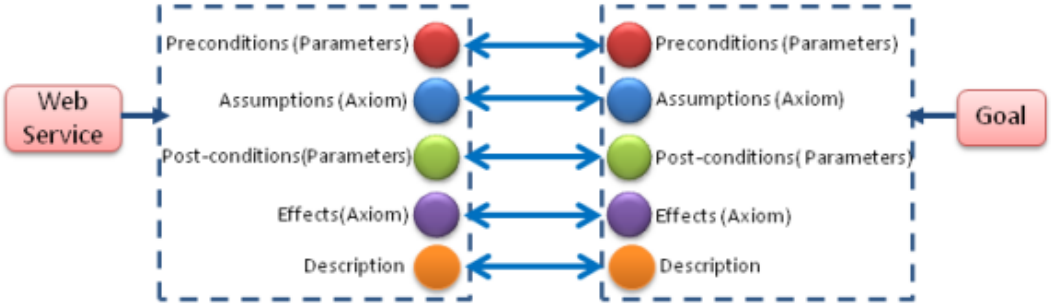


Fig 5.10 – Matching du Goal et du Service Web

Premièrement, pour (Prs, Prg) et (Pos, Pog) nous effectuons un matching pair à pair des paramètres.

Deuxièmement, Comme les hypothèses (Ass, Asg) et les effets (Efs, Efg) correspondent aux contraintes sur les entrées et sorties, définies comme des axiomes, nous effectuons un matching à base de règle logique sur les axiomes existants dans les assumptions et les effets.

Chaque axiome des Ass d'un WS est comparé à un autre axiome des Asg du Goal, dans ce cas, on ne considère que l'égalité des deux axiomes avec un degré de matching égale à 1. Pour cela, nous utilisons des implémentations existantes et relatives à l'ontologie WSMO telle que le moteur de raisonnement IRIS [5] qui est facile à employer dans notre application.

Troisièmement, pour la description de S(Dss) et G(Dsg), telle qu'elle est présentée sous forme textuelle, nous avons extrait des mots-clés de chaque description et nous avons comparé ces mots-clés en utilisant les algorithmes de calcul de similarité sémantique à base du WordNet.

5. IRIS (Integrated Rule Inference System) available at http ://iris-reasoner.org.

Afin de réaliser le processus de l'appariement, nous utilisons les quatres types de matching "Exact", "Plugin", "Subsume" et "Fail" introduits dans [Paolucci *et al.*, 2002], sachant que dans notre processus de découverte des Services Web, l'appariement est effectué sur les mêmes catégories de paramètres, c'est à dire sur les paires d'entrées, ou les paires de sorties.

[Paolucci *et al.*, 2002] a défini les règles de base pour calculer le degré de matching (Cf. Fig 5.11). Les définitions de matching illustrent quatre possibles variantes de résultats qui pourraient survenir lors du matching des capabilités des Services Web.

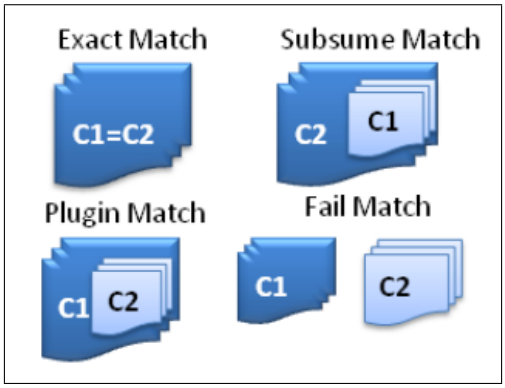


Fig 5.11 – Définition des types de matching selon [Paolucci *et al.*, 2002]

- Exacte : Deux paramètres sont dits similaires si leurs concepts C1 et C2 sont équivalents : $C1 \equiv C2$.
- Plug-in : Deux paramètres sont dits similaires si le concept C1 du premier paramètre est subsumé par le concept C2 du deuxième paramètre (C1 est plus spécifique que C2) : $C1 \sqsubseteq C2$.
- Subsume : Deux paramètres sont dits similaires si le concept C1 du premier paramètre est plus général que le concept C2 du deuxième paramètre (C1 Subsume C2) : $C2 \sqsupseteq C1$.
- Fail : Signifie qu'il n'y a aucune relation de subsomption entre les concepts..

A des fins de notre proposition, nous adaptons cette approche et nous affectons un poids arbitraire à chaque type de matching (Cf. table 5.2).

Tab 5.2 – Attribution d'un poids à chaque type de matching - Adaptée de [Paolucci *et al.*, 2002]

Type de matching	**Poids**
Exact	3
Plug-in	2
Subsume	1
Fail	0

Ci-dessous, nous présentons uniquement un algorithme pour chaque catégorie de matching.

L'algorithme 5.4 présente le mécanisme de matching des paramètres (Prs, Prg) (seulement les pré-conditions). Nous considérons seulement le cas si un matching existe, où le score n'est pas nul, il est donc crucial pour le succès de l'algorithme. S'il n'existe pas de matching, on attribue la valeur 0, nous concluons donc que les paramètres du Goal et des Services Web ne sont pas équivalents. Le matching des post-conditions du Goal et des WS (Pos, Pog) est réalisé de la même manière que pour les pré-conditions.

Algorithme 5.4 Matching des pré-conditions(Prs,Prg)

Entrée : Pré-conditions des Services Web (Prsi,i allant de 1 à n / n :nombre de paramètres)
Pré-conditions des "Goals" (Prgj,j allant de 1 à m / m :nombre de paramètres)
Algorithme de measure de similatité
Sortie : MS_Pr : Score de matching
Début
1 : Pour tout Prsi(i=1 à n) faire
2 : Pour tout Prgj(j=1 à m) faire
3 : Calculer la similarité entre Prsi et Prgj
4 : Si Prsi=Prgj alors $Matching\_Score \leftarrow 3$
5 : Sinon Si Prsi>Prgj alors $Matching\_Score \leftarrow 2$
6 : Sinon Si Prsi<Prgj alors $Matching\_Score \leftarrow 1$
7 : Sinon $Matching\_Score \leftarrow 0$
8 : FinSi
9 : FinPour
10 : $MS\_Pr \leftarrow MS\_Pr + Matching\_Score$;
11 : FinPour
12 : FinPour
13 :Retourner MS_Pr

Fin

Complexité. L'algorithme de matching des pré-conditions (Cf. Algorithme 5.4) s'éxécute tant qu'il y a de pré-conditions du Service Web et celles du "Goal". Ainsi la complexité de cet algorithme est de l'ordre *O(m*n)*(où m (respectivement n) est le nombre de pré-conditions relatives au Services Web (respectivement au "Goal").

L'Algorithme 5.5 présente en le matching des axiomes (seulement les assumptions : hypothèses). Le matching des effets du Goal et des WS (Efs, Efg) est réalisé de la même manière que pour les assumptions (hypothèses).

Algorithme 5.5 Matching des Assumptions(Ass,Asg)

Entrée : Ass : un ensemble i d'axiomes (i allant de 1 à n / n : nombre d'axiomes)
Algorithme de mesure de similarité
Sortie : MS_Ass : Matching_Score
Début
1 : Pour tout Assi (i allant de 1 à n) faire
2 : Pour tout Asgj (j allant de 1 à m) faire
3 : Comparer chaque axiome
4 : Si Assi=Asgj alors Matching_Score=1
5 : FinSi
6 : $MS\_Ass \leftarrow MS\_Ass + Matching\_Score$
7 : FinPour
8 : FinPour
9 : Retourner MS_Ass

Fin

Complexité. L'algorithme de matching des assumptions (Cf. Algorithme 5.5) s'éxécute tant qu'il y a des assumptions du Service Web et celles du "Goal". Ainsi la complexité de cet algorithme est de l'ordre *O(m*n)*(où m (respectivement n) est le nombre d'assumptions relatives au Services Web (respectivement au "Goal").

En ce qui concerne la description, d'abord nous allons extraire tous les mots-clés de la valeur textuelle du Goal et des Services Web. Ensuite, nous effectuons un matching de ces mots-clés en utilisant un algorithme de mesure de similarité sémantique. La valeur de la description ainsi, est obtenue à partir d'une propriété non-fonctionnelle.

Enfin, la similitude globale entre le Goal (G) et les Service Web (WS) peut être obtenue en effectant la somme des différents matching relatifs aux pré-conditions, post-conditions, assumptions, effets et description comme présenté par la formule (1) :

$$MS := MS\_Pr + MS\_Ass + MS\_Po + MS\_eff + MD \qquad (1)$$

Telles que :

- MS : Repésente le score final de matching.
- MS_Pr : Repésente le score du matching des pré-conditions.
- MS_Ass : Repésente le score du matching des assumptions.
- MS_Po : Repésente le score du matching des post-conditions.
- MD : Repésente le score du matching des descriptions.

Ainsi, ces mesures sont calculées et normalisées. La normalisation consiste involves convertir la valur de la mesure pour obtenir une nouvelle valeur comprise entre 0 et 1. La valeur 1 indique une équivalence sémantique totale entre les entités.

Enfin, le matching entre le Goal et un ensemble de Services Web peut être mesurée quantitativement. Le Service Web qui possède un score de similitude plus élevé est celui qui est susceptible de satisfaire le "Goal". Par conséquent, le résultat du matching peut retourner plusieurs Services Web ayant le même score élevé.

c) Sélection du meilleur Service Web

A la fin du processus de découverte, le système pourrait renvoyer plusieurs Services Web appropriées avec le même degré élevé de matching. Il est indispensable donc de choisir un seul Service Web pour satisfaire les besoins des utilisateurs.

La question qui se pose : "Comment peut-on choisir ce meilleur Service ?". L'idée donc est de retenir uniquement le meilleur Service Web parmi la liste des Services Web retournés durant la phase 4. Dans ce cas, nous avons utilisé les propriétés non-fonctionnelles des Services Web, telles que la précision, la performance et l'évolutivité. Nous effectuons une classification des Services Web en fonction de ces trois critères (précision, performance et évolutivité) avec le même poids. Chaque classement (tri) est stocké dans un tableau contenant les Services Web.

En outre, pour chaque Service Web, nous calculons la somme de ses rangs dans chaque tableau. Nous considérons le Service Web dont la somme des rangs prend la valeur la plus faible.

Le principe de sélection est comme suit : nous supposons un exemple de trois Services Web retournés de l'étape de découverte avec le même score élevé : WS1 , WS2 et WS3.

Pour chacun des critères choisis arbitrairement, nous effectuons une classification des trois Services Web, un tableau est dressé pour chaque critère, ce tableau comporte les trois services avec leur classement respectifs. Nous présentons le rang de chaque Service Web dans le tableau selon les trois critères choisis :

Ainsi, les rangs respectifs de chaque Service Web dans le tableau du critère est comme suit (Cf. Fig 5.12) :

« Accuracy »	«Performance »	«Scalability »
WS1	WS3	WS2
WS3	WS1	WS1
WS2	WS2	WS3

WS	Table1	Table2	Table3	Total
WS1	1	2	2	5
WS2	3	3	1	7
WS3	2	1	3	6

Fig 5.12 – Procédure de sélection du meilleur Service Web.

- Pour la propriété précision : (WS1 , 1) , (WS2 , 3) , (WS3 , 2).
- Pour la propriété Performance : (WS1 , 2) , (WS2 , 3) , (WS3 , 1).
- Pour la propriété évolutivité : (WS1 , 2) , (WS2 , 1) , (WS3 , 3).

Par la suite, nous calculons les sommes des rangs de classement des Services Web dans chaque tableau, ce qui en résulte : (WS1 , 5), (WS2 , 7) , et (WS3 , 6) et nous sélectionnons donc le Service Web (WS1, 5) qui a la plus petite valeur dans le tableau consolidé, ce qui signifie que ce Service figure dans les premiers rangs des autres tableaux.

Enfin, les critères de sélection que nous avons choisises sont arbitraires ; nous supposons que la performance, l'évolutivité et la précision sont des caractéristiques intéressantes d'un Service Web. Cependant, le mécanisme de sélection présenté ci-dessus peut être étendu pour supporter plusieurs propriétés non-fonctionnelles.

5.5. Conclusion

Dans ce chapitre, Nous avons fourni une description détaillée de nos contributions relatives à la ré-ingénierie des Services Web vers les Services Web Sémantiques ainsi que la découverte des Services Web à base des préférences des utilisateurs. Les deux contributions sont basées sur le concept de matching ontologique.

L'approche de ré-ingénierie proposée consiste à analyser en amont un document WSDL pour extraire les informations pertinentes. Le système développé reçoit en entrée un document WSDL et plusieurs ontologies de domaine et génère en sortie deux fichiers d'extension WSML : une "Ontologie" et un "Service Web" selon le modèle concetuel de l'ontologie de service WSMO-Lite. L'approche montre l'efficacité du processus de ré-ingénierie des applications Services Web, en offrant un compromis coût/temps et permet l'évolution du système existant.

Le processus de ré-ingénierie passe principalement par deux (2) grandes étapes, une étape de rétro-ingénierie, et une étape d'ingénierie, ces deux étapes sont réparties en (5) phases, depuis le chargement du fichier WSDL jusqu'à la validation des éléments produits. La première phase permet la classification du Service afin d'identifier l'ontologie de domaine adéquate pour l'annotation, la deuxième, une phase d'annotation du fichier WSDL selon les mécanismes utilisés dans SAWSDL, en vue d'enrichir sémantiquement le fichier WSDL, la troisième phase consiste en une extraction des informations pertinentes à partir du fichier WSDL annoté, la quatrième phase permet la génération des deux éléments : "Ontologie" et "Services Web" et enfin la phase de validation permet de vérifier la compatibilité et la conformité des deux éléments produits au langage WSML.

La deuxième contribution de cette thèse est la proposition d'une approche de découverte des Services Web baée sur les préférences des utilisateurs. Cette approche consiste à formuler un but("Goal") à partir d'informations introduites dans une page HTML et trouver les Services Web susceptibles de satisafire ce Goal. Le système mis en oeuvre reçoit en entrée des données et retourne en sortie un ou plusieurs Services Web correspondants aux besoins des utilisateurs.

L'approche de découverte comprend trois principales étapes : (1) Une catégorisation des services basée sur la sémantique ; (2) Une Découverte sémantique et (3) Une sélection des meilleurs Services. La première étape concerne l'évolution et la maintenance du système et la mise à jour de l'ensemble des différents référentiels. Les deux dernières étapes concernent le processus de découverte dans son ensemble : découverte des Services Web pertinents et la sélection du meilleur Service susceptible de répondre aux besoins des utilisateurs.

L'approche de découverte utilise la notion de traçabilité, elle consiste à garder l'historique de l'exécution dans une base de données qui sera mise à jour régulièrement afin de rendre la découverte plus rapide.

Le chapitre suivant présente l'implémentation des deux approches illustrée par des études de cas des scénarios du monde réel. L'implémentation comprend un outil nommé WSDL2WSMO-Lite pour le processus de ré-ingénierie et un cadre de travail complet relatif à la tâche de découverte.

Chapitre

6 Expérimentation

Sommaire

6.1. Introduction

Un Service Web est un logiciel accessible via internet. Les technologies actuelles des Service Web permettent l'échange de messages entre Services Web avec le (SOAP), décrivant l'interface technique avec le fichier (WSDL), et la publication des Services Web dans un registre (UDDI). Ces technologies ne fournissent aucune information sur la signification de l'information utilisée ou la description explicite de la fonctionnalité du service suffisament nécessaire pour l'utilisation automatisée et l'interopérabilité des Services Web.

L'amélioration des technologies des Services Web visent à des techniques plus sophistiquées pour décrire les Services Web, mettant l'accent sur le concept des Services Web Sémantiques. Dans notre compréhension, un Service Web sémantique est définie comme une "ressource logicielle sémantiquement balisée qui peut être publiée, découverte, composée avec d'autres ressources et exécutée à travers le Web dans une tâche conduite de manière automatique". Enfin, les machines traitent des descriptions pertinentes des informations fournies des Services Web et les tâches telles que la découverte peuvent être exécutées. La découverte automatique

consiste à trouver des Services Web qui peuvent résoudre une requête définie par un demandeur de service.

Dans ce chapitre nous décrivons les prototypes implémentant la ré-ingénierie et la découverte des Services Web, ces prototypes sont indépendants. La réalisation de nos contributions a été faite sur trois grandes étapes. Une première étape a été nécessaire pour la ré-ingénierie des Services Web WSML. Une deuxième étape a consisté à la découverte et la sélection des services susceptibles de répondre aux besoins des utilisateurs. Quant à la troisième étape évalue les performances des algorithmes de matching utilisés pendant le processus de découverte afin de décider quel est l'algorithme le plus approprié à l'approche de découverte proposée pour valider son intégration au sein du prototype mis en oeuvre.

Ce chapitre ne reprend pas les détails techniques des développements du code source et de déploiement. Il est dédié plutôt à la présentation des démarches de réalisation des mécanismes et des processus pour mesurer les performances des systèmes de ré-ingénierie et de découverte basés sur nos approches et d'en effectuer la mise au point et l'évaluation. Ceci nous a servi pour valider les approches ainsi que s'assurer que les performances soient meilleures que celles d'autres approches.

6.2. Objectifs principaux des prototypes

L'objectif des prototypes que nous avons développés est double. Tout d'abord, il s'agit de valider au travers d'une implémentation certains éléments du cadre méthodologique que nous avons proposé. Ensuite, il s'agit d'effectuer un plan d'expérimentation prouvant la pertinence de nos approches sur des cas d'étude réels.

Nous avons développé un outil semi-autmomatique pour la ré-ingénierie. Cet outil comporte pluieurs modules. L'outil de ré-ingénierie montre comment les informations extraites à partir d'un fichier WSDL seront utilisées pour la conception des éléments "Ontologie" et "Service Web" de l'ontologie de service WSMO-Lite. Cette démarche procède, tout d'abord, par une étape de rétro-ingénierie pour l'identification et l'analyse des informations pertinentes fournies par le WSDL et par ingénierie pour générer les sorties correspondantes du WSMO-Lite.

Pour la découverte, nous avons développé un cadre de travail complet comportant plusieurs modules et outils qui travaillent de manière complémentaire. Ce système consiste en plusieurs référentiels pour le stockage des éléments utiles, aussi il est constitué d'une base de données pour la sauvegarde de la trace d'exécution qui joue le rôle de bases des faits.

Dans ce qui suit, nous allons présenter séparément chaque prototype avec des exemples explicatifs.

6.3. Le système WSDL2WSMO-Lite

Dans cette section, nous présentons le système de ré-ingénierie relatif à l'approche proposée.

6.3.1 Architecture du WSDL2WSMO-Lite

Cette section présente les différents éléments qui caractérisent le développement de l'outil WSDL2WSMO-Lite. Après avoir rappelé les fonctionnalités principales de l'outil, nous présentons l'architecture technique générale du prototype (Cf. fig 6.1), ensuite nous détaillons la fonctionnalité des principaux modules développés.

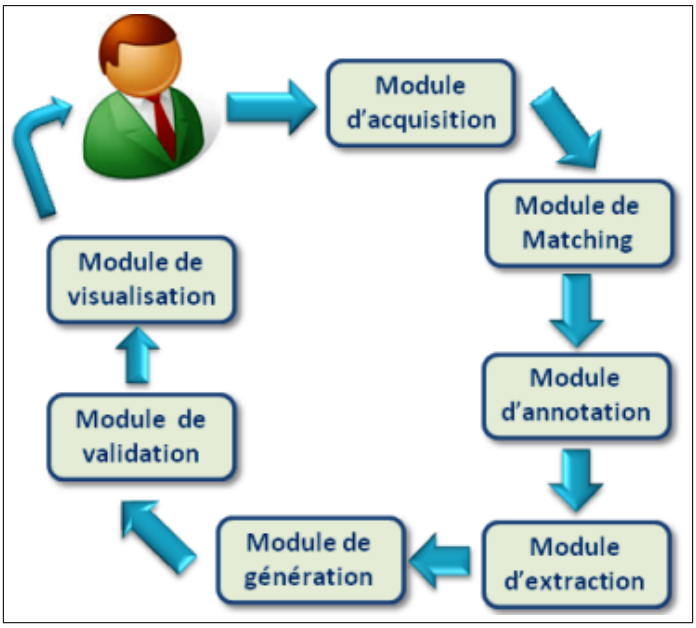


Fig 6.1 – Architecture générale de l'outil WSDL2WSMO-Lite.

Le prototype est développé en utilisant le couple (langage Java, plateforme NetBeans). Ce choix est justifié par la puissance du langage Java et la consistance de ses différentes API surtout en ce qui concerne les documents XML. Quant à NetBeans, nous l'avions utilisé en raison de sa simplicité et son assistance au développeur et du fait aussi qu'il est "opensource".

L'architecture générale du prototype WSDL2WSMO-Lite comprend plusieurs modules (Cf. fig 6.1) qui travaillent d'une façon complémentaire en vue de fournir les résultats attendus.

- **Module d'acquisition :** Ce module permet le chargement du fichier WSDL choisi et les ontologies de domaine. Des ressources sont configurées par défaut mais l'utilisateur peut toujours sélectionner lui même son référentiel.

- **Module de matching :** Ce module permet de visualiser dans un tableau les différentes ontologies avec leurs mesures de similarité correspondantes (Les scores). Rappellons, que nous avons utilisé l'algoritme de calcul de similarité de Wup et Palmer [Wu and Palmer, 1994]. L'utilisateur peut décider donc sur le seuil à fixer pour le matching et par conséquent sur l'ontologie à choisir, notamment celle dont le degré de similarité est le meilleur.
- **Module d'annotation :** Une fois l'ontologie de domaine est sélectionnée, ce module effectue une annotation du fichier WSDL selon le mécanisme ModelReference utilisé dans l'apparoche SAWSDL. Comme résultat, nous obtenons un fichier WSDL annoté, un fichier enrichi sémantiquement qui peut servir pour les étapes suivantes.
- **Module d'extraction :** Ce module permet l'extraction des informations utiles à partir du fichier WSDL annoté. Durant cette étape, nous effectuons une extraction des informations utiles à notre système. Les informations à prendre en considération sont celles du schéma XML du fichier WSDL ainsi que les informations relatives aux opérations du Service Web.
- **Module de génération :** Ce module, par le biais de règles de mapping, utilise les informations fournies par le module d'extraction pour les transformer en des spécifications WSMO-Lite. Vu la consistance et la complexité du langage WSML, nous nous sommes contentés à quelques passages au langage WSML selon des fichiers modèles établis et remplis à la volée.
- **Module de validation :** Ce module permet la validation syntaxique des éléments produits selon les spécifications du langage WSML.
- **Module de visualisation :** Ce module permet la visualisation des éléments "Services Web" et "Ontologie", il représente l'étape finale du fonctionnement de notre prototype.

6.3.2 Usage du WSDL2WSMO-Lite

Lors de l'exécution du WSDL2WSMO-Lite, le module d'acquisition permet le chargement d'un fichier WSDL, l'utilisateur pourra choisir un fichier WSDL.

L'étape suivante consiste en un "matching" de toutes les ontologies du dossier sélectionné avec l'ontologie du WSDL (élaborée à partir du schéma XML), les résultats sont visualisés dans un tableau pour permettre à l'utilisateur de choisir l'ontologie de domaine dont le degré de matching est le meilleur.

L'étape suivante présente l'annotation du fichier WSDL avec l'ontologie de domaine sélectionnée. Rappelons que le mécanisme d'annotation est similaire à celui du "ModelReference" de l'approche SAWSDL. Le résultat de cette étape est un fichier WSDL annoté que l'utilisateur peut stocker dans un dossier par défaut.

WSDL2WSMO-Lite utilise ensuite le module d'extraction pour extraire les informations pertinentes et utiles pour les étapes suivantes. Le module de génération transforme ces informations avec des règles de "mapping" en des éléments "ontologie" et "Service Web" spécifiés dans le langage WSML.

Le module de validation prend la main pour vérifier la syntaxe des deux éléments. L'utilisateur peut au besoin porter certaines modifications sur les fichiers produits. Enfin, les résultats sont affichés à l'utilisateur par le module de visualisation.

6.4. Le système de découverte des Services Web

Le processus de découverte dans le domaine "e-health" constitue un nouveau défi en raison de différentes contraintes. La découverte à base de descriptions sémantiques semble être l'approche la plus prometteuse aujourd'hui car, elle facilite le processus de matchmaking.

Par conséquent, le principal avantage de l'application du processus de découverte fondé sur la sémantique est qu'il permet de développer et de maintenir les Services Web du domaine e-health à un moindre coût en fournissant un résultat plus rapide .

En particulier, la technologie des Services Web Sémantiques peut optimiser plusieurs processus dans le domaine des Analyses médicales. Ces processus sont principalement liés à des interactions avec l'homme et, par conséquent, avec les coûts qui leur sont associés. Le principal avantage de l'application de la technologie des Services Web Sémantiques est qu'elle permet de développer et de maintenir des services, notamment ceux relatifs au domaine du e-health avec des coûts minimes et un temps bien réduit.

Dans cette section, nous allons illustrer les avantages de notre contribution relative à la découverte à travers deux scénarios réels. Dans de tels scénarios, un patient veut effectuer un bilan médical, nous considérons deux différents cas (Cf. fig 6.2).

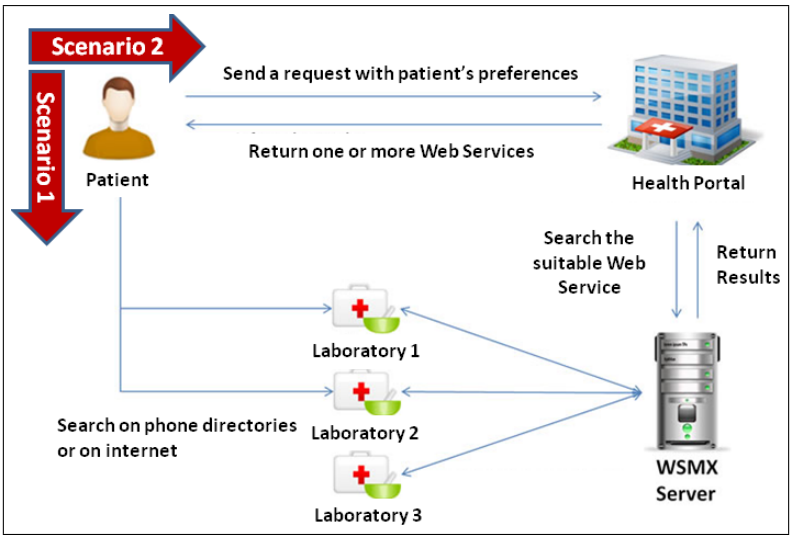


Fig 6.2 – Les deux scénarions de découverte des laboratoires d'analyse

Dans le scénraio 1 (Cf. fig 6.2 : scénario 1), pour effectuer des analyses médicales, un patient doit se déplacer ou parcourir tous les laboratoires afin de trouver un laboratoire susceptible de répondre à ses besoins. Cependant, la tâche de recherche sera plus complexe sans compter les coûts y afférents, le temps de réponse, aussi bien que la qualité de service retourné.

Toutefois, pour surmonter ces limitations, nous avons mis en place un laboratoire d'analyses médicales virtuel (VMAL : Virtual Medical Analysis Laboratory) (Cf. fig 6.2 : scénario 2) qui est une plateforme d'exécution des Services Web Sémantiques permettant de découvrir les laboratoires d'analyse appropriés. Notre but est d'automatiser totalement ou partiellement ces processus afin de répondre aux exigences de l'utilisateur et d'assurer un certain degré de confiance.

Sur la base des algorithmes proposés, nous avons mis en place un ensemble d'outils pour effectuer chaque tâche de l'approche en utilisant le langage de programmation Java et l'IDE NetBeans qui permettra à notre contribution de fonctionner sur diverses platesformes telles que Windows, Unix, Mac, etc, aussi il existe un grand nombre d'API java dsiponibles sur internet qui facilitent l'analyse des pages web, la construction du "goal" et des ontologies ainsi que le matching ontologique.

Dans le scénario 2 (Cf. fig 6.2 : scénario 2), le patient va interagir avec l'application à travers une interface graphique conviviale contenue dans une page Web intégrée dans un portail. Le patient soumet une requête au portail avec ses préférences en introduisant les informations utiles dans la page web. Un But("goal") est donc créé à partir de ces préférences, ensuite, un processus de recherche des Services Web adéquats est effectué dans le référentiel WSR. A la fin, le patient reçoit une réponse concernant un ou plusieurs Services Web (laboratoires)

correspondants à sa demande (s'ils sont disponibles).

Ainsi, le déroulement du processus du bilan médical est présenté comme suit :

- L'utilisateur informe le système sur le bilan qu'il doit effectuer,
- L'agent système découvre les cliniques pouvant effectuer ce type de bilan,
- L'agent système retourne à l'utilisateur le(s) Service(s) Web conséquents.

6.4.1 Architecture du système de dévouverte

Le système mis en oeuvre est constitué d'un ensemble de modules qui procèdent automatiquement ou semi-automatiquement comme présenté dans la figure 6.3.

Cette section va présenter les différents modules qui caractérisent le développement du système de découverte. Après avoir rappelé les fonctionnalités principales de l'outil, nous présentons l'architecture technique générale du prototype (Cf. fig 6.3), ensuite nous détaillons la fonctionnalité des principaux modules développés.

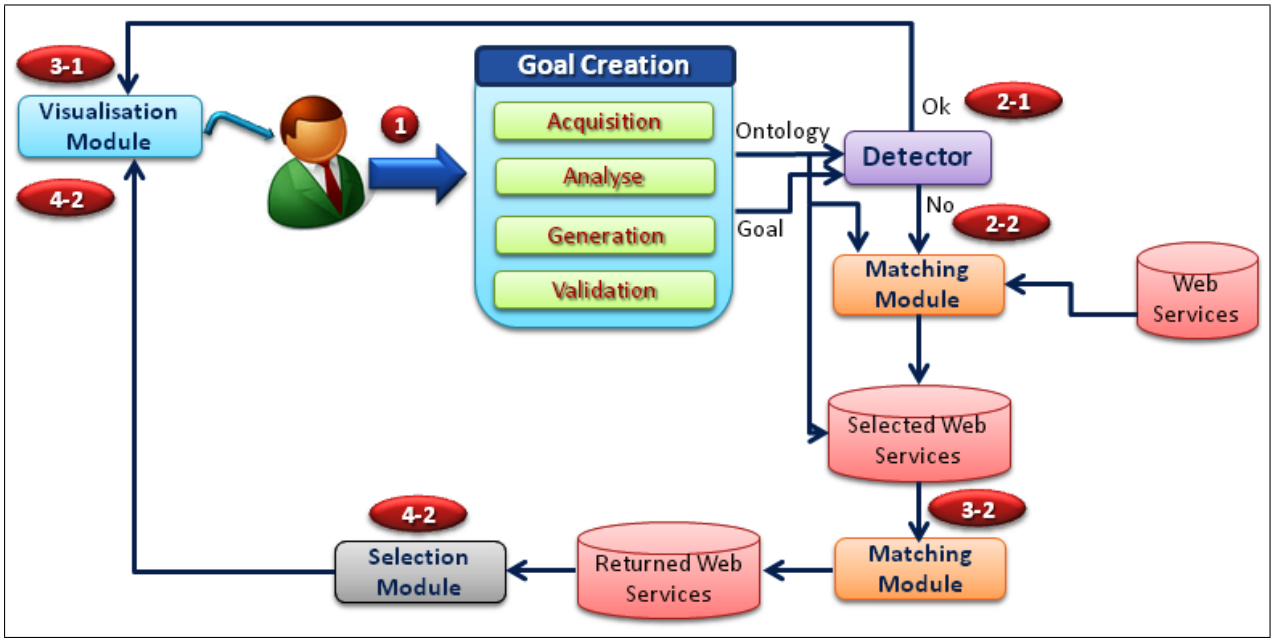


Fig 6.3 – Architecture générale du système de découverte.

L'architecture générale du système de découverte comprend plusieurs modules (Cf. fig 6.3) qui travaillent d'une façon complémentaire en vue de fournir les résultats attendus. La chronologie d'interaction entre ces modules est numérotée sur la figure selon l'odre d'exécution.

- **Module d'acquisition :** Ce module permet la capture des informations introduites par l'utilisateur dans la page web du portail considéré. Durant cette étape, nous effectuons

une extraction des informations utiles à notre système. Les informations à prendre en considération sont celles du formulaire HTML et autres éléments candidats comme les tableaux et les listes.

- **Module d'analyse :** Ce module permet de filtrer et d'analyser les informations capturées.
- **Module de génération :** A travers des règles de mapping, ce module utilise les informations fournies par le module d'extraction pour les transformer en des spécifications WSMO en vue de créer le But("goal") et son ontologie correspondante. Vu la consistance et la complexité du langage WSML, nous nous sommes contentés à quelques passages au langage WSML selon des fichiers modèles établis et remplis à la volée..
- **Module de validation :** Ce module permet la vérification et validation syntaxique des éléments "Goal" et "Ontologie" produits selon les spécifications du langage WSML.
- **Module de visualisation :** Ce module permet la visualisation du résultat à l'utilisateur, à savoir les Services Web disponibles retournés.
- **Module de matching :** Ce module permet effectue le matching du "goal" avec les Services Web du référentiel WSR. Le rôle de ce module est double, en premier lieu, il réalise un matching de domaine, ensuite il effectue un matching des Services Web avec le But du même domaine d'intérêt.
- **Module de sélection :** Sur la base des Services Web retournés, ce module permet de renvoyer le meilleur Service Web en fonction de quelques propriétés non-fonctionnnelles choisises arbitrairement.

6.4.2 Etapes du processus de découverte

Rappelons que le processus de découverte passe par trois grandes étapes :

- Catégorisation basée sur la sémantique qui concerne la maintenance et l'évolution du système ainsi que le classement des Services Web selon leurs domaines respectifs ;
- Découverte sémantique qui représente le processus de découverte dans son intégralité ;
- Sélection des meilleurs Services Web concerne le choix d'un Service si le système renvoi, en résultat, plusieurs Services Web.

Dans ce qui suit, nous allons présenter les étapes du processus de découverte proposé.

a) Catégorisation sémantique

Rappelons que le système de découverte que nous avons proposé, dispose de plusieurs référentiels contenant respectivement tous les fichiers utiles pour le processus de découverte, come le référentiel des services, WSR, le réferentiel des ontologies, OR. Ces référentiels doivent être mis à jour régulièrement.

Cette étape concerne la maintenance et l'évolution du système. Elle est effectuée occasionnellement en hors-ligne, chaque fois qu'une opération d'insertion ou de suppression d'un service est nécessaire, elle permet de mettre à jour tous les référentiels du système. Aussi, aprés chaque traitement considéré (insertion ou suppression), une catégorisation des Services Web du référentiel WSR est effectuée en vue de classer les Services selon leurs domaines d'intérêt respectifs pour faciliter et accélérer la recherche des Services. La figure 6.4 représente le diagramme des cas d'utilisation de l'étape de catégorisation. Cependant, en vue de preserver l'intégrité des informations, les traitements décrits précédemment sont exécutés uniquement par l'adminsitrateur du système, c'est la raison pour laquelle, des mots de passe ont été mis en place en vue de protéger le système. Une procédure d'authentification est indispensable pour exécuter les traitements considérés.

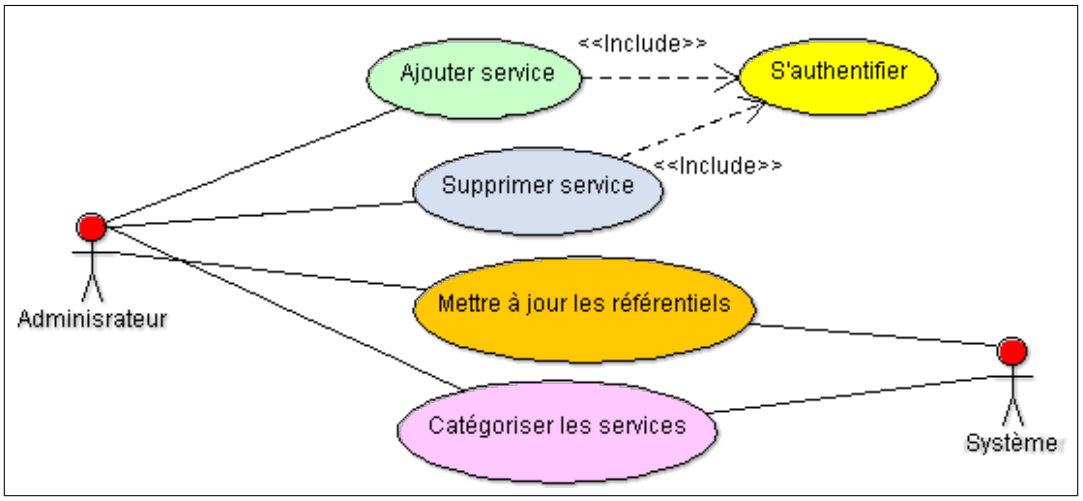


Fig 6.4 – Cas d'utilisation relatif à l'étape de catégorisation sémantique

Le cas d'utilisation "ajouter un service" (Cf. fig 6.4) consiste en l'ajout d'un Service Web et son ontologie correspondante. Les deux fichiers sont exprimés dans le langage WSML. Les deux fichiers sont rajoutés respectivement aux réferentiels WSR et OR.

Le cas d'utilisation "supprimer un service" (Cf. fig 6.4) consiste en la suppression du Service Web et son ontologie correspondante des référentiels WSR et OR. La procédure de suppression implique aussi la suppression du Service de la base de données de l'historique DBH.

Ces traitements, en particulier la suppression des Services Web sont reportés automatiquement sur la base de données de l'historique d'exécution (DBH), c'est la raison pour laquelle, une mise à jour de la DBH est necéssaire pour assurer la cohérence des différents référentiels.

A la fin des processus de mise à jour, un clustering des Services Web est indispensable pour les étapes suivantes, et qui consiste à classer les Services Web selon des catégories de domaines. Ainsi, une recherche à base de domaine contribue à la fiabilité et à l'efficacité du fonctionnement de notre système.

b) Découverte des Services Web

L'étape de la découverte renferme plusieurs phases. La figure 6.5 représente un diagramme d'activité qui résume ces phases.

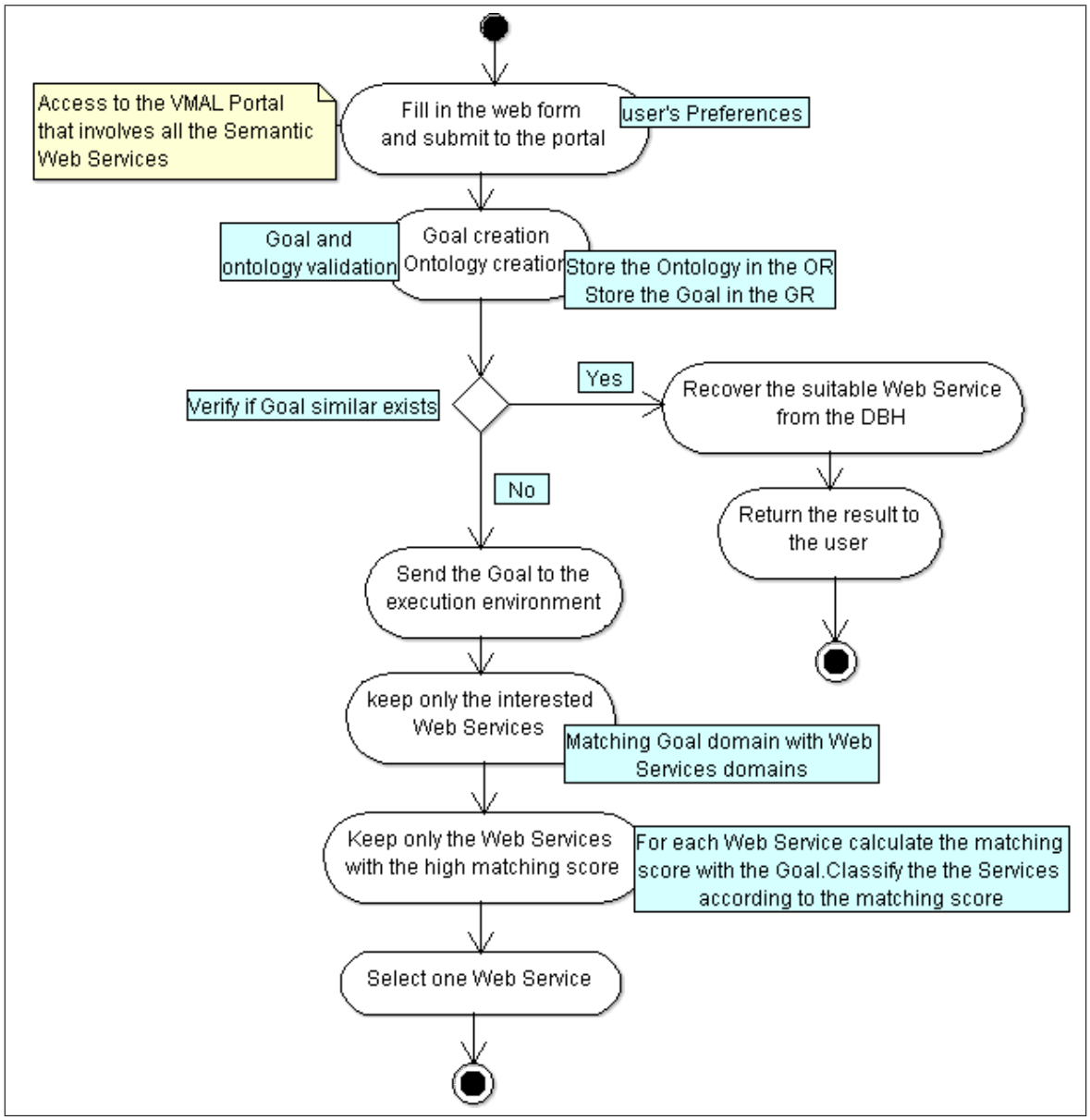


Fig 6.5 – Diagramme représentant les activités du système en vue de réaliser la découverte

Le point d'entrée du système de découverte consiste en une interface conviviale, elle a été créée selon les exigences des besoins relatifs au domaine des analyses médicales. Cependant, pour un autre domaine d'intérêt, cette interface doit être modifiée mais les processus de catégorisation, aussi bien que la découverte et la sélection restent inchangés.

- **Création des Buts** : L'utilisateur remplit le formulaire de la page Web relative au portail et soumet sa requête au portail, il séléctionne les types de bilans à effectuer.

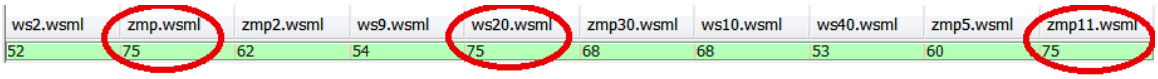
  Le système capture les informations introduites dans les champs du formulaire de la page Web et les enregistre dans un fichier XML. Ensuite, il effectue un filtrage et enrichissement sémantique à l'aide d'une ontologie de domaine, la génération du But et de son ontologie correspondante est réalisée avec le langage WSML et par conséquent la validation de ces deux éléments.

- **Recherche des Buts similaires :** Durant cette phase, le système recherche des buts similaires dans le référentiel des buts, GR. S'il un tel but existe, il récupère directement le service de la base de l'historique de l'exécution et par conséquent le résultat est renvoyé à l'utilisateur. Si un but similaire n'existe pas, nous continuons le processus de découverte. L'existence d'un goal similaire signifie que ce type de bilan a été réalisé pour un patient précédent.

- **Matching du domaine du But avec les domaines des Services Web :** Durant cette phase, le système effectue un premier matching entre le but formulé à partir de la requête des utilisateurs et tous les Services Web du référentiel WSR.

  Cette phase permet d'éliminer tous les Services ne correspondant pas au domaine du But, c'est à dire les services n'apprteneant pas au domaine des analyses médicales, ce qui permet de réduire l'espace de recherche. Il est à noter que les Services Web éliminés ne sont pas supprimés du réferetiel WSR.

- **Matching du But et des Services Web :** A ce stade, le système établit un matching pair à pair des Services Web retenus de la phase précédente et le but déjà formulé pour trouver les Services susceptibles de satisfaire le but (Cf Fig 6.6). Chaque Service Web est reporté avec son degré de matching relatif. L'utilisateur peut décider sur l'algorithme de matching à choisir parmi les huit (8) algorithmes de calcul de similarité sémantique mis en oeuvre (Le lecteur trouvera une présentation détaillée de ces huit algorithmes dans l'annexe B). Il pourra également définir un seuil à partir d'une suite de choix, et plus le seuil est grand, plus, le système renvoi les résultats avec plus de précision.



ws2.wsml	zmp.wsml	zmp2.wsml	ws9.wsml	ws20.wsml	zmp30.wsml	ws10.wsml	ws40.wsml	zmp5.wsml	zmp11.wsml
52	75	62	54	75	68	68	53	60	75

Fig 6.6 – Exemple de score enregistré (en pourcentage)

.

Les Services Web retenu (Cf. fig 6.6)seront donc ceux dont le degré de matching est le meilleur par rapport au seuil prédéfini.

c) Sélection du meilleur Service Web

Après la phase de découverte, plusieurs Services Web peuvent être retenus par le système. Ces Services ont des degrés de matching similaires et qui représentent les meilleurs scores enregistrés durant l'étape de découverte. A ce stade, il est intéressant de sélectionner un Service de la liste et renvoyer la réponse à l'utilisateur. Pour cela, nous avons mis en place une stratégie de sélection basée sur l'utilisation des propriétés non-fonctionnelles.

Les propriétés choisies arbitrairement pendant le processus de sélection constituent, la précision, la performance et l'évolutivité. Cependant, nous pourrons toujours étendre la liste de critères sans d'autant modifier le traitement.

6.5. Validation

A cette étape, nous proposons de valider la satisfaction des exigences par rapport aux applications à base de services construites. Pour cela, nous utilisons deux approches : (1) une approche empirique assurant la validation des éléments WSMO-Lite produits et une approche automatique permettant d'évaluer la performance des algorithmes de matching ontologique utilisé dans le processus de découverte.

En ce qui concerne la première approche, les éléments WSMO-Lite produits ont été intégrés dans les référentiels des services et utilisés pendant la découverte des Services Web basée sur les préférences des utilisateurs.

Aussi, afin d'évaluer la performance des algorithmes de matching ontologique utilisé dans notre approche, il est nécessaire de les confronter avec des différents tests de découverte des Services Web et comparer par la suite les résultats. Ainsi, une étude comparative a été faite pour déterminer quelle mesure de similarité est appropriée à cette notre approche et donne de meilleurs résultats.

Supposons qu'un système ait retourné dix (10) services en réponse à une requête. Supposons aussi que l'on ait vérifié ces services manuellement et nous estimons que, seuls cinq (5) de ces

services sont pertinents. On dira alors que le système a une précision de 50%. Ceci fonctionnerait parfaitement sans un petit problème. Celui-ci est dû au fait que seuls cinq (5) services sont pertinents dans toute la collection. Trouver les cinq services serait alors un résultat parfait, et ne doit pas être vu comme seulement 50% de succès. Par conséquent, on a besoin de définir une autre mesure comme le ratio des services pertinents trouvés sur les services pertinents dans toute la collection. Pour cette mesure, notre exemple aura comme valeur cinq/cinq, donc 100%. Cette mesure est communément appelée Rappel alors que la première mesure, le ratio des services trouvés pertinents sur le total des services trouvés, est appelée Précision.

La précision et le rappel sont des critères trés importants provenant de la recherche d'information et adaptés à la tâche du matching ontologique. Les indicateurs précision et rappel sont basés sur la comparaison des Services Web résultants du processus de découverte.

Les mesures de Précision, Rappel et F-mesure [Do *et al.,* 2002] ont été des métriques largement exploitées pour évaluer la qualité des alignements ontologiques obtenus. Les mesures de précision/rappel sont obtenues en partitionnant l'ensemble des Services, restitués par le système, en deux catégories : les services pertinents et les services non pertinents. Ces deux catégories se définissent comme suit :

- **Taux de rappel** : le rappel mesure la capacité du système à retrouver tous les services pertinents répondant à une requête. Il est donné par le rapport entre les services retrouvés pertinents et l'ensemble des services pertinents de la base.
- **Taux de précision** : la précision mesure la capacité du système à rejeter tous les services non pertinents à une requête. Il est donné par le rapport entre l'ensemble des services sélectionnés pertinents et l'ensemble des services sélectionnés.

Les taux de précision et de rappel sont donnés par les formulations suivantes :

$$Rappel = \frac{R_+}{R} \qquad \dots (1)$$

$$Précision = \frac{R_+}{M} \qquad \dots (2)$$

où :

- R : désigne le nombre de services pertinents dans toute la collection,
- M : représente le nombre de service sélectionnés par le système,
- R_+ : est le nombre de service pertinents sélectionnés par le système.

Parce que les mesures précision et rappel sont des mesures bien connues et facilement expliquées, il est intéressant de les impliquer et de les étendre. Aussi, les mesures dérivées de la précision et du rappel apportent également un avantage majeur, comme F-mesure (ou connue aussi avec F-score), peut encore être calculée. La mesure F-score est définie comme suit :

$$F - score = \frac{2 * Précision * Rappel}{Précision + Rappel} \qquad \ldots(3)$$

Le tableau 6.1 illustre les valeurs des indicateurs précision,rappel et F-Score du système pour les huit (8) algorithmes de calcul de similarité sémantique suite à des requêtes quelconques et des seuils variés sur un data set de 60 Services Web. La zone de haute précision représente le cas où le système retourne peu de services non pertinents, alors que la zone de haut rappel concerne le cas où seule une petite partie des services pertinents, est omise par le système.

- La colonne "Algorithme" indique l'algorithme de matching utilisé ;
- La colonne "Précision" indique la valeur de la précision en % ;
- La colonne "Rappel" indique la valeur du rappel en % ;
- La colonne "F-Score" indique la valeur de la mesure f-score en % ;

Tab 6.1 – Synthèse des statistiques de la précision, rappel et F-Score pour le processus de découverte.

Algorithme	**Précision (%)**	**Rappel (%)**	**F-Score (%)**
RES [Resnik, 1995]	73,50	85,00	78,83
HSO [Hirst and Onge, 1998]	75,25	91,32	82,51
JCN [Jiang and Conrath, 1997]	68,45	81,20	74,28
WUP [Wu and Palmer, 1994]	74,20	91,60	81,99
LCH [Leacock and Chodorow, 1998]	65,69	80,12	72,19
PATH [Rada *et al.*, 1989]	71,60	90,40	79,91
LIN [Lin, 1998]	52,36	65,98	58,39
LESK [Lesk, 1986]	54,52	69,00	60,91

Le graphique des résultats de cette expérimentation (table 6.1) pour les huit (8) algorithmes est représenté dans la figure 6.7.

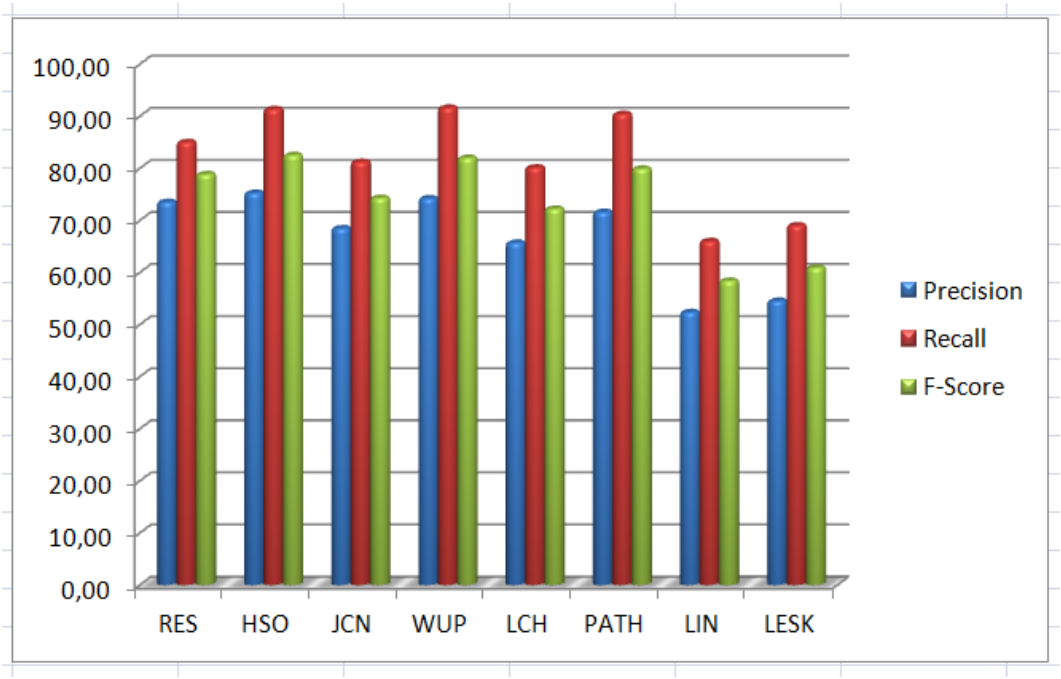


Fig 6.7 – Rapport Rappel/Précision

- La couleur bleue représente le taux de précision des différents algorithmes de similarité ;
- La couleur marron foncée désigne le taux de rappel des différents algorithmes de similarité ;
- La couleur verte représente le taux de précision des différents algorithmes de similarité

Sur la Figure 6.7, on distingue bien la différence de performances de découverte des différents algorithmes de calcul de similarité sémantique utilisé pour l'ensemble des collections de ce banc d'essai. Les algorithmes de HSO et WUP offrent des variantes nettement plus performantes que celles des autres algorithmes.

Sur le graphique ci-dessus, il est facile de conclure la relation entre la précision et le rappel pour tous les algorithmes de similarité sélectionnés. Le graphique montre qu'en termes de rappel et de précision, les mesures HSO et WUP donnent de meilleurs résultats par rapport les mesures LCH, RES, JCN et la mesure du PATH. Cependant, les mesures Lin et LESK donnent des résultats plus faibles. La performance des deux mesures HSO et WUP est conséquente avec d'autres évaluations des mesures de similarité WordNet.

Il est à noter que les résultats obtenus et présentés dans le tableau 6.1 pour les mesures HSO et WUP sont les meilleurs scores moyens enregistrés sur les tests réalisés avec les huit (8) algorithmes de calcul de similarité sémantique à base du WordNet.

6.6. Conclusion

Dans ce chapitre, nous avons présenté une étude expérimentale des prototypes que nous avons implémentés afin d'évaluer et de valider nos contributions qui rentrent dans le cadre de la ré-ingénierie et la découverte des Services Web.

En ce qui concerne la première approche relative à la ré-ingénierie des Services Web, nous avons développé un outil semi-automatique nommé WSDL2WSMO-LITE. Cet outil regroupe plusieurs modules complémentaires fournissant les fonctionnalités essentielles pour construire l'ontologie WSMO-Lite et a pour but de valider certains éléments du cadre méthodologique de l'approche que nous avons proposée dans le chapitre précédent.

Nous avons également expérimenté notre prototype sur des cas réels comme les analyses médicales, le tourisme, le e-Commerce (comme celui d'Amazon). L'expérimentation a révélé que l'approche est plus flexible au niveau de la spécification partielle des éléments du WSMO-Lite mais en revanche, elle a révélé aussi le fait qu'il peut y avoir des problèmes de complétudes dûes aux insuffisances des informations obtenues à partir du fichier WSDL, ce qui explique le besoin de prendre en charge d'autres sources d'informations telles que , les messages SOAP et les informations de l'UDDI et bien d'autres.

Nous avons aussi présenté un cadre de travail complet pour la découverte des Service Web basée sur les préférences des utilisateurs. Cette approche consiste à capturer un But("Goal") à partir d'une requête fondée par l'utilisateur et comparer ce But avec les Services Web disponibles en vue de retourner les Services appropriés répondant aux besoins des utilisateurs. Dans cette proposition, les différentes étapes sont impliquées en processus hors-ligne et en-ligne et de nouveaux outils sont produits pour assister le développeur.

Un test expérimental des techniques proposées a été mis en oeuvre, et qui a montré l'impact des algorithmes proposés afin d'optimiser le temps et l'effort des processus de ré-ingénierie et de découverte. En outre, les résultats expérimentaux sont prometteux et confirment que les approches proposées auront un impact positif sur les processus de ré-ingénierie et de découverte dans leur ensemble.

Chapitre

7 Conclusion Générale

Sommaire

7.1. Principales contributions

Le but de ce travail de thèse est de proposer nos approches de ré-ingénierie et de découverte des Services Web en utilisant le modèle concetuel de l'ontologie de service WSMO. Les deux contributions se basent sur l'utilisation des algorithmes de calcul de similarité sémantique à base du WordNet.

L'approche de ré-ingénierie se situe dans le contexte de reprise d'existant (ficher WSDL) et sa transformation en présentation partielle de l'ontologie WSMO-Lite. Cette transformation passe par plusieurs étapes, en commençant par la catégorisation du Service Web, en utilisant par un certain nombre de règles de mapping pour faire correspondre les informations du WSDL à des spécifications WSMO-Lite. Notre stratégie se base sur les informations fournies par le WSDL, en particulier les schémas XML, et les ontologies de domaine. Ainsi, l'idée d'utiliser des ontologies de domaine permet un modèle normalisé, plus riche sémantiquement et facile à exploiter par à la fois les êtres humains et les machines.

L'approche de découverte des Services Web consiste à formuler un but("Goal") à partir d'une requête de l'utilisateur et trouver les Services Web susceptibles de satisfaire ce "Goal". Cette requête est fournie sous forme d'informations introduites dans une page HTML. Le système mis en oeuvre reçoit en entrée des données et retourne en sortie un ou plusieurs Services Web correspondants aux besoins des utilisateurs.

L'approche de découverte utilise la notion de traçabilité, elle consiste à garder l'historique de

l'exécution dans une base de données mise à jour régulièrement afin de rendre la découverte plus rapide.

7.2. Principales limites

Le travail présenté dans ce mémoire est guidé par de nombreuses questions issues des préoccupations de la communauté des Services Web Sémantiques. Comme la problématique d'intégration est généralement complexe, nous nous sommes efforcés tout au long de ce travail de prendre suffisamment de recul afin de proposer un cadre relativement global et complet afin de mieux guider les utilisateurs dans leur processus d'intégration flexible.

Cependant, le travail présenté demeure incomplet et limité du fait que certaines limitations ont été effectuées. Il s'agit en particulier des limites liées à :

- L'insuffisance des travaux de recherche relatifs à l'ontologie de service WSMO, ainsi l'ontologie WSMO est liée principalement aux projets industriels et économiques ;
- L'insuffisance des informations fournies par le fichier WSDL ;
- La construction des ontologies de domaine ;
- La complexité du langage WSML, qui est encore en cours d'étude ;
- La difficulté d'automatiser le processus de manière totalement complète ;

7.3. Conclusion

A l'issue de ce travail, nous croyons pouvoir affirmer que l'intégration sémantique d'applications web constitue un problème dur et complexe qui nécessite à notre sens beaucoup de travaux de recherche.

Cette intégration doit être, à notre sens, basée sur des ontologies et elle est mieux résolue dans un contexte d'architecture orientée services.

7.4. Perspectives

Conscients des limites que présentent nos travaux et soucieux de la continuité des travaux qui peut être mise en œuvre, nous avons envisagé les perspectives principales suivantes :

- Etendre l'approche de ré-ingénierie : Un point important porte sur l'extension de ces travaux afin d'établir une spécification complète de l'ontologie WSMO-Lite en incluant à la fois les buts ("Goals") et les médiateurs. Il est à noter qu'à nos jours la formulation des goals et des médiateurs avec le framework WSMO-Lite constitue un projet de recherche

d'actualité ;

- Pour remédier au manque d'ontologies de domaine spécifiques, il est intéressant d'utiliser le paradigme des données liées ouvertes ("Linked Open data") pour extraire les ontologies de domaine appropriées.
- Améliorer l'approche de découverte et de sélection des services en utilisant d'autres propriétés non-fonctionnelles telles que la qualité des services afin d'obtenir une meilleure précision.
- Etendre l'architecture du système de découverte pour prendre en compte les aspects comportementals d'un service qui ne sont pas explicitement abordés par l'architecture de référence.

Enfin, il est clair qu'atteindre tous ces objectifs laissés en perspectives nécessitera un travail de longue haleine et dont l'aboutissement prendra sans doute plusieurs années. Cependant, nous demeurons convaincus, compte tenu de son intérêt, que ce travail doit être soutenu et poursuivi.

Annexe

A Le Langage WSML

Sommaire

A.1. Introduction

Le langage de modélisation des Services Web (WSML : Web Service Modeling Language) est un langage pour la spécification des différents aspects des Services Web Sémantiques [De Bruijn *et al.*, 2006]. Il fournit un langage formel pour la modélisation des Services Web selon les spécifications de l'ontologie WSMO et est basé sur les formalismes logiques bien connus, en précisant un cadre linguistique cohérent pour la description des Services Web Sémantiques.

A.2. Couches du WSML

Le langage WSML propose un certain nombre de variantes avec différentes expressivités. Toutes les variantes et leurs inter-relations sont illustrées dans la Figure A.1. L'extension signifie que toutes les spécifications valides d'une variante étendue représentent également une spécification valide de la nouvelle variante.

– WSML-Core se base sur l'intersection entre la logique de description SHIQ et la logique de Horn [Grosof *et al.*, 2003]. Les éléments de base de WSML-Core sont les concepts, les attributs, les relations binaires et les instances. WSML-Core supporte aussi les types de données, ainsi que l'hiérarchie des concepts et des relations.

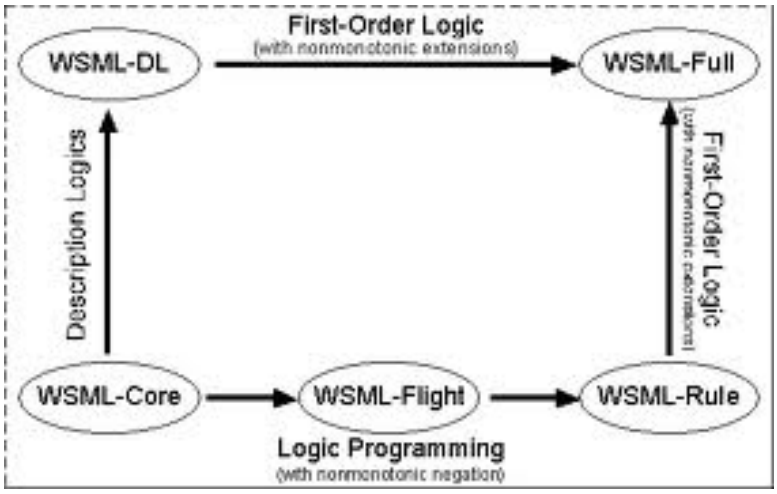


Fig A.1 – Les différentes variantes du langage WSML selon [De Bruijn *et al.*, 2006].

- WSML-DL repose sur la logique de description SHIQ (D), qui est une partie de OWL-DL [Dean and Schreiber, 2004].
- WSML-Flight étend WSML-Core en ajoutant un ensemble de caractéristiques telles que la méta-modélisation, les contraintes et la négation monotone. Cela permet de fournir un langage de règles puissant.
- WSML-Rule étend WSML-Flight avec plus de caractéristiques de la programmation logique, à savoir l'utilisation des symboles de fonction, les règles incertaines et la négation non stratifiée.
- WSML-Full regroupe WSML-DL et WSML-Rule avec des extensions permettant de supporter la négation monotone de WSML-Rule. La sémantique de WSML-Full est actuellement un problème de recherche ouvert.

Comme illustré dans la Figure A.1, WSML possède deux alternatives de représentation en couche, notamment : WSML-Core => WSML-DL => WSML-Full et WSML-Core => WSML-Flight => WSML-Rule => WSML-Full.

Les deux alternatives commencent par WSML-Core et se terminent par WSML-Full. Les deux alternatives sont disjointes dans le sens où l'interopérabilité dans WSML, entre la variante de description logique (WSML-DL) d'un côté et la variante de programmation logique (WSML-Flight et WSML-Rule) de l'autre côté, n'est possible qu'à travers un noyau commun (WSML-Core) ou à travers un super-ensemble expressif (WSML-Full).

A.3. Syntaxe générale du langage WSML

WSML fournit une syntaxe générale pour la spécification des ontologies, des Services Web, des buts et des médiateurs. Cette syntaxe est partagée entre toutes les variantes, à l'exception

de quelques restrictions qui s'appliquent pour la spécification des ontologies en WSML-Core et WSML-DL.

a) Les ontologies

Une ontologie en WSML porte sur cinq éléments : concept, relation, instance, instance de relation (lien) et axiome. En plus, une ontologie peut avoir des propriétés non-fonctionnelles et peut importer d'autres ontologies. Nous commençons la description des ontologies WSML par un exemple qui montre les éléments d'une ontologie dans le Listing A.1, ensuite nous allons détailler ces éléments.

```
wsmlVariant _"http://www.wsmo.org/wsml/wsml-syntax/wsml-flight"
namespace {_"http://example.org/bookOntology#", dc _"http://purl.org/dc/elements/1.1/"}
ontology _"http://example.org/bookOntology"
   nonFunctionalProperties
        dc#title hasValue "Example Book ontology"
        dc#description hasValue "Example ontology about books and shopping carts"
   endNonFunctionalProperties
concept book
        title ofType _string
        hasAuthor ofType author
        concept author subConceptOf person
        authorOf inverseOf(hasAuthor) ofType book
concept cart
    nonFunctionalProperties
        dc#description hasValue "A shopping cart has exactly one id
        and zero or more items, which are books."
   endNonFunctionalProperties
         id ofType (1) _string
         items ofType book
```

Listing A.1 – Exemple d'une ontologie WSML

- Les concepts : la notion de concept (parfois appelée classe) joue un rôle central dans les ontologies. Les concepts forment la terminologie de base du domaine de discours. Un concept peut avoir des instances et on peut lui associer des attributs. Les propriétés non-fonctionnelles et les attributs sont groupés ensemble dans un seul fragment, comme c'est le cas pour le concept book dans l'exemple précédent.

  La définition des attributs peut prendre deux formes : par contrainte (en utilisant **ofType**) ou par inférence (en utilisant **impliesType**). La définition des attributs par contrainte spécifie des contraintes de type sur les valeurs d'un attribut, similaires aux contraintes d'intégrité de bases de données. La définition des attributs par inférence implique que le type des valeurs de cet attribut est inféré à partir de la définition de l'attribut, similaire à la restriction du "range" d'une propriété en RDF(S) [Brickley and Guha, 2004]et OWL [Dean and Schreiber, 2004]. Chaque définition d'un attribut peut être associée

d'un nombre de caractéristiques, notamment la transitivité, la symétrie, la réflexivité, l'inverse d'un attribut, ainsi que les contraintes de cardinalité maximale et minimale.

La définition des attributs par contrainte et les contraintes de cardinalité ne sont pas permises dans WSML-Core et WSML-DL. A l'inverse des caractéristiques du rôle dans les logiques de description, les caractéristiques des attributs telles que la transitivité, la symétrie, la réflexivité et l'inverse d'un attribut sont locales pour un concept en WSML. Ainsi, aucune de ces caractéristiques ne peut être utilisée en WSML-Core et WSML-DL.

- Les relations : les relations en WSDL peuvent avoir une dimension arbitraire, et peuvent être organisées en hiérarchie en utilisant **subRelationOf**. Les paramètres peuvent être typés en utilisant une définition de type de paramètre de la forme (**ofType** type) ou (**impliesType** type), où type est un identificateur de concept. **ofType** et **impliesType** sont utilisés comme pour la définition des attributs. Ainsi, la définition des paramètres par le mot-clé **ofType** est utilisée pour vérifier le type des valeurs des paramètres, tandis que la définition des paramètres par le mot-clé **impliesType** est utilisée pour inférer les concepts associés aux valeurs des paramètres.

  La dimension permise de la relation dépend de la variante du langage WSML. WSML-Core et WSML-DL permettent uniquement des relations binaires et, de même que pour la définition des attributs, ils permettent de donner un type de paramètre par le mot-clé **impliesType**.

- Les instances : un concept peut avoir un nombre d'instances. Les instances spécifiées explicitement dans une ontologie sont celles partagées comme une partie de l'ontologie.

  Cependant, la plupart des instances existent hors ontologie, dans des bases de données privées. WSML ne décrit pas comment se connecter à la base de données à partir d'une ontologie, puisque différentes organisations vont utiliser la même ontologie pour interroger différentes bases de données qui sont typiquement non partagées.

  Une instance peut être membre de zéro ou plusieurs concepts et on peut lui associer un nombre de valeurs d'attributs. Notons que la spécification des concepts est optionnelle et les attributs utilisés dans la spécification de l'instance ne doivent pas nécessairement apparaître dans la définition du concept associé. Par conséquent, les instances WSML peuvent être utilisées pour représenter des données semi-structurées, puisque sans les concepts et les contraintes sur l'utilisation des attributs, les instances forment un graphe étiqueté orienté. Grâce à cette possibilité de représenter des données semi-structurées, la plupart des graphes RDF peuvent être représentés comme données instances en WSML, et vice versa.

- Les axiomes : les axiomes permettent d'ajouter des expressions logiques arbitraires à une ontologie. Ces expressions logiques peuvent raffiner la définition des concepts et des relations dans une ontologie. Elles permettent aussi d'ajouter des connaissances axiomatiques arbitraires ou exprimer des contraintes.

b) Les Services Web

Un Service Web possède une fonctionnalité(capabilité) et un nombre d'interfaces. La fonctionnalité est décrite en exprimant les conditions sur son pré et poste-état (l'état avant et après l'exécution du service Web) en utilisant des expressions logiques. Les interfaces décrivent comment interagir avec le service. En plus, WSML permet la spécification des propriétés non-fonctionnelles du Service Web. Listing A.2 décrit un simple Service Web pour l'ajout des items dans un panier d'achat (shopping cart).

```
webService _"http://example.org/bookService"
  nonFunctionalProperties
       dc#title \bf{hasValue} "Example book buying service"
       dc#description hasValue "A simple example web service for adding
       items to a shopping cart"
   endNonFunctionalProperties
importsOntology _"http://example.org/bookOntology"
capability
      sharedVariables {?cartId, ?item}
      precondition
           definedBy
                 ?cartId memberOf _string and ?item memberOf book.
      postcondition
           definedBy
               forall ?cart (?cart[ id hasValue ?cartId] memberOf cart implies
                    ?cart[items hasValue ?item]).
```

Listing A.2 – Exemple d'un Service Web WSML

La fonctionnalité

la fonctionnalité implique plusieurs éléments, à savoir les pré-conditions, les post-conditions, les suppositions et les effets. Les pré-conditions et les suppositions décrivent l'état avant l'exécution du service Web.

Tandis que les pré-conditions décrivent les conditions sur l'espace d'information, i.e. les conditions sur les inputs, les suppositions décrivent les conditions qui peuvent ne pas être vérifiées directement.

-Les post-conditions décrivent la relation entre les inputs et les outputs. Dans ce sens, elles décrivent l'état de l'espace d'information après l'exécution du service.

-Les effets décrivent le changement dans le monde réel causé par le service.

-La structure **sharedVariables** est utilisée pour indiquer les variables partagées entre les pré-conditions, les post-conditions, les suppositions et les effets. Les variables partagées peuvent être utilisées pour référencier les mêmes inputs et outputs.

Le listing A.2 décrit un simple Service Web pour l'ajout des items dans un panier d'achat : on donne l'identificateur du panier d'achat et un nombre d'items. Les items sont ajoutés dans le panier d'achat ayant cet identificateur.

Les interfaces

Les interfaces décrivent comment le client interagit avec le service (chorégraphie) et comment le service interagit avec les autres services (orchestration). Les descriptions de la chorégraphie et de l'orchestration sont externes. WSML permet de référencier la chorégraphie et l'orchestration par des IRIs.

Notons que WSML ne spécifie pas encore tous les aspects fonctionnels des services. La description du comportement dynamique des Services Web (chorégraphie et orchestration) dans le contexte de WSMO est en cours de développement, mais n'est pas encore intégré dans WSML.

c) Les buts("Goals")

les buts sont symétriques aux Services Web dans le sens où les buts décrivent les fonctionnalités désirées et les Services Web décrivent les fonctionnalités offertes. Ainsi, la description des buts est constituée des mêmes éléments de modélisation utilisés pour la description d'un Service Web, notamment les propriétés non fonctionnelles, la fonctionnalité et les interfaces.

d) Les médiateurs

les médiateurs font la liaison entre les différents buts, services Web et ontologies, et permettent l'interopérabilité en résolvant le problème de disparités entre les différents formats de représentation, modes de codage, protocoles de traitement, etc. La connexion entre les médiateurs et les autres éléments WSML peut être établie par deux différentes façons :

- Chaque élément WSML prend en compte les spécifications d'un certain nombre de médiateurs par le mot clé **uses Mediator**.
- Chaque médiateur (selon son type) possède une ou plusieurs sources, et une cible. La source et la cible sont optionnelles pour permettre des médiateurs génériques.

Un médiateur effectue sa fonctionnalité de médiation soit par un service, qui fournit le service de médiation, soit par un but qui peut être utilisé pour découvrir dynamiquement le service Web (de médiation) approprié.

Annexe

B Mesures de Similarité Sémantique

Sommaire

B.1. Introduction

L'identification de la similarité dans les ontologies est un concept fondamental qui est adopté par plusieurs techniques telles que le regroupement, la fouille de données (data mining), le Web Sémantique et en particulier, le domaine de la recherche de l'information. Cette dernière repose largement sur des mesures pour l'identification de la similarité entre les documents. Les rapports ontologiques entre les mots peuvent être détectés par un processus de calcul de similarité entre des paires d'objets contenus dans l'ontologie.

Dans la littérature, plusieurs travaux sur la mesure de similarité sémantique entre les objets d'une ontologie ont été développés dans différents contextes. Les méthodes qui utilisent WordNet peuvent fournir des résultats excellents [Meng *et al.*, 2013].

B.2. Définitions

Une mesure de similarité normalisée est une fonction : Sim : $E * E \rightarrow [0 \cdot \cdot 1]$ appliquée à une paire d'éléments (exemple : chaînes de caractères, nombres) et qui renvoie un nombre réel dans $[0 \cdot \cdot 1]$ exprimant la similarité entre ces deux éléments telle que :

- Sim(a, a) $= 1$ (maximalité)
- $\forall a, b \sqsubseteq E, Sim(a, b) = Sim(b, a)$ $\cdot \cdot \cdot \cdot$ (Symétrie)

B.3. WordNet

WordNet est une ressource lexicale de large couverture, développée depuis plus de 20 ans pour la langue anglaise. Elle est utilisable librement, y compris pour un usage commercial, ce qui en a favorisé une diffusion très large. Plusieurs autres ressources linguistiques ont été constituées (manuellement ou automatiquement) à partir de, en extension à, ou en complément à WordNet. Des programmes issus du monde de l'Intelligence Artificielle ont également établi des passerelles avec WordNet.

WordNet est une base de données lexicale développée depuis 1985 par des linguistes du laboratoire des sciences cognitives de l'université de Princeton. C'est un réseau sémantique de la langue anglaise, qui se fonde sur une théorie psychologique du langage. La première version diffusée remonte à juin 1991. Son but est de répertorier, classifier et mettre en relation de diverses manières le contenu sémantique et lexical de la langue anglaise. Le système se présente sous la forme d'une base de données électronique qu'on peut télécharger sur un système local. Des interfaces de programmation sont disponibles pour de nombreux langages.

B.4. Mesures de similarité à base de WordNet

Dans la section suivante, nous présentons l'ensemble des mesures de similarité sémantique à base de WordNet, celles utilisées dans le processus de découverte.

a) Mesure de Lesk

L'algorithme de Lesk [Lesk, 1986]est une méthode de désambiguïsation bien connue qui consiste à compter le nombre de mots communs entre les définitions d'un mot (généralement trouvées dans un dictionnaire électronique) et les définitions des mots de son contexte. Le sens retenu correspond à la définition pour laquelle on compte le plus de mots communs avec le contexte. Cette idée simple a donné des résultats intéressants et s'est avérée meilleure que bon nombre de techniques plus évoluées.

b) Mesure de Wu et Palmer

La mesure de similarité de Wu et Palmer est une autre mesure de similarité prenant en compte à la fois (1) la profondeur des concepts comparé dans la hiérarchie et (2) la structure de cette hiérarchie par la proximité relative de leur père commun. La similarité entre $c1, c2 \sqsubseteq C$ est définie par :

La mesure de similarité de [Wu and Palmer, 1994] est basée sur le principe suivant : Etant

donnée une ontologie formée par un ensemble de noeuds et un noeud racine R. Soit C1 et C2 deux éléments de l'ontologie dont nous allons calculer la similarité. Le principe de calcul de similarité est basé sur les distances (N1 et N2) qui séparent les noeuds C1 et C2 du noeud racine et la distance qui sépare le concept subsumant [1] (CS) de C1 et de C2 du noeud R.

$$\boxed{Sim(C1, C2) = 2 * \frac{2*N}{N1+N2}}$$

Où N représente la distance entre le noeud R et le concept subsumant CS.

c) Mesure de Resnik

La notion du contenu informationnel (CI) a été initialement introduite par [Resnik, 1995] qui a prouvé qu'un objet (mot) est défini par le nombre des classes spécifiées et que la similarité sémantique entre deux concepts est mesurée par la quantité de l'information qu'ils partagent. Pour évaluer la pertinence d'un objet il faut calculer le contenu informationnel. Le contenu informationnel est obtenu en calculant la fréquence de l'objet dans le corpus (WordNet). La formule proposée par Resnik est définie par :

$$\boxed{Sim(C1, C2) = Max[E(CS(C1, C2))] = Max[-log(p(CS(C1, C2))]}$$

Où CS(C1, C2) représente le concept le plus spécifique (qui maximise la valeur de similarité) qui subsume (situé à un niveau hiérarchique plus élevé) les deux concepts C1 et C2 dans l'ontologie. Cette mesure est un peu sommaire car elle ne dépend que du concept le plus spécifique.

d) Mesure de Lin

Lin propose d'évaluer la similarité entre deux concepts, en tenant compte à la fois de leur contenu informatif commun (comme la mesure de Resnik) mais également de leurs caractéristiques propres. La similarité entre $C1, C2 \sqsubseteq C$ est définie par :

$$\boxed{Sim(C1, C2) = \frac{2*\psi(cccom)}{\psi(C1)+\psi(C2)}}$$

Cette mesure utilise une approche hybride qui combine deux sources de connaissances différentes (Thesaurus, corpus). En plus, elle représente la similarité comme degré probabiliste de chevauchement des concepts descendants de C1 et C2.

1. Le concept subsumant c'est le concept commun le plus spécifique.

e) Mesure de Rada

Cette mesure [Rada *et al.*, 1989] est adoptée dans un réseau sémantique et elle est fondée sur le fait qu'on peut calculer la similarité en se basant sur les liens hiérarchiques "is-a".

Pour calculer la similarité de deux concepts dans une ontologie, on doit calculer le nombre des arcs minimums qui les séparent. Cette mesure, basée sur le calcul de la distance entre les noeuds par le chemin le plus court, présente un des moyens les plus évidents pour évaluer la similarité sémantique dans la hiérarchie d'une ontologie .

dist_edge(C1, C2) représente la longueur du plus court chemin entre deux concepts C1, C2 dans la hiérarchie de concepts, la similarité entre $C1, C2 \sqsubseteq C$ est définie par :

$$\boxed{Sim(C1, C2) = \frac{1}{dist\_edge(C1,C2)}}$$

f) Mesure de Hirst et Onge

L'idée de cette mesure est que deux concepts lexicalisés sont sémantiquement étroits si leurs ensembles synonymes (synsets) de WordNet sont reliés par un chemin qui n'est pas trop long et qui "ne change pas la direction trop souvent". Avec cette mesure, toutes les relations contenues dans un réseau WordNet sont prises en considération. Dans le travail de [Hirst and Onge, 1998], les auteurs ont classé la direction des liens en lien haut (super-classe), lien bas (sous-classe) et lien horizontal (antonyme). Le calcul de la similarité, avec cette méthode, s'effectue entre objets (mots) par le poids du plus court chemin allant d'un terme à un autre, en plus des classifications qui indiquent les changements de direction. La force du rapport est donnée par :

$$\boxed{Simht = T - PCC - K * nd}$$

Où T et K sont des constantes, PCC est la distance du plus court chemin en nombre d'arc et nd le nombre de changements de direction.

g) Mesure de Jiang et Conrath

Pour remédier au problème présenté au niveau de la mesure de Resnik, [Jiang and Conrath, 1997] a apporté une nouvelle formule qui consiste à combiner le contenu informationnel du concept spécifique à celui du concept dont on cherche la similarité (combine les techniques basées sur les arcs et les techniques basées sur les noeuds qui consistent à compter les arcs afin d'améliorer les résultats par des calculs basés sur les noeuds). La mesure adoptant cette méthode est basée sur la combinaison d'une source de connaissance riche

(thesaurus) avec une source de connaissance pauvre (corpus). Notons que cette formule est définie par l'inverse de la distance sémantique. La similarité entre C1 et C2 est définie par :

$$\boxed{Sim(C1, C2) = \frac{1}{distance(C1,C2)}}$$

Sachant que la distance entre X et Y est calculée par la formule suivante :

$$\boxed{distance(C1, C2) = E(C1) + E(C2) - (2 * E(CS(C1, C2)))}$$

h) Mesure de Leacock et Chodorow

Une autre méthode présentée par [Leacock and Chodorow, 1998] qui combine entre la méthode de comptage des arcs et la méthode du contenu informationnel. La mesure proposée par Leacock et Chodorow [Leacock and Chodorow, 1998] est basée sur la longueur du plus court chemin entre deux synsets de Wordnet. Les auteurs ont limité leur attention à des liens hiérarchiques "is-a" ainsi que la longueur de chemin par la profondeur globale P de la taxonomie. La formule est définie par :

$$\boxed{Sim(C1, C2) = -log(\frac{cd(C1,C2)}{2*M})}$$

où M est la longueur du chemin le plus long qui sépare le concept racine, de l'ontologie, du concept le plus en bas. On dénote par cd(C1, C2), la longueur du chemin le plus court qui sépare C1 de C2.

Références bibliographiques

[Acuna and Marcos, 2006] C-J. Acuna and E. Marcos. Modeling Semantic Web Services : a case study. In proceedings of the 6th international conference on Web engineering (ICWE 2006). ACM Press, New York, NY, USA, 32–39, 2006.

[Agre *et al.,* 2007] G. Agre, Z. Marinova, T. Pariente and A. Micsik. Towards Semantic Web Service engineering. Service Matchmaking and Resource Retrieval in the Semantic Web, p. 91, 2007.

[Aljoumaa *et al.,* 2010] K. Aljoumaa, Saïd Assar and Carine Souveyet. Publishing intentional services using new annotation for WSDL, iiWAS '10 Proceedings of the 12th International Conference on Information Integration and Web-based Applications and Services, Pages 881-884, 2010.

[Amar Bensaber and Malki, 2008] D. Amar Bensaber and M. Malki. Development of semantic web services : Model driven approach. In proceedings of the 8th international conference on New technologies in distributed systems, p. 40. ACM, 2008.

[Bellwood *et al.,* 2005] T. Bellwood, L. Clement, and C. von Riegen. "Universal description, discovery and Integration. Technical report, OASIS UDDI Specification Technical Committee". Http ://www.oasis-open.org/cover/uddi.html, 2005.

[Bener *et al.,* 2009] A. B. Bener, V. Ozadali and E. S. Ilhan. Semantic Matchmaker with Precondition and Effect Matching Using SWRL. Expert Systems with Applications 36 (5), 9371–9377, 2009.

[Bentahar *et al.,* 2007] J. Bentahar, Z. Maamar, D. Benslimane and P. Thiran. An Argumentation Framework for Communities of Web Services. IEEE Intelligent Systems, 2007.

[Berners-Lee *et al.,* 2001] T. Berners-Lee, J. Hendler and O. Lassila. The Semantic Web. Scientific American,284(5) :34 - 43, 2001.

[Booth *et al.,* 2004] D. Booth, H. Haas, F. McCabe, E. NewCorner, M. Champion, C. Ferris and D. Orchard, W3c working group note - Web Services Architecture, 2004.

[Borst, 1997] W. Borst. Construction of Engineering Ontologies. PhD thesis, University of Tweente, Enschede, NL-Centre for Telematica and Information Technology, 1997.

[Bouchiha *et al.,* 2012] D. Bouchiha, M. Malki, A. Alghamdi and K. Alnafjan. Semantic Web Service Engineering : Annotation Based Approach. COMPUTING AND INFORMATICS, Vol 31, No 6+, 2012.

[Brambilla *et al.,* 2007] M. Brambilla, S. Ceri and F-M. Facca. Model-Driven Design and Development of Semantic Web Service Applications. ACM Transactions on Internet Technology (TOIT), Volume 8, Issue 1, 2007.

[Brickley and Guha, 2004] D. Brickley and R-V Guha. RDF Vocabulary Description Language 1.0 : RDF Schema. W3C Recommendation. Available at : http ://www.w3.org/TR/rdf-schema/, 2004.

[Cabral *et al.,* 2004] L. Cabral, J. Domingue, E. Motta, T. Payne and F. Hakimpour. Approaches to Semantic Web Services : An Overview and Comparisons. Pages 225 - 239, 2004.

[Castano *et al.,* 2006] S. Castano, A. Ferrara and S. Montanelli. Matching Ontologies in Open Networked Systems : Techniques and Applications, journal on Data Semantic, vol. 5, pp. 25-63, 2006.

[Chabeb and Tata 2010] Y. Chabeb and S. Tata. Publication and discovery of YASA web services, in Proceedings ACM Symposium on Applied Computing (SAC'10), New York, USA, p. 2493–2494, 2010.

[Chauvet, 2002] JM. Chauvet. Services Web avec SOAP, WSDL, UDDI,ebXML. Eyrolles, Paris, 2002.

[Chikofsky *et al.,* 1990] E-J. Chikofsky and J-H Cross. Reverse engineering and design recovery : A taxonomy. IEEE Softw., 7(1) :13–17, 1990.

[Chinnici *et al.,* 2007] R. Chinnici, J.J Moreau, A. Ryman, and S. Weerawarana. Web Services description language (wsdl) version 2.0, 2007.

[da Silva *et al.,* 2009] L. da Silva Santos, G. Guizzardi, L. Pires, M. van Sinderen, C.A. Heuser, and G. Pernul From user goals to service discovery and composition, in Advances in Conceptual Modeling – Challenging Perspectives ER 2009 Workshops, vol. 5833, Springer, p.265–274, 2009.

[Dean and Schreiber, 2004] M. Dean and G. Schreiber. OWL Web Ontology Language Reference. W3C Recommendation. Available at http ://www.w3.org/TR/owl-ref/, 2004.

[De Bruijn *et al.,* 2006] J. De Bruijn, H.Lausen, A. Polleres and D. Fensel. The Web Service Modeling Language WSML : An Overview. Book chapter in "The Semantic Web : Research and Applications" book. Springer Berlin/Heidelberg, 2006.

[De Paoli *et al.,* 2008] F. De Paoli, M. Palmonari, M. Comerio and A. Maurino. A Meta-model for Non-functional Property Descriptions of Web Services. IEEE Int. Conf. Web Serv., no. 1, pp. 393–400, 2008.

[Do *et al.,* 2002] H. Do, S. Melnik, E. Rahm. Comparison of schema matching evaluations. Proceedings of the 2nd Int. Workshop on Web Databasess, German Informatics Society, Erfurt, Germany, p. 221-237, 2002.

[Dong *et al.,* 2004] X. Dong, A. Halevy, J. Madhavan, E. Nemes and J. Zhang. Similarity Search for Web Services. In the 30th International Conference on Very Large Data Bases. Vol. 30. pp. 372–383, 2004.

[Driss *et al.,* 2010] M. Driss, N.Moha, Y. Jamoussi, J.M. Jezequel and H. H. Ben Ghezala. A Requirement-Centric Approach to Web Service Modeling, Discovery, and Selection, in 8th Int. Conf. on Service-Oriented Computing (ICSOC), LNCS, vol. 6470, P. P. Maglio, M. Weske, J. Yang, and M. Fantinato, Ed. Springer, p. 258–272, 2010.

[Duo *et al.,* 2005] Z. Duo, L.Juan-Zi and X. Bin. Web Service Annotation Using Ontology Mapping. Proceedings of the IEEE International Workshop on Service-Oriented System Engineering (SOSE'05), IEEE, pp. 235-242, 2005.

[El Bouhissi *et al.,* 2009a] H. EL BOUHISSI, M.Malki and D. Bouchiha. Towards WSMO Ontology Specification From Existing Web Services, 2nd Conférence Internationale sur l'informatique et ses Applications (CIIA'09) Saida, Algeria, 2009.

[El Bouhissi *et al.,* 2009b] H. EL Bouhissi, M. Malki, M.K Rahmouni and D. Bouchiha. Semantic evolution in web services : WSMO ontology. Second International on Web and Information Technologies, ICWIT'09, Kerkenah - Tunisia.

[El Bouhissi and Malki, 2009c] H. EL Bouhissi and M. Malki. Reverse Engineering Existing Web Service Applications. 16th Working Conference on reverse Engineering, WCRE'09, Lille - France, 2009.

[El Bouhissi and Malki, 2011] H. EL Bouhissi and M. Malki. Semantic Web Services : WSMO Goal Based Architecture. 2nd International Conference on Information and Communication Systems, ICICS'11, Irbid – Jordan, 2011.

[El Bouhissi and Malki, 2014a] H. EL Bouhissi and M. Malki. Building Semantic Web Service Ontology : A Reverse Engineering Approach. 1st International Conference on Information Technology for Organisation Development (IT4OD), Tebessa – Algeria, 2014.

[El Bouhissi and Malki, 2014b] H. EL Bouhissi and M. Malki. Semantic Web Service Ontology Construction : A Reverse Engineering Approach. The 11th ACS/IEEE International Conference on Computer Systems and Applications (AICCSA'2014), November 10-13, 2014, Doha, Qatar.

[El Bouhissi *et al.,* 2014c] H. EL Bouhissi, M. Malki and M. A. Sidi Ali Cherif. Improve Web Service Discovery : Goal Based Approach. The 11th ACS/IEEE International Conference on Computer Systems and Applications (AICCSA'2014), November 10-13, 2014, Doha, Qatar.

[El Bouhissi *et al.,* 2014d] H. EL Bouhissi, M. Malki and 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 (IJITWE), 9(3), 15-39, July-September, 2014.

[Farrell and Lausen,2007] J. Farrell and H. Lausen. SAWSDL : Semantic Annotations for WSDL and XML Schema, rapport technique, W3C, http ://www.w3.org/TR/sawsdl/, 2007.

[Fensel and Bussler, 2002] Dieter Fensel and Christoph Bussler, The Web Service Modeling Framework WSMF. Electronic Commerce Research and Applications ,1(2) :113-137, 2002.

[Fensel *et al.,* 2010] D. Fensel, F. Fischer, J. Kopecký, R. Krummenacher, D. Lambert and T. Vitvar. WSMO-Lite : Lightweight Semantic Descriptions for Services on the Web. W3C Member Submission 23 August, 2010.

[Gomez *et al.,* 2004] J. M. Gomez, M. Rico, F. Garcia-Sanchez, R. M. Bejar, C.Bussler. GODO : Goal Driven Orchestration for Semantic Web Services. In the 1st Workshop on Web Services Modeling Ontology Implementations (WIW 2004). Vol. 113. CEUR Workshop Proceedings, 2004.

[Gronmo *et al.,* 2005] R. Gronmo, M-C.Jaeger and H. Hoff. Transformations between UML and OWL-S, Springer-Verlag. The European Conference on Model Driven Architecture - Foundations and Applications (ECMDA-FA), Nuremberg, Germany, 2005.

[Grosof *et al.,* 2003] B. N. Grosof, I. Horrocks, R. Volz, and S. Decker. Description logic programs : Combining logic programs with description logic. In Proc. of the Twelfth International World Wide Web Conference (WWW 2003), pages 48-57. ACM, 2003.

[Gruber, 1993] T. Gruber. A translation approach to portable ontology specifications, Knowledge Acquisition 5(2), pages 199-220, 1993.

[Guarino and Giaretta, 1995] N. Guarino and P. Giaretta. Ontologies and knowledge bases : Towards a terminological clarification. Towards Very Large Knowledge Bases : Knowledge Building and Knowledge Sharing, pages 25–32, 1995.

[Heß *et al.,* 2008] A. Heß, E. Johnston and N. Kushmerick. ASSAM : A Tool for Semi-Automatically Annotating Semantic Web Services. The 12th Int'l Conf. Web Technologies, pp. 470-475, 2008.

[Heijst *et al.,* 1997] G-V. Heijst, G. Schreiber and B. J. Wielinga. Using explicit ontologies in kbs development. Int. J.Hum.-Comput. Stud., 46(2-3) :183- 292, 1997.

[Hirst and Onge, 1998] G.Hirst et D.St Onge. Lexical chains as representations of context for the detection and correction of malapropisms. In Christiane Fellbaum (editor), WordNet : An electronic lexical database, Cambridge, MA :The MIT Press, 1998.

[Jaeger *et al.,* 2005] M-C. Jaeger, L. Engel and K. Geihs. A methodology for developing owl-s descriptions. The First International Conference on Interoperability of Enterprise Software and Applications Workshop on Web Services and Interoperability, 2005.

[Janev and Vranes, 2009] V. Janev and S. Vranes. Semantic Web Technologies : Ready for Adoption ?. IT Professional, vol. 11, no. 5, pp. 8-16,2009.

[Jiang and Conrath, 1997] J. J. Jiang and D. W. Conrath. Semantic similarity based on corpus statistics and lexical taxonomy", Proceedings of International Conference on Research in Computational Linguistics, August 22-24 ; Taipei, TaiWan, 1997.

[Karray *et al.,* 2013] A. Karray, R. Teyeb and M. Ben Jemaa , A Heuristic Approach for Web Service Discovery and Selection. In International Journal of Computer Science & Information Technology (IJCSIT) Vol 5, No 2, April, 2013.

[Keller *et al.,* 2004] U. Keller, R. Lara, A. Polleres, I. Toma, M. Kifer and D. Fensel. Wsmo Web Service discovery, 2004.

[Kifer *et al.,* 1995] M. Kifer, G. Lausen, and J. Wu. Logical foundations of object-oriented and frame-based languages. Journal of the ACM, 42 :741-843, 1995.

[Klusch and Kapahnke, 2009] M. Klusch and P. Kapahnke. OWLS-MX3 : An adaptive hybrid semantic service matchmaker for OWL-S. In Proc. 3rd Int. SMR2 Workshop on Service Matchmaking and Resource Retrieval in the Semantic Web, Washington DC, USA, , Vol-525, 2009.

[Kuropka *et al.,* 2008] D. Kuropka, P. Trger, S. Staab, M.Weske, D. Kuropka,P. Trger, S. Staab and M. Weske. Semantic service provisioning. Springer Publishing Company, Incorporated, 2008.

[Leacock and Chodorow, 1998] C. Leacock et M. Chodorow. Combining Local Context and WordNet Similarity for Word Sense Identification. In WordNet : An Electronic Lexical Database, C. Fellbaum, MIT Press, 1998.

[Lehman and Belady, 1985] M.M. Lehman and L.A. Belady. Program Evolution Processes of Software Change. Academic Press, London, 1985.

[Lei *et al.,* 2008] Z. Lei, Y. Xiaoying, Y. Yanni, and S. Bo. An Improved Semantic Annotation Method of Web Services Based on Ontology. In Proceedings of the 2008 ISECS International Colloquium on Computing, Communication, Control and Management ACM, pp 580-584, 2008.

[Lenat and Guha, 1990] D-B. Lenat and R-V. Guha. Building Large Knowledge-Based Systems ; Representation and Inference in the Cyc Project. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1990.

[Lesk, 1986] M. LESK. Automatic sense disambiguation using machine readable dictionaries : how to tell a pine cone from an ice cream cone. In SIGDOC '86, pages 24–26, New York, NY, USA. ACM, 1986.

[Lin, 1998] [Lin-1998] D. Lin. An Information-Theoretic Definition of similarity. In Proceedings of the Fifteenth International Conference on Machine Learning (ICML'98). Morgan Kaufmann : Madison, WI, 1998.

[Martin *et al.,* 2004] D. Martin, M.Paolucci, S. McIlraith, M. Burstein, D. McDermott, D. McGuinness, B. Parsia, T. Payne, M. Sabou, M. Solanki, N. Srinivasan and K. Sycara. Bringing

Semantics to Web Services : The OWL-S Approach. In Proceedings of the First International Workshop on Semantic Web Services and Web Process Composition (SWSWPC 2004), San Diego, California, USA, 2004.

[Meng *et al.,* 2013] L. Meng, R. Huang and J. Gu. A Review of Semantic Similarity Measures in WordNet. International Journal of Hybrid Information Technology. Vol. 6, No., 2013.

[Miller *et al.,* 2005] J. Miller, M. Nagarajan, M. Schmidt, A. Sheth, K. Verma, R. Akkiraju and J. Farrell. Web service semantics : Wsdl-s. Technical report, IBM, 2005.

[Mizoguchi, 1998] R. Mizoguchi. A step towards ontological engineering. In 12th National Conference on AI of JSAI, pages 24-31, Osaka, Japan, 1998.

[Oldham *et al.,* 2004] N. Oldham, C. Thomas, A. Sheth, and K. Verma. METEOR-S Web Service Annotation Framework with Machine Learning Classification. The First Int'l Workshop Semantic Web Services and Web Process Composition, pp. 137-146, 2004.

[Oundhakar *et al.,* 2005] S. Oundhakar, K. Verma, K. Sivashanmugam, A. Sheth, and J. Miller, Discovery of web services in a multi-ontology and federated registry environment, International Journal of Web Services Research, vol. 2, no. 3, pp. 1–32, 2005.

[Paolucci *et al.,* 2002] M. Paolucci, T. Kawamura, T. Payne, K. Sycara. Semantic matching of web services capabilities.In : Proceedings of the First International Semantic Web Conference, LNCS 2342, Springer-Verlag, 333-347, 2002.

[Paolucci *et al.,* 2003] M. Paolucci, N. Srinivasan, K. Sycara and T. Nishimura. Towards a semantic choreography of Web services : From WSDL to DAML-S. In proceedings of the International Conference on Web Services. IEEE, 2003.

[Pan ans Horroks, 2004] J. Z. Pan and I. Horrocks. OWL-E : Extending OWL with expressive datatype expressions. IMG Technical Report IMG/2004/KR-SW-01/v1.0, Victoria University of Manchester. Available from http ://dl-web.man.ac.uk/Doc/IMGTR-OWL-E.pdf, 2004.

[Patil *et al.,* 2004] A.A. Patil, S.A. Oundhakar, A.P. Shethand, and K. Verma, METEOR-S Web Services Annotation Framework, Proc. 13th Int'L Conf. World Wide Web (WWW), 2004.

[Rada *et al.,* 1989] R. Rada, H. Mili, E. Bichnell et M. Blettner, Development and application of a metric on semantic nets. IEEE Transaction on Systems, Man, and Cybernetics : pp 17-30. 1989.

[Resnik, 1995] P. Resnik. Using information content to evaluate semantic similarity. Proceedings of the 14th International Joint Conference on Artificial Intelligence, (1995) August 20-25 ; Montréal Québec, Canada, 1995.

[Rolland and Prakash, 2000] C. Rolland and N. Prakash. Bridging the gap between organizational needs and erp functionality, Requirements Engineering 5 (2000), no. 3, 180-193.

[Roman *et al.*, 2005] D. Roman, U. Keller, H. Lausen, J. de Bruijn, R. Lara, Mi. Stollberg, A. Polleres, C. Feier, C. Bussler and D. Fensel. Web Service Modeling Ontology, Applied Ontology, 1(1) : 77 - 106, 2005.

[Sangers *et al.,* 2013] J. Sangers, F. Frasincara, F. Hogenbooma, V. Chepegin. Semantic Web Service Discovery Using Natural Language ProcessingTechniques, in Expert Systems with Applications, Volume 40, Issue 11, 1 September, Pages 4660–4671, 2013.

[Seekda, 2009] Semantic Technology Institute, 2009. Seekda ! From : http ://seekda.com/.

[Sheth *et al.,* 2005] A. Sheth, J.A. Miller, I. Budak and K. Verma. Meteor-s : Semantic web services and processes, 2005.

[Sowa, 1995] J-F. Sowa. Top-level ontological categories. Int. J. Hum.-Comput. Stud., 43(5-6) :669–685, 1995.

[Srinivasan *et al.,* 2006] N. Srinivasan, M. Paolucci and K. Sycara. Semantic web service discovery in the OWL-S IDE. In System Sciences, HICSS'06. Proceedings of the 39th Annual Hawaii International Conference on, vol. 6, pp. 109b 109b. IEEE, 2006

[Stollberg and Kerrigan, 2007] M. Stollberg and M. Kerrigan. Goal-Based Visualization and Browsing for Semantic Web Services, in Web Information Systems Engineering – WISE 2007 Workshops, LNCS, vol. 4832, M. Weske, M.- S. Hacid, and C. Godart, Ed. Springer, p. 236–247, 2007.

[Sycara *et al.,* 2002] K. Sycara, S. Widoff, M. Klusch, and J. Lu, LARKS : dynamic matchmaking among heterogeneous software agents in cyberspace, in Proceedings of the International Conference on Autonomous Agents and Multiagent Systems, 2002.

[Timm and Gannod, 2005] J-T-E. Timm and G-C. Gannod. A Model-Driven Approach for Specifying Semantic Web Services. Proceedings of the 3rd IEEE International Conference on Web Services (ICWS 2005), 8 pages, 2005.

[Thoma, 2005] Thomas Erl. Service-oriented Architecture : Concepts, Technology, and Design. Prentice Hall,2005.

[Toma *et al.,* 2005] I. Toma, D. Fensel, M. Moran, K. Iqbal, T. Strang and D. Roman. An Evaluation of Discovery approaches in Grid and Web services Environments. In The 2nd International Conference on Grid Service Engineering and Management, Erfurt, Germany, 2005.

[Vitvar *et al.,* 2007] T. Vitvar, A. Mocan, M. Kerrigan, M. Zaremba, M. Moran, M. Cimpian, E.T. Haselwanter and D. Fensel. Semantically-enabled service oriented architecture : concepts, technology and application. Service Oriented Computing and Applications, vol. 1, no. 2, pp. 129-154, 2007.

[Wu and Palmer, 1994] Z. Wu and M. Palmer. Verb semantics and lexical selection. In proceedings of 32nd annual Meeting of the Association for Computational Linguistics,June 27-30 ; Las Cruces, New Mexico, 1994.

[Zachos *et al.,* 2008] K. Zachos, N. Maiden and R. Howells-Morris. Discovering Web Services to Improve Requirements Specifications : Does It Help ?, in REFSQ'08, LNCS, vol. 5025, B. Paech and C. Rolland, Ed. Berlin/Heidelberg : Springer, p. 168–182, 2008.

[Zhang *et al.,* 2008] M. Zhang, Z. Duan, and C. Zhao. Semi-automatically annotating data semantics to Web services using ontology Mapping. Proceedings of the 12th International Conference on Computer Supported Cooperative Work in Design, IEEE Computer Society, pp. 470-475, 2008.