الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et Populaire
وزارة التعليم العالي والبحث العلمي
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

N₀ Ref :……

Centre Universitaire
Abd elhafid boussouf Mila

Institut des sciences et de la technologie **Département de Mathématiques et Informatique**

Mémoire préparé En vue de l'obtention du diplôme de Master en:

Filière : Informatique Général

Spécialité sciences et technologies de l'information et de la communication stic

Thème

La découverte des services web composés à base de graphes

Préparé par : Boulfoul Ishak

Soutenue devant le jury :

- **Président : Boubakir Mohamed** **M.A.A C.U.Mila**
- **Examinateur : Kimouche Abdelkader** **M.A.A C.U.Mila**
- **Promoteur : Merabet Adil** **M.A.A C.U.Mila**

Année Universitaire 2014/2015

REMERCIMENTS

En premier lieu,

Je tiens à remercier Dieu le Tout Puissant, qui nous a donné La force et la patience d'accomplir ce modeste travail.

Je remercie vivement Mr ***Merabet Adil*** *mon encadreur pour sa présence, son aide et surtout pour ses précieux conseils qui m'ont beaucoup assisté pour l'accomplissement de ce projet.*

Mes vifs remerciements vont également aux membres du jury pour l'intérêt qu'ils ont porté à ce travail en acceptant de l'examiner et de l'enrichir par leurs propositions.

Je tiens à exprimer mes sincères remerciements à tout le personnel de l'institut de sciences et de la technologie du centre universitaire de Mila surtout les enseignants qui m'ont enseigné durant ces cinq années d'étude.

Je remercie également toutes les personnes qui nous ont aidés, de près ou de loin pour la réalisation de ce travail en particulier à mes parents, et mes frères, sœurs, amis qui nous avons encouragé, soutenu durant tout mon cursus.

Je remercie tous les étudiants de la promotion 2014/2015 pour avoir été liés et unis tout au long de cette année et tous ceux qui ont collaborés de près ou de loin à l'élaboration de ce travail.

Dédicace

Je dédie ce modeste travail à ceux qui m'ont toujours tenu et sacrifié leur bon moment pour que je réussisse dans ma vie mes parents.

A ma Mère CHIMOUSSA

Et Mon père RABIE

A mon frère : YASSINE et sa famille

A mes sœurs : KARIMA, AMINA, SOUAD, LAMIA, NADJLA ET AHLEM.

A mon oncle RABAH et sa famille.

A mes amies surtout : HAMZA KHERROUBI.

À Tous Mes Collègues d'étude.

Aux membres de la famille Boulfoul et Helou.

Ishak

Table des matières

Chapitre III : Modélisation conceptuelle

Chapitre IV : Implémentation

LISTE DES FIGURES

Chapitre I

Chapitre II

Chapitre III

Chapitre IV

LISTE DES ACRONYMES

XML	eXtensible Markup Language
HTTP	HyperText Transfer Protocol
URI	Uniform Resource Identifier
W3C	World Wide Web Consortium
SOAP	Simple Object Access Protocol
WSDL	Web Service Description Language
UDDI	Universal Description Discovery and Integration
HTML	HyperText Markup Language
RDF	Resource Description Framework
OWL	Ontology Web Language
SDM	Semantic Distance Mesure
OIL	Ontology Interchange Language
OWL-S	Ontology Web Language for Services

Résume

Comme toute technologie, les services web génèrent beaucoup d'intérêt et promettent de nombreux avantages. Ils fournissent des fonctionnalités accessibles via les protocoles standards du web qui permettent aux entreprises d'échanger des données à distance et une exploration facile entre les entreprises, leurs partenaires et leurs clients.

Cependant, un seul service peut ne pas suffire à satisfaire tous les besoins. La plupart du temps il est nécessaire de composer plusieurs services web. Nous parlons donc, de la composition des services web qui a connu beaucoup d'intérêt ces dernières années.

Dans ce contexte, ce mémoire est essentiellement axé sur l'étude d'un modèle de composition des services web, par utiliser un algorithme de découverte basé sur l'algorithme de chainage en arrière dans un graphe des services qui est conçu à l'aide de la distance sémantique.

MOTS-CLES : services web, composition des services, algorithme de découverte, algorithme de chainage en arrière, distance sémantique.

الملخص

مثل أي تكنولوجيا، خدمات الويب تولي اهتماما كبيرا للعديد من المزايا. بحيث توفر وظائف يمكن الوصول إليها عبر بروتوكولات الإنترنت القياسية التي تمكن الشركات من تبادل البيانات عن بعد. وذلك بتقديم تسهل إيجاد المعلومات او خدمات بين الشركات والعملاء.

ومع ذلك، لا يمكن لخدمة واحدة أن تكون كافية لتلبية جميع الاحتياجات. في معظم الأحيان من الضروري أن يؤلف عدة بين خدمات ويب. إذن نحن نتكلم، على تكوين خدمات الشبكة التي شهدت الكثير من الاهتمام في السنوات الأخيرة.

في هذا السياق، تركز هذه الأطروحة في المقام الأول على دراسة تكوين نموذج من الخدمات على شبكة الإنترنت، التي تستخدم الخوارزمية الاكتشاف المبنية على خوارزمية تسلسل بالتراجع في الرسم البياني للخدمات الذي تم تصميمه بمساعدة المسافة الدلالية.

كلمات البحث: خدمات الويب، تكوين الخدمات، خوارزمية الاكتشاف، خوارزمية التسلسل بالتراجع، المسافة الدلالي.

Introduction générale

La notion de service web désigne essentiellement une application (un programme) mise à disposition sur Internet ou sur un réseau interne d'une entreprise par un fournisseur de service, et accessible par les clients à travers des protocoles standards. Des exemples de services actuellement disponibles concernent : les prévisions météorologiques, la réservation de voyage en ligne, les services bancaires, moteur de recherche (exemple : *google*) et de la vente en ligne (*amazon* par exemple).

L'architecture des services web repose sur trois standards : WSDL qui permet de décrire un service, UDDI qui permet de référencer ce service et SOAP qui décrit la communication avec ce service. Mais pour exploiter au maximum les avantages de ce nouveau type d'applications, la communauté des services web doit se doter d'outils donnant la possibilité de pouvoir assembler différents services entre eux. L'idée de la *composition de services web* est de créer un nouveau service en combinant des services existants en se basant sur leurs comportements pour répondre aux besoins de l'utilisateur.

La problématique dans ce domaine repose sur la satisfaisante et la précision des résultats obtenus par l'algorithme de découverte. Notre solution est d'enrichir les résultats en introduisant le concept de la composition.

Le but dans ce thème est de concevoir une application permet aux clients de chercher des services web éventuellement composés. Afin de rendre le processus de découverte efficace, nous avons utilisé essentiellement la mise en correspondance de leurs entrées-sorties avec la requête demandée par le client. Une éventuelle intervention d'un ensemble d'ontologies peut être envisagée afin d'améliorer les paramètres d'une certaine connaissance.

Le processus de la *découverte* dans notre application est fondé sur le calcul de la distance sémantique et l'algorithme de chaînage en arrière dans le graphe des services. Ce dernier, est le produit de deux parseurs XML (le parseur des services web et le parseur des ontologies) et conçu en calculant la distance sémantique entre les services. Nous allons utiliser la formule de Tamer A. Farrag, Ahmed I. Saleh, H.A. Ali [29] pour calculer cette distance.

Ce mémoire est structuré de la manière suivante :

Le premier chapitre est constitué de trois parties. Nous allons décrire en première partie les définitions les plus liées aux services Web. Ensuite nous allons expliquer la notion du Web sémantique en deuxième partie. Puis dans la troisième partie, nous allons aborder la composition des services web.

Dans le deuxième chapitre, nous allons présenter en première partie le concept de l'ontologie, ensuite dans la deuxième partie nous allons expliquer la notion de la distance sémantique et la mesure de similarité sémantique basée sur l'ontologie.

Dans le troisième chapitre, nous présentons la conception de notre application, en détaillant les différents modules conçus, parmi ces modules : le module de gestion des services web, le module de gestion des ontologies, module de construction de graphe des services et l'algorithme de découverte.

Enfin, dans le quatrième chapitre, nous allons présenter l'implémentation de notre application en définissant dans une première partie les différents outils utilisés. Et dans une deuxième partie, nous allons présenter les fonctionnalités et le mécanisme de fonctionnement de l'application à l'aide des différentes interfaces disponibles.

Puis, nous allons terminer ce mémoire par une conclusion générale et des perspectives à cette étude.

Chapitre I

Généralité sur les services web

Introduction

Avec l'essor du Web qu'il y a eu dans les dernières années, il a surgi le besoin de permettre qu'une application client invoque un service d'une application serveur en utilisant l'internet. Ce besoin a été à l'origine de ce qui se connaît comme Services Web, en prenant en compte que les Services Web permettent de connecter des applications différentes.

L'objectif principal des Services Web est de faciliter l'accès aux applications entre entreprises et ainsi de simplifier les échanges de données. Ils poursuivent un vieux rêve de l'informatique distribuée où les applications pourraient interopérer à travers le réseau, indépendamment de leur plateforme et de leur langage d'implémentation. Pour ces raisons les activités de recherche et de développement autour de ce sujet ont un dynamisme très haut.

Dans ce chapitre nous allons décrire en première partie les technologies les plus liées aux services Web, ensuite nous allons expliquer la notion du Web sémantique en deuxième partie. Puis dans la troisième partie nous allons aborder la composition des services web.

Partie 1.Service Web

De plus en plus, avec l'essor d'Internet, le développement tend vers les technologies du web. Mais faire interagir des programmes différents en réseau a toujours été une affaire complexe.

Les web services sont une solution parfaite pour faciliter cette interaction car ce sont un ensemble de protocoles qui permettent de faire communiquer des programmes tournant sur des machines différentes et écrits dans des langages de programmation différents. Ils le font en reprenant certains principes du Web (transport sur HTTP, formatage en XML) mais en mettant l'accent sur la communication entre applications, pas entre humains.

Les web services servent principalement au développement d'applications distribuées et sont accessible depuis n'importe quel type de clients (WAP, Web, applicatif…).

1.1. Qu'est-ce qu'un service Web ?

Un service Web est un programme informatique permettant la communication et l'échange de données entre applications et systèmes hétérogènes dans des environnements distribués. Il s'agit donc d'un ensemble de fonctionnalités exposées sur Internet ou sur un intranet, par et pour des applications ou machines, sans intervention humaine et en temps réel.
Le consortium *W3C*[1] définit un service Web comme étant une application ou un composant logiciel qui vérifie les propriétés suivantes : [1]

- ✓ Il est identifié par un URI.
- ✓ Ses interfaces et ses liens (binding) sont décrits en XML.
- ✓ Sa définition peut être découverte par d'autres services Web.
- ✓ Il peut interagir directement avec d'autres services Web à travers le langage XML et en utilisant des protocoles Internet (http, ftp,…).

[1] World Wide Web Consortium

Exemple de service Web :

Un client demande le prix d'un article en envoyant un message sur le Web, ce message contient la référence de l'article. Le service web va recevoir la référence, effectuer le traitement du service et renvoyer le prix au client via un autre message.

1.2. Pourquoi les services Web ?

- ✓ Fournir une architecture générale pour les applications réparties sur Internet.
- ✓ Les services Web permettent d'interconnecter :
  - Différentes entreprises.
  - Différents matériels.
  - Différentes applications.
  - Différents clients.
- ✓ Distribuer et intégrer des logiques métiers.
- ✓ Les services Web sont faiblement couplés : Indépendamment du système d'exploitation et du langage de programmation utilisé. [2]

1.3. Architecture de base des services Web

L'architecture de référence des Services Web (voir figure I.1) s'articule autour des trois rôles suivants : [3]

- ✓ **Le fournisseur de service** (service provider) :
  - définit le service.
  - publie sa description dans l'annuaire.
  - réalise les opérations.
- ✓ **L'annuaire** (discovery agency) :
  - reçoit et enregistre les descriptions de services publiées par les fournisseurs.
  - reçoit et répond aux recherches de services lancées par les clients.
- ✓ **Le client** (service requestor) :
  - obtient la description du service grâce à l'annuaire.
  - utilise le service.

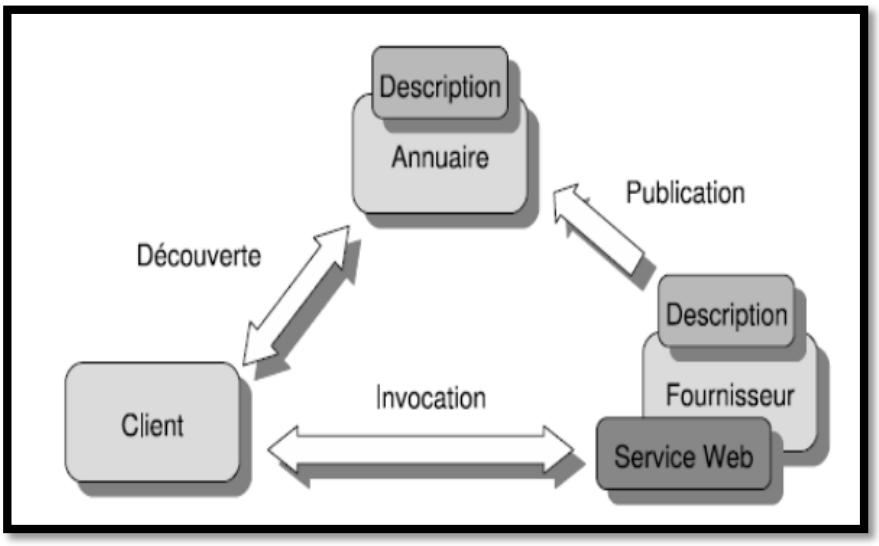

Figure I.1 : Découverte, publication et invocation des services web [3].

1.3.1. Opérations dans l'architecture des services Web

Pour qu'une application puisse tirer le plus grand avantage des services Web, les trois opérations suivantes sont primordiales : [3]

- ✓ **Publish** (*publication*) : pour qu'un service Web soit accessible par un demandeur de services, son fournisseur doit publier sa description dans un annuaire.

- ✓ **Find** (*recherche et découverte*) : le demandeur de services retrouve la description du service en interrogeant l'annuaire pour le type de service demandé.

- ✓ **Bind** (*invocation*) : dans l'opération d'invocation, le demandeur de services invoque ou établit une interaction avec le service. Cela en utilisant les détails d'invocation présents dans la description du service afin de localiser, contacter et invoquer le service.

1.3.2. Les couches technologiques des Services Web

Le fournisseur de services définit la description de son service et la publie dans un annuaire de service.

Le client utilise les facilités de recherche disponibles au niveau de l'annuaire pour retrouver et sélectionner un service donné. Il examine ensuite la description du service sélectionné pour récupérer les informations nécessaires lui permettant de se connecter au fournisseur du service et d'interagir avec l'implémentation du service considéré.

Pour garantir l'interopérabilité des trois opérations précédentes (publication, recherche et invocation), des propositions de standards ont été élaborées pour chaque type d'interactions. Nous citons, notamment les standards émergents suivants :

- ✓ SOAP : définit un protocole de transmission de messages basés sur XML.
- ✓ WSDL : introduit une grammaire commune pour la description des services.
- ✓ UDDI : fournit l'infrastructure de base pour la publication et la découverte des services.

Toutes ces technologies sont disposées en couches et constituent l'architecture Services Web comme l'illustre ce schéma :

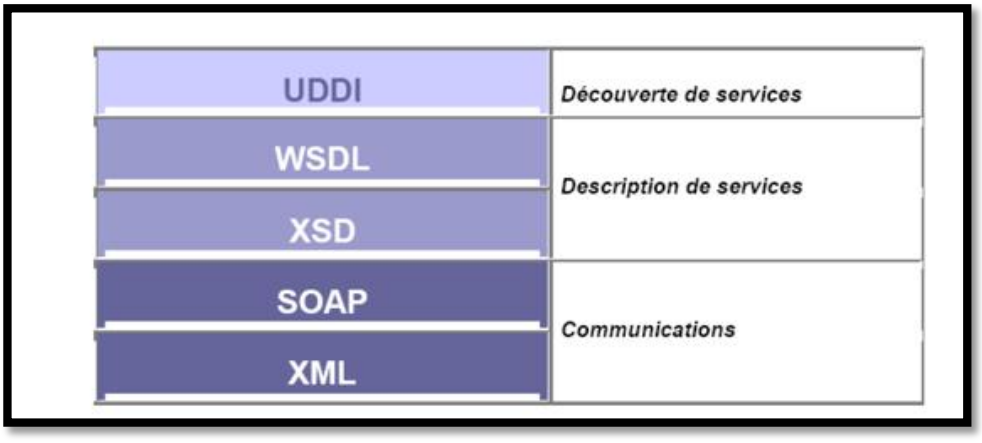

Figure I.2 : Couches technologiques des Services Web.

1.4. Le XML et les trois standards SOAP, WSDL et UDDI

1.4.1. XML

Le XML (eXtensible Markup Language) est le fondement des standards des Services Web. Ce format permet de dissocier l'information de la présentation. Il permet d'échanger des messages simples ce qui va s'avérer très pratique pour les Services Web.

Promulgué en 1998 par le W3C, on retrouve dans XML une généralisation des idées contenues dans HTML (HyperText Markup Language).

XML a été conçu pour des documents complexes, en s'appuyant sur 5 grands principes :

- ✓ Lisibilité par les machines et les utilisateurs
- ✓ Définition sans ambiguïté du contenu d'un document
- ✓ Définition sans ambiguïté de la structure d'un document
- ✓ Séparation entre documents et relations entre documents
- ✓ Séparation entre structure du document et présentation du document

Le format XML a été conçu pour permettre le déchiffrement direct par l'utilisateur comme par les programmes.

Le contenu d'un document est décrit par une succession d'éléments (blocs de texte encadrés par de paires de balises ouvrantes et fermantes) qui sont les unités de contenu. Ces éléments sont liés entre eux par une hiérarchie, certains éléments apparaissent imbriqués dans d'autres.

Voici un simple exemple pour illustrer la structure XML :

```
<?xml version=' 1.0' encoding="ISO-8859- 1"?>
    <catalogue>
        <stage id="XMLpres">
            <intitule>XML et les bases de données</intitule>
            <prerequis> connaître les langages SQL et HTML</prerequis>
        </stage>
</catalogue>
```

Figure I.3 : Exemple de structure XML.

1.4.2. Le protocole SOAP

Le protocole SOAP initialement crée par Microsoft et ensuite développé en collaboration avec IBM, lotus et Userland, basé sur le langage XML car sa syntaxe unique résoudre les conflits syntaxiques lors de l'encodage et transmission de données.

SOAP est un protocole permettant aux services Web d'échanger des informations dans des systèmes distribuées, indépendamment de toute plateforme ou langage de programmation utilisée. [4]

Le SOAP est basé sur 2 standards :

- ✓ XML pour bien structurer les messages.
- ✓ HTTP pour le transport des messages.

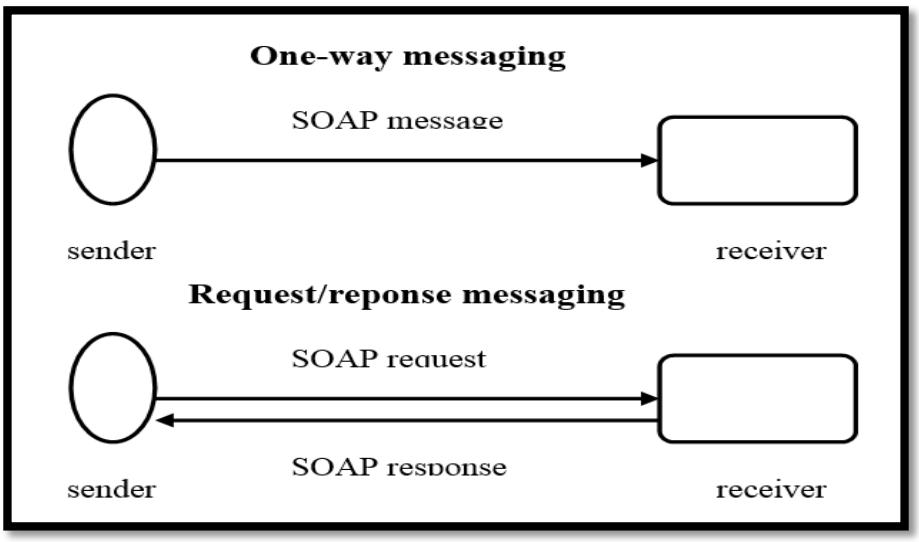

Figure I.4 : les messages SOAP. [5]

Pourquoi utiliser HTTP ?

- ✓ HTTP est le protocole de communication de l'Internet.
- ✓ HTTP est disponible sur toutes les plates-formes très rapidement.
- ✓ HTTP est un protocole simple, qui ne requière que peu de support pour fonctionner correctement.
- ✓ HTTP offre un niveau de sécurité simple et effectif.

Structure d'un Message SOAP

Un message SOAP est un document XML constitue d'un enveloppe contenant un entête (header) facultatif et le corps (body) du message : [6]

- ✓ ***Entête / Header :*** élément optionnel, contient des informations spécifiques telles que des informations sur l'authentification, le paiement, etc.
- ✓ ***Corps / Body :*** élément obligatoire, contient les entrées du message, définit la méthode appelée, valeurs des paramètres, valeur de retour, Peut contenir un élément Fault en cas d'erreur. Il contient les informations principales transmises dans un message SOAP.

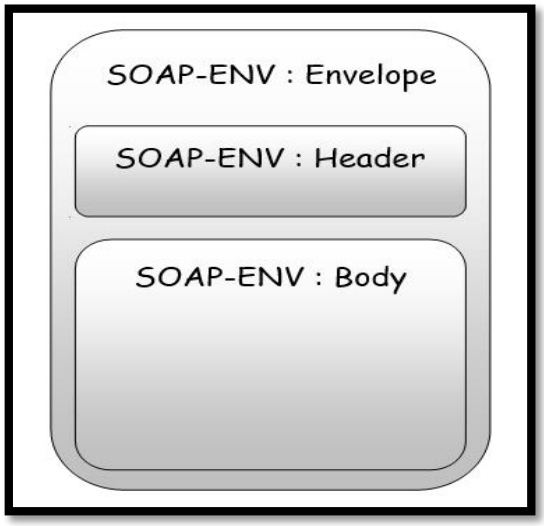

Figure I.5 : Structure d'un message SOAP [6].

Exemple d'un message SOAP Requête

```
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope  xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
    <S:Header/>
    <S:Body>
        <ns2:retirer xmlns:ns2="http://PkgBanque/">
            <NCompte>1987</NCompte>
            <Cle>1111</Cle>
            <Montant>400.0</Montant>
        </ns2:retirer>
    </S:Body>
</S:Envelope>
```

Figure I.6 : Exemple d'un message SOAP Requête.

Exemple d'un message SOAP Réponse

```
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
    <S:Body>
        <ns2:retirerResponse xmlns:ns2="http://PkgBanque/">
            <return>200.0</return>
        </ns2:retirerResponse>
    </S:Body>
</S:Envelope>
```

Figure I.7 : Exemple d'un message SOAP Réponse.

En résumé, nous pouvons retenir que le protocole SOAP permet l'invocation des services Web. Il assure la communication entre services par l'échange de messages au format XML, ce qui le rend indépendant de tout langage ou système d'exploitation.

1.4.3. Langage WSDL

WSDL est un langage de description de services Web, au format XML. Il permet de décrire de façon précise les services Web, en incluant des détails tels que les protocoles, les serveurs, les ports utilisés,…etc.

WSDL est vu comme une interface qui est le point d'entrée de n'importe quel service Web : on y trouve la localisation du service et les opérations qu'il est possible d'invoquer en utilisant SOAP. [7]

Donc une description WSDL : [7]

- ✓ Décrit le type d'un service Web (méthodes, types de données utilisées,…).
- ✓ Décrit les aspects techniques d'implantation d'un service Web (le protocole utilisé, la localisation du service (URI/URL)).

Structure du fichier WSDL

Un fichier WSDL contient une description de tout ce qui est nécessaire à l'appel d'un service Web : [3]

- ✓ **Types** : cette balise décrit les types utilisés, c.à.d. décrit les structures de données utilisées contenues dans les messages échangés entre le client et le serveur.

Message : cette balise décrit la structure d'un message échangé.

- ✓ **PortType** : cette balise décrit un ensemble d'opérations (interface d'un service Web) donc il définit les opérations offertes par le service Web.

  Operation : cette balise décrit une opération réalisée par le service Web.

- ✓ **Binding** (liaison) : cet élément définit les protocoles de communication utilisés lors des appels du service Web. Ainsi que le format des messages.

- ✓ **service** : cet élément permet de décrire les points d'accès du service Web. C'est-à-dire que la localisation du service est indiquée dans l'élément service.

  Port : cette balise décrit un port au travers duquel il est possible d'accéder à un ensemble d'opérations.

Exemple d'un fichier WSDL :

```
<definitions targetNamespace="http://PkgBanque/" name="WSBanqueService">

 <types>
        <xsd:schema>
           <xsd:import namespace="http://PkgBanque/"
            schemaLocation="http://localhost:8080/WebAppBanque/
            WSBanqueService?xsd=1"/>
        </xsd:schema>
    </types>
    <message name="Transferer">
        <part name="parameters" element="tns:Transferer"/>
    </message>
    <message name="TransfererResponse">
        <part name="parameters" element="tns:TransfererResponse"/>
    </message>
.
    <portType name="WSBanque">
         <operation name="Transferer">
            <input message="tns:Transferer"/>
            <output message="tns:TransfererResponse"/>
         </operation>
    </portType>
.
    <service name="WSBanqueService">
      <port name="WSBanquePort" binding="tns:WSBanquePortBinding">
          <soap:address
          location="http://localhost:8080/WebAppBanque/WSBanqueService"/>
      </port>
    </service>
</definitions>
```

Figure I.8 : Exemple d'un fichier WSDL.

1.4.4. L'annuaire UDDI

UDDI est un annuaire de services fondé sur XML et plus particulièrement destiné aux services Web. Un annuaire UDDI permet de localiser sur le réseau le service Web recherché. C'est un élément clé dans les spécifications de services Web, car il permet l'accès aux répertoires par les utilisateurs potentiels de services Web. [3]

Les fichiers de description WSDL ne jouent aucun rôle si un annuaire de service n'existe pas. Donc, un annuaire va stocker et localiser les descriptions des services pour permettre aux autres services et applications clientes de trouver et sélectionner le service correspond à leurs besoins. Donc UDDI a une fonctionnalité majeure qui est la recherche des services fournis par les fournisseurs dans un référentiel de services. [3]

a. Recherche de services Web avec UDDI

UDDI distingue trois types de registres, chacun d'eux peut être utilisé pour faire une recherche via UDDI : [8]

- ✓ **Les pages blanches (White paper)** : ce composant permet de connaître les informations à propos de l'organisation proposant le service. Cette description contient toutes les informations jugées pertinentes pour identifier l'organisation (son nom, son adresse physique, numéros de téléphone,…).
- ✓ **Les pages jaunes (yellow paper)** : les pages jaunes détaillent la description de l'organisation faite dans les pages blanches en répertoriant les services proposées. Donc, cette section décrit par exemple : la catégorie de l'entreprise, les services offerts par cette organisation, le type de service,…
- ✓ **Les pages vertes (green paper)** : ce composant fournit des informations plus techniques sur les services répertoriés telles que des descriptions des services en WSDL.

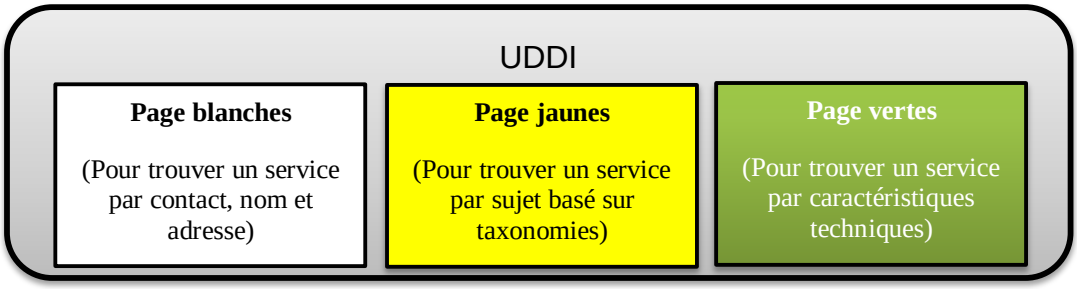

Figure I.9 : Structure de données de l'annuaire UDDI. [8]

1.5. Avantage des services Web

Parmi les avantages des services Web : [3]

- ✓ Peuvent être appelés à distance à travers un réseau.
- ✓ Les interactions sont faiblement couplées : indépendamment du système d'exploitation et du langage de programmation utilisé (Les services Web peuvent être implémentés sur différentes plates-formes et avec des langages variés).
- ✓ Les services Web fournissent l'interopérabilité entre divers logiciels fonctionnant sur diverses plates-formes.
- ✓ Les services Web utilisent des standards (wsdl, soap,…) et des protocoles (http,…).
- ✓ La composition des services : les fonctionnalités offertes par les fournisseurs des services peuvent être combinés pour offrir des services plus forts.

1.6. Inconvénients des services Web

Parmi les problèmes des services Web : [8]

- ✓ **Problèmes de sécurité** : Il est facile de contourner les mesures de sécurité mises en place par les pare-feu.
- ✓ **Problèmes de disponibilité** : Les services Web peuvent bien satisfaire un ou plusieurs besoins du client. Seront-ils pour autant toujours disponibles et utilisables ? Ça reste un défi pour les services Web.
- ✓ **Problèmes de performance** : Les services Web sont encore relativement faibles par rapport à d'autres approches de l'informatique répartie (CORBA, …).
- ✓ **Problèmes de fiabilité** : Il est difficile d'assurer la fiabilité d'un service car on ne peut garantir que ses fournisseurs, ainsi que les personnes qui l'invoquent travaillent d'une façon fiable.

Partie 2.Web Sémantique

2.1. Définition du Web sémantique

Le *Web Sémantique* est une extension du Web actuel, dans lequel l'information a un sens bien défini, et permet une meilleure coopération dans le travail entre les humains et les ordinateurs… « Les données web qui peuvent être traitées par les ordinateurs».

Le *Web sémantique* est donc une nouvelle approche pour l'organisation du contenu du *Web* instaurée dans le but d'améliorer l'interopérabilité, la *découverte* et la récupération des ressources. [9]

2.2. Définition du web services sémantiques

Les services web sémantiques sont la combinaison de deux technologies (Figure I.11) : celle des services web et celle du web sémantique. Les services web sémantiques sont des services web dont la description est améliorée par des langages empruntés au web sémantique, tel que RDF et OWL. Cet emprunt au web sémantique permet à ces services web d'être découverts et sélectionnés automatiquement par des machines ou d'autres services web distants. Ceci permet aux services web sélectionnés de répondre au mieux à la requête du client. [10]

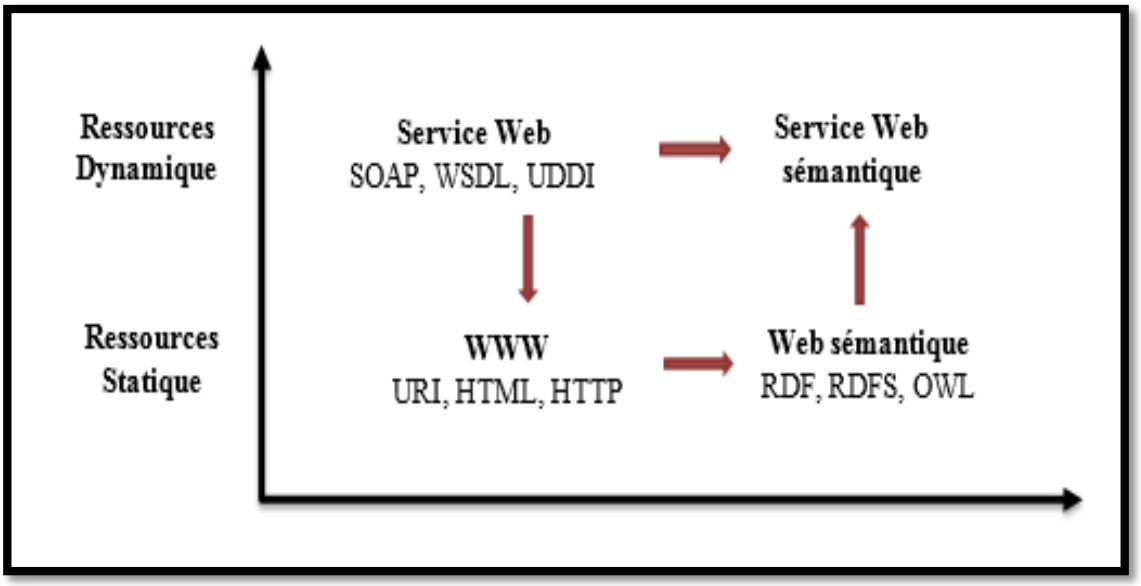

Figure I.10 : Origine des Web services sémantiques.

2.3. Approches proposées pour les services Web sémantiques [11]

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

2.3.1. WSDL-S

WSDL-S est un langage de description sémantique des services Web. Une description WSDL-S de SW est une description WSDL augmentée d'annotation sémantique. Les annotations sémantiques peuvent être des références à des concepts définis dans des ontologies externes. (Voir partie ontologie pour plus de détaille)

WSDL-S ne prescrit aucun langage particulier d'ontologies et il est défini pour être lié au langage de représentation sémantique. Il définit un modèle sémantique pour capturer les termes et les concepts utilisés pour décrire et représenter la connaissance. Cette sémantique est ajoutée en deux étapes :

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

2.3.2. OWL-S

OWL-S est une ontologie (voir chapitre 2) et un langage pour les services Web développés dans le cadre du projet DAML. OWL-S se base sur OWL qui permet de décrire des ontologies. Ainsi, OWL-S est une ontologie OWL particulière.

OWL-S décrit un service à l'aide des trois classes suivantes :

- ✓ **ServiceProfile** : Chaque instance de « Service » produit zéro ou plusieurs profils de services. Un service Profile exprime « Que fait un Service », aux fins des avertissements et

sert comme un Template pour les requêtes de services, permettant ainsi la découverte et leurs arrangements. OWL-S fournie cette classe pour décrire un SW.

- ✓ **ServiceModel** : Définit le fonctionnement du SW. Les SWs peuvent être modélisés avec OWL-S en tant que processus.
- ✓ **ServiceGrounding** : Définit les détails techniques permettant d'accéder au SW publié.

Partie 3.Service web composé

Les services Web, tels qu'ils sont conçus, peuvent également se voir employés dans le cadre d'applications distribuées. En effet, en exploitant les standards ouverts du Web et en assurant un couplage faible des composants, la technologie service Web présente une approche flexible et universelle pour l'interopérabilité de systèmes hétérogènes. Ceci a pour effet de permettre une intégration des applications plus rapide, moins couteuse et avec des perspectives prometteuses d'évolution et de réutilisation pour les entreprises. Cette opération d'intégration est appelée composition.

3.1. Définition

Selon Gardarin [12] : « La composition est une technique permettant d'assembler des services Web afin d'atteindre un objectif particulier, par l'intermédiaire de primitives de contrôle (boucle, test, traitement d'exception, etc.) et d'échange (envoi et réception de messages). Les services composants existent au préalable et peuvent ne pas être fournis par la même organisation ».

Selon S. Dustdar et W. Schreiner [13] : « L'infrastructure de base des services Web suffit pour la mise en œuvre d'interactions simples entre un client et un service Web. Si la mise en œuvre d'une application métier implique l'invocation d'autres services web, il est nécessaire donc de combiner les fonctionnalités de plusieurs services web. Dans ce cas, nous parlons d'une composition de services Web ».

La composition ou l'agrégation de services Web est une opération qui consiste à construire de nouvelles applications ou services appelés services composites ou agrégats par assemblage de services déjà existants nommés services basiques ou élémentaires.
La composition spécifie quels services doivent être invoqués, dans quel ordre et sous quelles préconditions. Les services basiques peuvent être soient des services atomiques soient des services composites.

La composition de services Web vise essentiellement quatre objectifs :

- ✓ Créer de nouvelles fonctionnalités en combinant des services déjà existants.
- ✓ Résoudre des problèmes complexes auxquels aucune solution n'a été trouvée.
- ✓ Faire collaborer plusieurs entreprises ensemble.
- ✓ Optimiser et améliorer une fonctionnalité existante.

3.2. Exemple motivant

Dans le cycle de vie de la composition, nous focalisons notre attention sur l'aspect modélisation de la synthèse. Une composition est ainsi formée d'un enchaînement de services reliés par leurs paramètres d'entrée/sortie.

A titre d'exemple, supposons qu'un utilisateur souhaite obtenir la date de publication d'un livre (figure I.12). Cet utilisateur connaît le nom de l'auteur et le titre du livre. Si sa requête ne peut être satisfaite par un service Web atomique, on peut envisager une combinaison d'autres services.

Supposons que les services suivants soient disponibles figure I.12(b) : le premier, AuthorNameBookTitle_ISBN, fournit l'ISBN d'un livre contre le nom de l'auteur et le titre du livre ; le second, ISBN_PubliDate, fournit la date de publication d'un livre contre son ISBN. Un service composite satisfaisant la requête peut être synthétisé. Il est obtenu en combinant les services AuthorNameBookTitle_ISBN et ISBN_PubliDate comme indiqué dans la figure I.12(c). La composition ainsi formée fournit l'information attendue par l'utilisateur, c'est-à-dire la date de publication du livre comme montre la figure I.12(a).

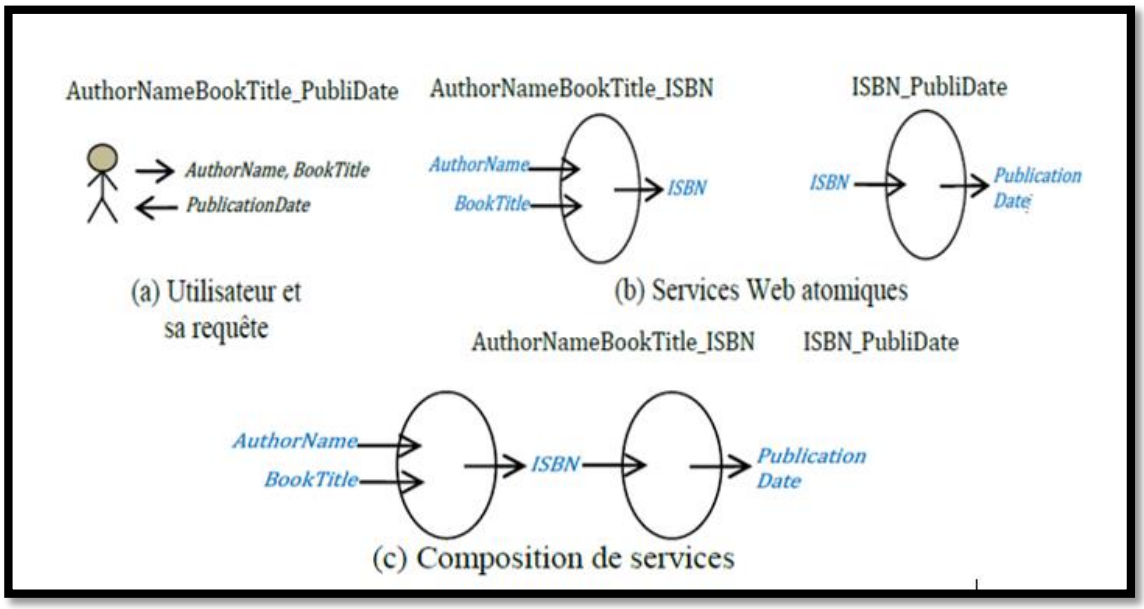

Figure I.11 : Exemple d'une interaction de services Web (c) élaborée à partir de deux services atomiques (b) et d'une requête (a).

3.3. Catégories de la composition de services

La composition de services reste un problème très complexe. Cette complexité tient au fait que les solutions de composition de services doivent tenir compte du nombre croissant de services déployés sur le web, de leur mise à jour continuelle et de leur hétérogénéité. En conséquence, elles nécessitent de traiter les problématiques de la coordination des services, leurs transactions, leur exécution et leur modèles de conversation (ou d'interaction).
Les solutions proposées peuvent être classifiées selon deux axes : [14]

Selon que la sélection des services et la gestion du flot sont faites ou non a priori, qui sont dite :

3.3.1. Une approche statique ou dynamique.

a. Composition statique (proactive)

On dit qu'une composition est statique lorsqu'elle prend place à l'étape de conception, au moment où l'architecture et la conception du système logiciel sont planifiées. Les composants (ou services) qui seront utilisés sont préalablement choisis et relies, et la gestion du flot est effectuée a priori, l'agrégation des différents services Web se fait de manière statique avant l'exécution. Les approches statiques de composition de services sont celles adoptées par

l'industrie, elles s'inspirent de la gestion de processus métiers quant à la description des données et des flots de contrôle pour le processus de composition.

Donc la composition statique des services web est considérée trop restrictive et les composants doivent s'adapter automatiquement aux changements imprévisibles (inattendus). Pour cela des nouveaux types de compositions ont été considérés.

Exemple :

Nous citons l'exemple de réservation de billets d'avions en ligne. Un service de réservation en ligne propose essentiellement les trois fonctions suivantes :

- ✓ Le contrôle de la validité du billet.
- ✓ Le contrôle de la carte de crédit.
- ✓ La mise à jour de la base de données appropriée.

Ce service composite est fréquemment utilisé. C'est pourquoi il serait plus approprié d'intégrer ces trois services ensembles pour répondre au nombre important de requêtes plutôt que d'essayer de créer un service à chaque fois qu'un client en exprime une requête. [3]

b. Composition dynamique (réactive)

Par opposition, une composition de services est dite dynamique si les services sont sélectionnés et composes à la volée en fonction des besoins formulés par l'utilisateur.

Une approche dynamique pour la composition de services offre le potentiel de réalise des applications flexibles et adaptables en sélectionnant et en combinant les services de manière appropriée sur la base de la requête et du contexte de l'utilisateur. Ce type de composition peut engendrer de nombreuses applications utiles qui n'ont pas été prévues à l'étape de conception. Par conséquent, la composition dynamique de services est propice dans un environnement tel que le web ou les composants disponibles sont dynamiques et les attentes des utilisateurs variables et personnalisées. C'est précisément ce type d'approche qui fait l'objet de notre travail.

Exemple :

Un manager de voyages d'affaires (en charge de planifier un voyage pour un client) doit collaborer avec: le manager de tickets, le manager d'itinéraires, le manager d'hôtels et le manager de voitures de location pour organiser le voyage.

Dans ce cas les interactions ne peuvent pas être prédéfinies au moment de la conception du système. [3]

En fonction du degré de participation de l'utilisateur dans la définition du schéma de composition, qui sont :

3.3.2. Ces propositions : manuelles, semi-automatiques ou automatiques.

a. Composition manuelle

La composition manuelle des services web suppose que l'utilisateur génère la composition à la main via un éditeur de texte et sans l'aide d'outils dédiés. Elle nécessite des connaissances de bas niveau de programmation.

b. Semi-automatique

Dans les approches semi-automatiques, des outils sont développées afin de proposer à l'utilisateur des suggestions sémantiques quant à la sélection des services à composer (ex. en fonction de leurs entrées/sorties, annotations, etc.) durant le processus de composition.
Les techniques de composition semi-automatiques sont un pas en avant en comparaison avec la composition manuelle, dans le sens qu'ils font des suggestions sémantiques pour aider à la sélection des services Web dans le processus de composition. [15]

c. Automatique

Cette approche de composition consiste de concevoir un compositeur qui est en sorte d'un agent logiciel chargé de sélectionner, composer et interopérer les services impliqués dans le processus de composition. Donc la composition automatique est une composition totalement automatisée qui prend en charge tout le processus de composition et le réalise automatiquement, sans aucune intervention de l'utilisateur.

3.4. Techniques de composition des services web

On peut classer ces techniques en deux grandes familles : les techniques de composition dites « statiques », qui sont définies à l'aide de processus métiers (orchestration et chorégraphie) ; et techniques de composition dites « dynamiques », lorsque la composition de services web tient compte des services disponibles, de leurs fonctionnalités et du but à atteindre que ce soit avant ou pendant l'exécution des services web.

3.4.1. Les différentes approches de la composition statique

La composition de services web peut se faire de deux manières : orchestration et chorégraphie.

a. Orchestration

L'orchestration de services web permet de définir l'enchaînement des services selon un canevas prédéfini et de les exécuter à travers des "scripts d'orchestration". Ces scripts décrivent les interactions entre les services en identifiant les messages échangés, les branchements logiques et les séquences d'invocation. Le composant exécutant les scripts d'orchestration est appelé moteur d'orchestration. Celui-ci agit comme une entité centralisée pour coordonner les interactions entre les services. [16], [17] et [18]

Parmi les langages d'orchestration : BPEL, WSFL, XLANG.

La figure suivante montre le workflow dans l'orchestration des services web. Le coordinateur prend le contrôle de tous les services web impliqués et coordonne l'exécution des différentes opérations des services web qui participent dans le processus.

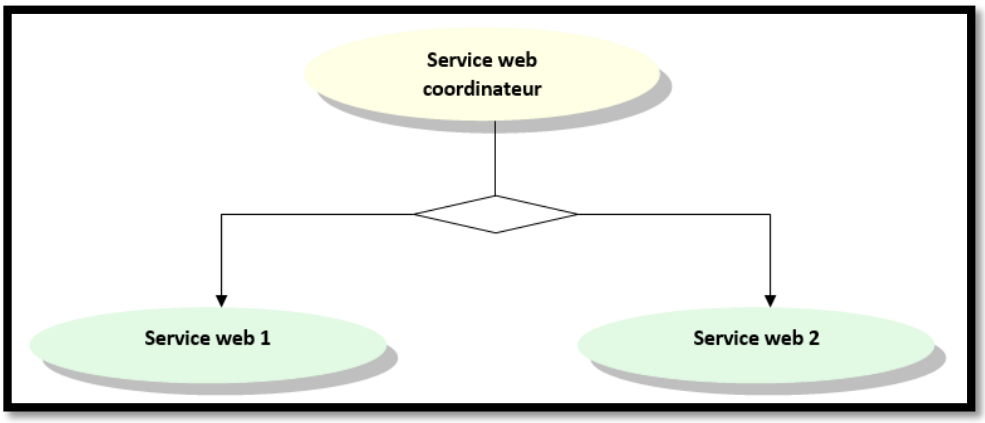

Figure I.12 : Schéma de l'orchestration des services web [19]

En résumé, l'orchestration est un processus centralisé qui contrôle et qui coordonne l'ordre et l'exécution des interactions des services web impliqués dans la composition. L'orchestration s'intéresse ainsi au côté interne (implémentation) d'un service web composite.

b. Chorégraphie

Contrairement à l'orchestration, la chorégraphie permet aux services d'interagir entre eux et avec le client directement, sans aucun contrôle centralisé. Ils sont capables d'exécuter un service composite sans passer par un coordinateur central. Chaque service intervenant dans la composition connaît exactement quand ses opérations doivent être exécutées et avec qui l'interaction doit avoir lieu. [20] et [16].

La collaboration dans la chorégraphie des services web peut être représentée de la manière suivante :

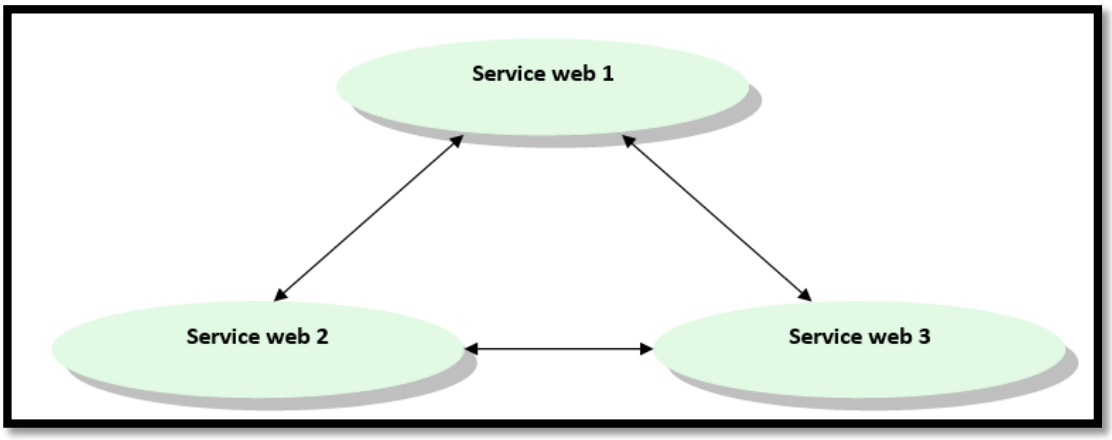

Figure I.13 : Schéma de la chorégraphie des services web [19]

La composition par chorégraphie décrit d'une part les interactions entre les services et d'autre part les relations qui existent entre ces interactions. Les opérations internes des participants ne sont pas considérées. [21]

Parmi les langages de chorégraphies : WSCI, WS-CDL

Depuis la perspective de la composition des services web, l'orchestration est un rapprochement plus flexible que la chorégraphie :

- ✓ Le responsable ou coordinateur de tout le processus métier est connu.
- ✓ Les services web peuvent être incorporés sans soucis, parce qu'ils n'ont pas conscience d'appartenir à un processus métier.

En résumé, l'orchestration et la chorégraphie sont toutes deux des manières de composition de service web agrégés. La différence principale entre ces deux manières, est que dans la chorégraphie, chaque service connaît les autres services qui interagissent alors que dans le cas de l'orchestration, le service qui joue le rôle de chef d'orchestre est le seul qui connaît les services en interaction.

3.4.2. Les différentes approches de la composition dynamique

Les approches proposées dans la littérature sont diverses. Aucune n'est pas reconnue comme une norme. La classification de ces approches permet de dégager trois grandes familles. [22]

a. Approche sémantique

L'approche sémantique de composition des services web est basée sur l'utilisation des ontologies, pour la description sémantique de tous les concepts des domaines partagés entre les services web. Pour explorer cette approche, la communauté académique a proposé des langages formels qui permettent de définir ce qu'un service web fait. Les deux principales initiatives sont DAML-S et OWL-S.

b. Approche formelle

L'approche formelle est un domaine d'étude qui fournit un langage pour décrire la spécification, la conception et le code source d'un logiciel. Ces méthodes fournissent des mécanismes de raisonnement pour vérifier l'interaction entre service web, et la solidité du service web composite.

C'est représenter les services web en utilisant des systèmes de transition qui communiquent en échangeant des messages. Cette approche se base sur les techniques de contrôle du modèle symbolique pour déterminer une composition parallèle de tous les services disponibles, et alors générer un contrôleur qui contrôle les services composés afin qu'ils satisfassent les besoins spécifiés par l'utilisateur.

c. Approche industrielle

BPEL4WS (Business Process Execution Language for Web Services) offre un langage pour les spécifications formelles des processus d'affaires et leurs interactions (BPEL4WS).
BPEL4WS supporte la modélisation de deux types de processus : les processus exécutables et ceux abstraits. Un processus abstrait est un protocole d'affaires qui spécifie le comportement des échanges de messages entre plusieurs composantes, sans pour autant révéler leurs comportements internes. Un processus exécutable spécifie l'ordre d'exécution entre les activités (activities) constituant le processus, les partenaires (partners) du processus, les messages échangés entre ces partenaires, et indique le comportement en cas d'erreur ou d'exception. [23]

Conclusion

Supporté par des standards, les services web sémantiques combinent URI, HTTP et RDF (OWL) pour bâtir un système de communication de machine à machine dans le but de partager l'information.

Les avantages de l'utilisation du Web sémantique par la description des services Web sont nombreux, en plus de rendre l'interface du service web accessible automatiquement par des machines, ils permettent également la description de propriétés non fonctionnelles telles que la qualité de services et les contraintes de sécurité, d'une manière uniforme et compréhensible par tous. Concrètement, ce qui nous s'intéresse c'est l'automatisation, autant que possible, des divers aspects relatifs aux services Web, en particulier la sélection et la composition des services Web sémantiques.

La composition des services web est un point crucial qui a un grand impact sur plusieurs domaines de recherches. Beaucoup d'efforts ont été fourni afin de permettre une composition utilisable et acceptable de services Web. Ces efforts ont été concrétisés par plusieurs standards et approches de compositions qui varient de celles qui aspirent à devenir des normes de l'industrie à celles qui sont beaucoup plus abstraites.

L'objectif de la composition automatique de services Web est de gagner du temps et faciliter la tâche d'un programmeur, afin de découvrir ou de créer un nouveau service Web complexe.

Chapitre II

Ontologie et distance sémantique

Introduction

Nées des besoins de représentation des connaissances, les ontologies sont à l'heure actuelle au cœur des travaux menés en Ingénierie des Connaissances (IC). Le terme « ontologie » est utilisé depuis le début des années 1990, et son champ d'application s'élargit considérablement.

Un des plus grands projets basés sur l'utilisation d'ontologies consiste à ajouter au Web une véritable couche de connaissances permettant des recherches d'informations au niveau sémantique. A terme, il est prévu que des applications internet pourront mener des raisonnements utilisant les connaissances stockées sur la Toile.

Au fur et à mesure des expérimentations, des méthodologies de construction d'ontologies et des outils de développement adéquats sont apparus. L'enjeu de l'effort engagé est de rendre les machines suffisamment sophistiquées pour qu'elles puissent intégrer le sens des informations.

Dans ce chapitre, nous allons présenter en première partie le concept de l'ontologie, ensuite nous allons expliquer dans la deuxième partie la notion de distance sémantique et la mesure de similarité sémantique basée sur l'ontologie.

Partie 1. Ontologie

Dans cette partie, nous allons relever une définition générale qui a été attribuée à la notion d'ontologie. Nous allons montrer aussi le but de l'utilisation des ontologies dans le domaine du Web service ainsi que leur place dans les systèmes à bases de connaissances. Ensuite, les outils utilisés pour servir leur représentation et développement, à savoir les langages de spécification, les moteurs d'inférences, les langages d'interrogation, et les éditeurs d'ontologie.

1.1.Définition

Le terme « ontologie » est employé dans des contextes très différents touchant la philosophie, la linguistique ou l'Intelligence Artificielle. De nombreuses définitions ont été offertes pour donner un éclaircissement sur ce terme, mais aucune de ces définitions ne s'est explicitement imposée.

En générale l'ontologie est un élément-clé de la résolution du problème de l'hétérogénéité sémantique, permettant ainsi l'interopérabilité sémantique entre les différentes applications Web et des services.

1.2.Pourquoi les ontologies ?

Une ontologie définit un vocabulaire commun pour les chercheurs qui ont besoin de partager l'information dans un domaine. Elle inclut des définitions lisibles en machine des concepts de base de ce domaine et de leurs relations.

Pour quelles raisons développer une ontologie ? En voici quelques-unes :

- ✓ Partager la compréhension commune de la structure de l'information entre les personnes ou les fabricants de logiciels.

Par exemple :

On a un certain nombre de sites Web contiennent de l'information médicale ou fournissent des services de e-commerce en médecine. Si ces sites partagent et publient tous la même ontologie, qui sont à la base des termes qu'ils utilisent, alors les agents informatiques peuvent extraire et agréger l'information de ces différents sites. Les agents peuvent utiliser cette information agrégée pour pouvoir répondre aux interrogations des utilisateurs ou comme données d'entrées pour d'autres applications.

L'ontologie :

- ✓ Permettre la réutilisation du savoir sur un domaine
- ✓ Expliciter ce qui est considéré comme implicite sur un domaine
- ✓ Distinguer le savoir sur un domaine du savoir opérationnel
- ✓ Analyser le savoir sur un domaine

Autrement dit, une ontologie est un modèle d'organisation des connaissances dans un domaine donné. On trouvera dans l'ontologie les classes d'objets à organiser (personnes, étudiant, professeur, thèse...), les types d'attributs pouvant être attachés aux objets (référence, description, adresse, nom...) et les types de relations entre les objets (un objet "étudiant " peut être relié par une relation "supervisé par" à un objet de type "professeur"), etc...

1.3.Notion de base

Une ontologie contient les primitives terminologiques du domaine (le vocabulaire conceptuel) ainsi que des axiomes qui restreignent l'interprétation des primitives. Le vocabulaire conceptuel est structuré en un ensemble de concepts et un ensemble de relations existantes entre ces concepts.

1.3.1. Les concepts

Les concepts sont appelés aussi termes ou classes de l'ontologie, correspondent aux abstractions pertinentes d'un segment de la réalité (le domaine du problème), retenues en fonction des objectifs qu'on se donne et de l'application envisagée pour l'ontologie.
Ces concepts peuvent être classifiés selon plusieurs dimensions :

- ✓ Niveau d'abstraction (concret ou abstrait) ;
- ✓ Atomicité (élémentaire ou composée) ;
- ✓ Niveau de réalité (réel ou fictif).

Un concept est caractérisé par une extension qui est l'ensemble des objets (appelés instances du concept) manipulés à travers ce concept, et une intension qui est l'ensemble des propriétés spécifiant la sémantique du concept.

1.3.2. Les rôles

Les rôles traduisent les associations (pertinentes) existantes entre les concepts présents dans le segment analysé de la réalité. Ces relations incluent les associations suivantes :

1) Sousclasse-de (généralisation – spécialisation) ;
2) Partie-de (agrégation ou composition) ;
3) Associée-à ;
4) Instance-de, etc.

Ces relations nous permettent d'apercevoir la structuration et l'interrelation des concepts, les uns par rapport aux autres.

1.3.3. Les axiomes

Les axiomes constituent des assertions, acceptées comme vraies, à propos des abstractions du domaine traduites par l'ontologie.

Les axiomes sont employés pour modéliser les phrases qui sont toujours vraies. Ils peuvent être inclus dans une ontologie pour plusieurs buts, tels que définir la signification de composants d'ontologie, définir les contraints complexes sur les valeurs des attributs, les arguments de relations, vérifier l'exactitude d'informations indiquées dans l'ontologie ou déduire une nouvelle information, etc ...

1.4.Langages de spécification d'ontologies

Plusieurs langages de spécification d'ontologies ont été développés pendant ces dernières années. Parmi ces langages, nous citons :

1.4.1. OIL (Ontology Interchange Language)

Dans l'optique d'une utilisation d'ontologies sur le Web, le langage RDF a été enrichi par l'apport du langage OIL (Ontology Interchange Language) qui permet d'exprimer une sémantique à travers le modèle des frames tout en utilisant la syntaxe de RDF.

1.4.2. OWL (Ontology Web Language)

OWL est considéré par W3C comme un langage d'ontologie standard (Figure II.1). Il a non seulement la capacité de décrire les concepts dans un domaine mais aussi d'un ensemble plus riche d'opérateurs, donc ses concepts sont bien définis et bien décrits.

On peut construire des concepts complexes en basant sur les définitions des concepts plus simples. En outre, on peut vérifier si tous les relations et les définitions dans l'ontologie sont conformés, et identifier quels concepts s'adaptent sous quelles définitions. Donc, on peut maintenir la hiérarchie correctement entre les classes.

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns="http://127.0.0.1/ontology/finance_th_web.owl#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:dc="http://purl.org/dc/elements/1.1/"
    xmlns:p1="http://127.0.0.1/ontology/"
    xmlns:daml="http://www.daml.org/2001/03/daml+oil#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xml:base="http://127.0.0.1/ontology/finance_th_web.owl">
  <owl:Ontology rdf:about=""/>
  <owl:Class rdf:ID="dealer">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="financial_agent"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="compound_interest_bond">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="bond"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="mortgage_backed_security">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="security"/>
    </rdfs:subClassOf>
```

Figure II. 1 : exemple d'un OWL.

Et il y a d'autres outils, par exemple : KIF[2], RDF(s)[3] … etc.

1.5. Ontologie et Web service

L'objectif premier d'une ontologie est de modéliser un ensemble de connaissances dans un domaine donné, qui peut être réel ou imaginaire.

Les ontologies sont employées dans le Web sémantique, le génie logiciel, l'informatique biomédicale ou encore l'architecture de l'information comme une forme de représentation de la connaissance au sujet d'un monde ou d'une certaine partie de ce monde.

Les travaux menés autour de la description des services Web utilisent de plus en plus les ontologies pour fournir une représentation sémantique de l'information, à la fois, détaillée, riche et facile à manipuler par les machines. Afin de profiter les avantages des ontologies dans le but d'améliorer la découverte des services Web.

[2]Knowledge Interchange Format
[3]Resource Description Framework and RDF schema

Partie 2. La distance sémantique

La sémantique c'est le sens, c'est ce qu'on appelle aussi parfois le « fond » par opposition à la forme. Par exemple si nous comparons les mots « voiture » et « véhicule » d'un point de vue sémantique ce sont des mots relativement proches par leur sens. Nous pouvons en général remplacer le mot voiture par le mot véhicule, cela ne changera pas le sens du discourt. La sémantique est naturelle et facile pour un humain, par contre un ordinateur est incapable de voir la proximité de sens entre ces deux mots.

2.1.Définition

La distance sémantique est une valeur attribuée à deux concepts qui dépend de la similarité entre ces derniers. Cette distance sémantique est calculée selon plusieurs moyens comme nous allons voir dans cette partie.

Notez que cela ne veut pas dire que l'ordinateur va comprendre réellement le sens du texte, mais il sera capable de les comparer et de mesurer leurs similitudes ou différences sémantique.

2.2.Mesure de similarité sémantique basée sur l'ontologie

Cette partie passe en revue les différents paradigmes utilisés pour calculer les mesures de similarité sémantique à partir d'une ontologie (SSMs : semantic similarity measures).
SSMs visent à estimer la ressemblance entre deux concepts en se basant sur les connaissances taxonomiques modélisées dans les ontologies.

2.2.1. Des approches à base d'arêtes

Les mesures de la similitude estimées entre deux concepts par ces approches seront selon la force de leur interconnexion dans l'ontologie. L'approche la plus courante considère la similitude comme une fonction qui calcule la distance qui sépare les deux concepts dans l'ontologie.

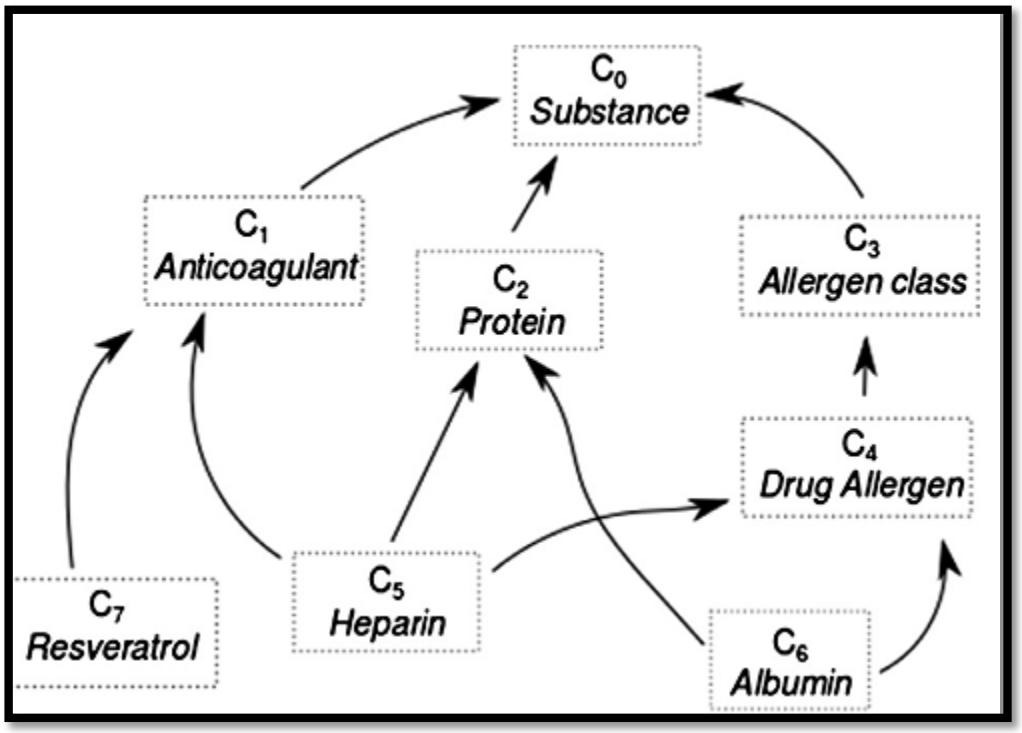

Figure II. 2 : Une Capture d'un ensemble des concepts définis dans l'ontologie ***SNOMED-CT***.

- Par exemple, Rada et al [24] estime la distance entre de deux concepts u, v comme le plus court chemin qui les relie (sp (u, v)) :(short path)

$$\mathbf{Dist}_{rada}\ (u, v) = sp\ (u, v)$$

Dans la figure II.2, le plus court chemin entre les deux concepts C_5 et C_3 est $C_5 \rightarrow C_4 \rightarrow C_3$.

- *Leacock et* Chodorow ont proposés une adaptation non-linéaire de la distance de Rada pour définir la mesure de similarité Sim_{LC} [27] :

$$\mathbf{Sim}_{LC(u,v)=} -\mathbf{log}\left(\frac{sp\ (u,v)}{2.MAX_{depth}}\right)$$

- ✓ La distance de *Rada* est normalisée par la profondeur maximale de l'ontologie, Max_depth, c'est-à-dire le plus long des chemins les plus courts reliant un concept au concept qui englobe tous les autres.

La racine de l'ontologie est C_0 dans la Figure II.2.

Certaines approches raffinent un peu plus en proposant de :

- ✓ Prendre en considération les variations du poids des liens entre les concepts.
- ✓ Et aussi plus les deux concepts sont profonds, plus leur relation sémantique est considérable.

Dans la plupart des cas, la similarité sémantique entre deux concepts est estimée en fonction de la profondeur de l'ancêtre commun le plus proche (*LCA* : *Least Common Ancestor*), c'est-à-dire le nœud (ancêtre) commun entre les deux concepts.

Dans la Figure II.2,

- ✓ Le *LCA* des concepts C_5 et C_3 est C_0.
- ✓ Le *LCA* des concepts C_5 et C_7 est C_1.
- ✓ Le *LCA* des concepts C_5 et C_6 est C_2.

Remarque :

Plus le LCA est profond ➡ plus la spécification entre les concepts est considérable ➡ Donc, plus de similarité entre les concepts comparés.

2.2.2. Les Approches à base de nœuds

Les approches à base de nœuds se concentrent sur l'évaluation des concepts définis dans l'ontologie.

Deux stratégies spécifiques peuvent être distinguées :

a- Stratégies à base de caractéristiques

Ces stratégies évaluent le concept comme un ensemble de caractéristiques. Ils se situent dans le modèle-à-caractéristiques proposé par Tversky [26].

Pour les mesures à base d'ontologies, les caractéristiques d'un concept sont généralement considérées comme l'ensemble des concepts qui l'englobent, c'est-à-dire, l'ancêtre $A(u)= \{v|u \leq v\}$.

Autrement dit, un concept se caractérise par les sémantiques qui sont hérités de ses ancêtres. À la (figure II.2), le concept C_6 sera donc représenté par l'ensemble des caractéristiques :

$A(C_6) = \{C_0, C_2, C_3, C_4, C_6\}$.

b- Stratégies basées sur la théorie de l'information

Une approche basée sur la théorie de l'information c'est une approche qui évalue la similitude des concepts selon la quantité d'informations qu'ils fournissent, c'est-à-dire leurs contenu d'information (IC : Information Content).

2.2.3. Les approches Hybrides

Les approches hybrides combinent les notions des approches basées sur les arrêtes et les notions des approches basées sur les nœuds. Ils sont généralement définis comme des agrégations pondérées des ancêtres, des degrés de nœuds et des caractéristiques de concepts. [27].

Conclusion

Dans la premier partie, l'ontologie est présentée comme un outil essentiel pour représenter le monde réel par la modélisation des concepts de ce monde dans un modèle représentatif décrit par les classes y existantes, et les propriétés qui définissent les relations établies entre ces concepts.

Le standard OWL s'est montré capable de jouer un rôle du pivot, grâce à sa compatibilité avec d'autres langages DAML+OIL, RDF, et RDFS.

Ensuite dans la deuxième partie, nous avons discuté sur la distance sémantique et les différents paradigmes utilisés pour calculer les mesures de similarité sémantique.
Il y a eu plusieurs approches pour calculer la distance sémantique. Dans cette partie, nous avons parlé sur les approches à base d'arêtes, les approches à base de nœuds et en fin, nous avons parlé sur les approches Hybrides.

Chapitre III

modélisation conceptuelle

Introduction

Dans ce chapitre, nous allons relever un algorithme de découverte des services web sémantique basé sur une composition entre les services. Nous allons commencer par deux programmes : le parseur OWLS (service web) et le parseur OWL (ontologie). Après la mise en œuvre des deux parseurs, les résultats sont : un ensemble de graphes d'ontologies et un ensemble de services web chacun présenté par le nom de service, la description de service, une liste des entrées et une liste des sorties.

Ensuite, un graphe rassemble tous les services de la base de données où chaque sommet représente un service web. Deux services S1, S2 sont reliés par une arrête si les sorties de S1 ressemblent aux entrées de S2. La ressemblance est évalué par le calcul de la distance sémantique entre les deux liste (les entrées de S2 et les sorties de S1)

Enfin, l'algorithme de découverte va chercher dans le graphe le service qui répond aux besoins du client. Une éventuelle composition peut être invoquée pour répondre à certaines requêtes.

1. Présentation de l'architecture

Afin d'assurer une modélisation cohérente, nous avons commencé par la création d'un certain nombre de modules où chacun d'eux assure des fonctionnalités distinctes, le regroupement de ces modules nous a permis par la suite de construire notre système de découverte.

Ces modules sont en coopération et en collaboration entre eux où chaque module assure ses tâches et fournit ses sorties comme des entrées à un autre module.

Là (figure III.1) présente les cinq modules composants de notre architecture ainsi que la liaison entre eux. C'est l'architecture du système que nous proposons.

Le Premier module prend en charge de l'extraction de l'information à partir d'un document des services web (OWLS), le deuxième module aussi prend en charge l'extraction de l'information mais à partir des fichiers des ontologies (OWL).

Le troisième module s'intéresse essentiellement à construire un graphe des services en utilisant les diffèrent concepts des ontologies et des distances sémantiques. Ensuite, le quatrième module

est l'algorithme de découverte des services qui est basé sur l'algorithme de chaînage en arrière. Et finalement, le cinquième module assure la gestion de l'affichage et la présentation des résultats.

Nous proposons par la suite l'architecture générale pour expliquer clairement le déroulement de notre système de découverte, à partir de l'entrée de la requête de l'utilisateur jusqu'à la récupération des services web pour lui répondre à certaines de ses besoins.

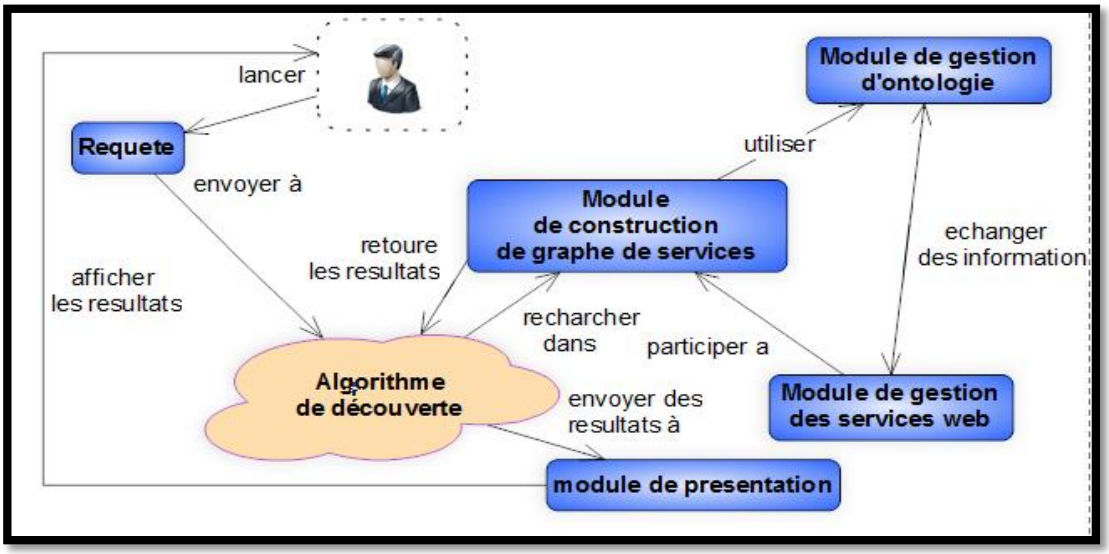

Figure III. 1 : L'architecture du système.

2. Module de gestion des services web

Nous traitons le contenu du document OWLS (figure III.2) pour extraire le vecteur des mots significatifs pour chaque Service Web. La construction de ce vecteur de données est composée à partir des trois étapes suivantes :

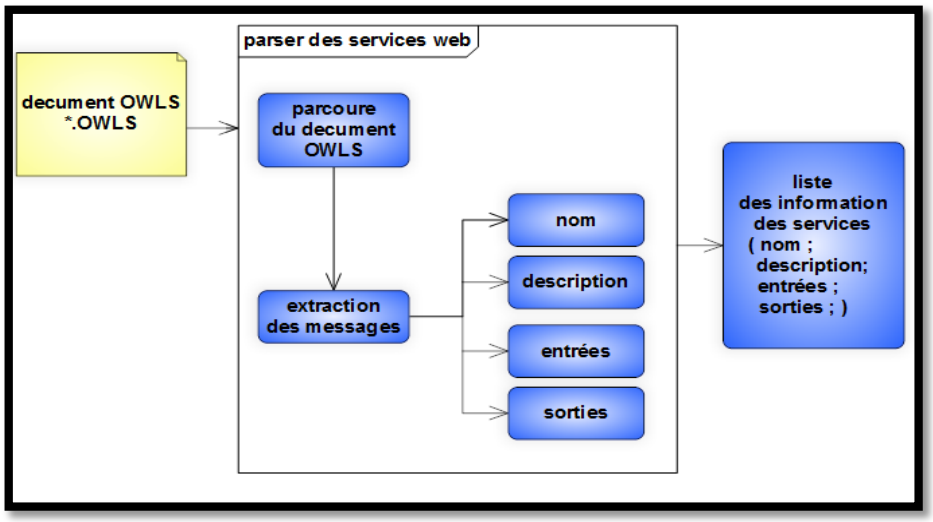

Figure III. 2 : Module de gestion des services web.

2.1.Parcoure du document OWLS

C'est l'extraction des informations utiles : cette étape consiste, grâce à des règles, à identifier et extraire les termes pertinents d'un fichier XML où se trouve la définition du service web.

2.2.Extraction des informations d'un document OWLS

Le nom et la description du service, les messages d'entrées et les messages de sorties sont les caractéristiques sélectionnées.

a. Nom : Le nom de service se trouve comme montré dans la (figure III.3) dans l'élément principal : *<profile:serviceNamexml:lang="en">*

Exemple : le nom de service *MyDESTINATIONService* comme la figure III.3 :

```
<profile:serviceName xml:lang="en">
MyDESTINATIONService
</profile:serviceName>
```

Figure III. 3 : le nom de service MyDESTINATIONService.

b. Description : Cet élément dans les documents OWLS décrit la description de service.

Exemple : la description d'un service *MyDESTINATIONService* comme montre la figure III.4 :

```
<profile:textDescription xml:lang="en">
It returns DESTINATION of my office.
</profile:textDescription>
```

Figure III. 4 : description de service MyDESTINATIONService.

***c. Entrées* :** Cet élément dans les documents OWLS décrit les entrées requis par les services.

Exemple : les entrées d'un service*MyDESTINATIONService* comme montre la figue III.5 :

```
<profile:hasInput  rdf:resource="#_ORGANIZATION"/>
<profile:hasInput  rdf:resource="#_SURFING"/>
```

Figure III. 5 : les entrées de service MyDESTINATIONService.

d. sorties : Cet élément dans les documents OWLS décrit les sorties produits par les services.

Exemple : les sorties d'un service*MyDESTINATIONService* comme montre la figue III.6 :

```
<profile:hasOutput rdf:resource="#_DESTINATION"/>
```

Figure III. 6 : la sortie de service MyDESTINATIONService.

Donc, ce service peut être représenté comme suite :

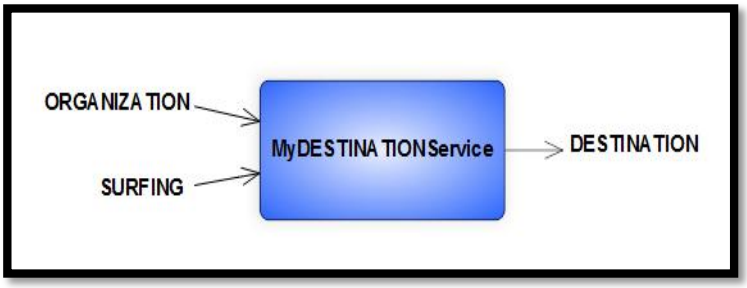

Figure III. 7 : présentation du service *MyDESTINATIONService.*

2.3. Algorithme de « parseur OWLS »

```
Algorithme : parseur de document OWLS
Entrée :
     Liste des Fichiers OWLS
Sortie :
     Liste des informations de chaque fichier (nom_service,
     discription_service, entrées_service, sorties_service)
Terminologie :
     Fichier : les fichiers OWLS
Debut
     Tant que ( ! List_fichiers_OWLS.isEmpty) Faire
          Service service = new Service () ;
          Service = parser_OWLS (fichier_OWLS (fichier_i)) ;
          List_information.add (service) ; i++ ;
     Fin Tant que ;
Fin.
```

Parser_OWLS (fichier) : fait la lecture du fichier OWLS (un fichier XML), ensuite l'extraction des informations nécessaires grâce à l'architecture XML qui est basé sur les balises.

3. Module de gestion des ontologies

Dans notre système l'ontologie a été utilisée pour la projection sémantique de la requête et les services web (les entrées et les sorties) afin d'extraire les concepts relatifs à chaque mot (figure III.8).

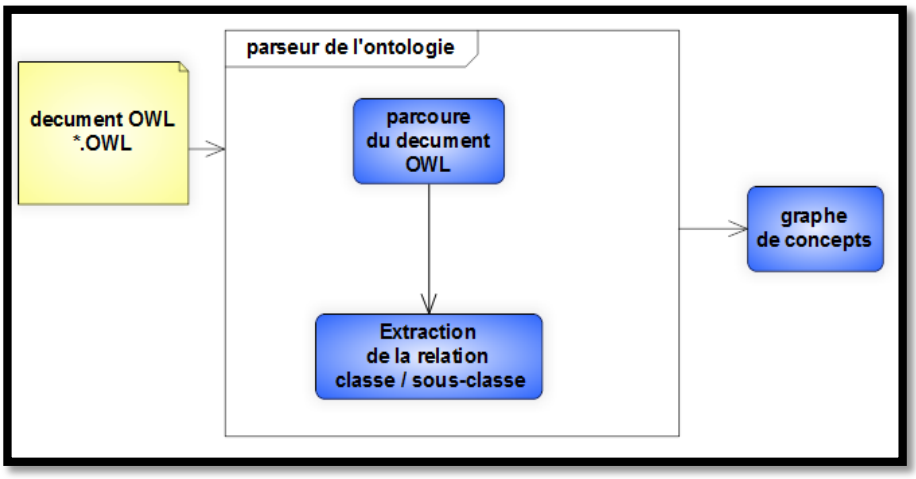

Figure III. 8 : Module de gestion des ontologies.

3.1.Parcoure du document OWL

L'objectif de cette étape est d'extraire les informations utiles dans ces fichiers. Elle consiste, grâce à des règles, à identifier et extraire les termes pertinents à partir des descriptions textuelles disponibles dans un fichier source.

3.2. L'extraction de la relation Classe et Sous-Classe

Les mots valides sont le concept père (classe) et le concept fils (sous-classe).Chaque concept fils a un père et il n'y a aucun concept isolé.

Exemple : le concept *EuropeTaxedPrice*est une sous-classe du concept *TaxedPrice* (figure III.9)

```
<owl:Class rdf:about="http://127.0.0.1/ontology/concept.owl#EuropeTaxedPrice">
<rdfs:subClassOf>
<owl:Class rdf:about="http://127.0.0.1/ontology/concept.owl#TaxedPrice">
</owl:Class>
```

Figure III. 9 : Exemple de classe et sous-classe.

3.3.Graphe de concepts (l'ontologie)

L'idée de cette étape est de générer automatiquement un graphe pour chaque ontologie afin d'utiliser la sémantique des termes. Pour préparer le graphe de l'ontologie nous commençons par l'extraction des différents concepts à partir des fichiers OWL en utilisant le «parseur_OWL».

L'aspect le plus important de l'utilisation de l'ontologie est d'identifier les concepts sémantiquement significatifs ou il existe des relations entre eux.

L'architecture de l'ontologie permet de définir un sous-concept (sous-classe ou fils) d'un concept père. En d'autres termes, les concepts de l'ontologie supportent le multi-héritage.

Pour simplifier la recherche dans l'ontologie, nous utilisons « le graphe de concepts ». La (figure III.10) présente un exemple de ce graphe :

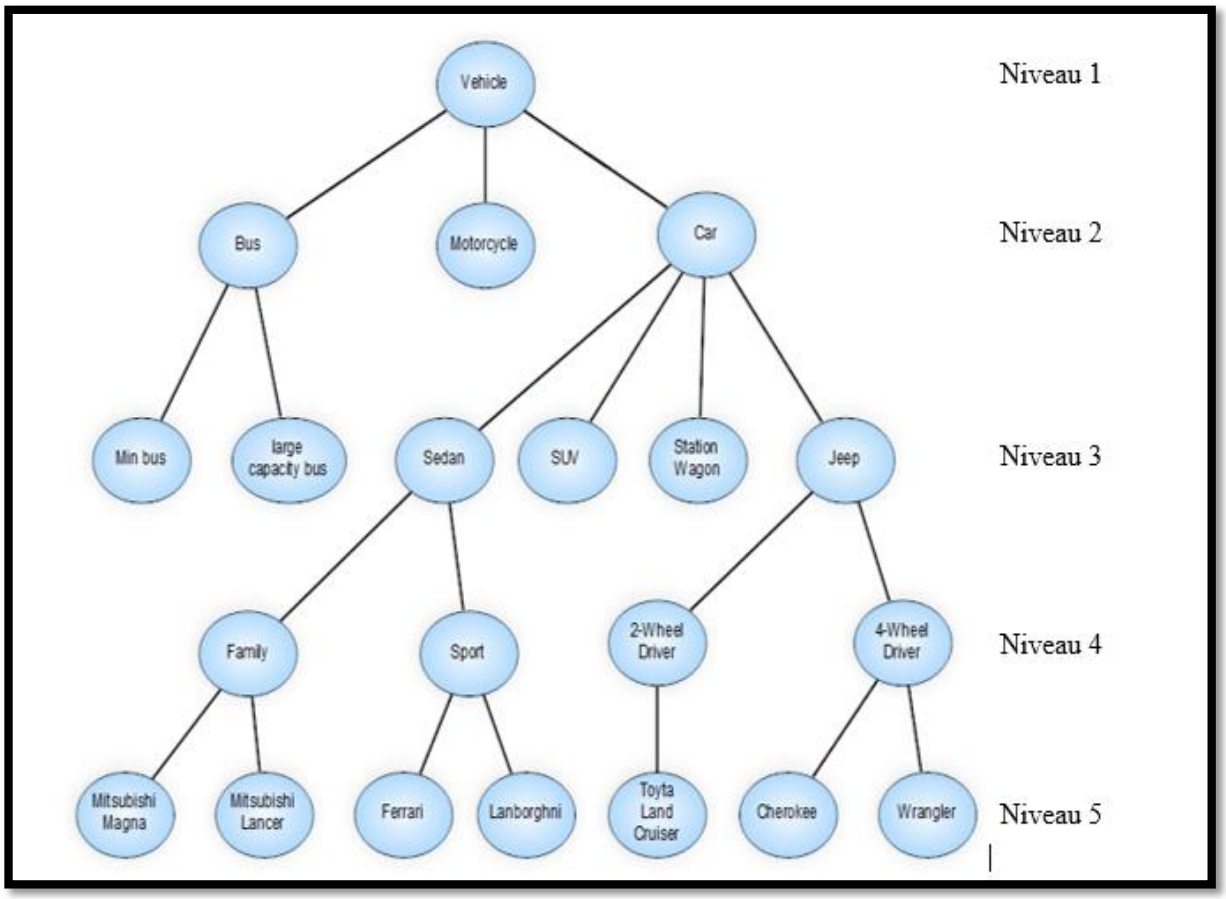

Figure III. 10 : Exemple de l'ontologie véhicule.

3.4. L'algorithme de « parseur OWL »

Le processus de création est basé sur les informations et les données disponibles dans le fichier OWL. Le processus commence par identifier le nom du nœud racine (nom de service), puis parcourir le fichier ligne par ligne (la ligne commencer par '<' jusqu'à '>'). Il fait à chaque ligne l'extraction des concepts et des sous-concepts utilisés pour créer le graphe des concepts, le programme s'arrête à la fin du fichier OWL.

```
Algorithme : parseur de document OWL
Entrée :
     Liste des Fichiers OWL
Sortie :
     Liste des graphes : un graphe à chaque fichier OWL
     (List_onto)
Terminologie :
     Fichier : les fichiers OWL
Debut
     Tant que ( !List_fichiers_OWL.isEmpty) Faire
          graphGraph_onto = new graph ();
          Graph_onto = parser_OWL (fichier_OWL (fichier_i));
          List_onto.add (Graph_onto); i++;
     Fin Tant que ;
Fin.
```

Parser_OWL : fait la lecture du fichier OWL, ensuite l'extraction des concepts en relation (classe et sous-classe) selon l'architecture XML de fichier OWL en construisant le graph au fur et à mesure.

4. Module de construction de graphe de services

Dans notre système, la construction de graphe de services est basée sur le calcul de la distance sémantique entre les termes (la similarité sémantique). Elle est utilisée pour calculer le degré de liaison sémantique et de la similitude entre les mots.

Le graphe de services est un ensemble des services web reliés entre eux par des arcs direct. Un service S1 est relié à S2 si les sorites de S1 sont similaires aux entrées de S2.

Chaque service est représenté par une liste de concepts en entrées et une liste de concepts en sorites. Donc, pour évaluer la similarité il suffit de calculer la distance sémantique entre les deux listes de concepts. Pour cela, nous allons utiliser les ontologies ainsi que la formule de *Tamer* [28] pour calculer la distance sémantique.

4.1.La formule de Tamer A. Farrag, Ahmed I. Saleh, H.A. Ali pour le calcul de la distance sémantique entre deux concepts

De nombreuses expériences sont conçues pour démontrer le grand impact d'utiliser les mesures de distance sémantique dans le domaine des services web. Plusieurs algorithmes sont proposés pour mesurer la distance sémantique entre les concepts.

Dans cette étape nous allons expliquer brièvement la formule [29] que nous avons utilisée, et illustrer comment elle est capable de mesurer la distance sémantique entre deux concepts C1 et C2.

$$SDM(C1, C2) = \begin{cases} \frac{MD - (LWc1 - LWc2) * PL - D}{MD} \% & \text{Pour C1 et C2 liés} \\ 0 & \text{Pour C1 et C2 non liés} \end{cases}$$

$$LW = \frac{MD - concept\ level + 1}{md}$$

Où :

- PL = Représente le nombre d'arcs comptés entre deux concepts C1 et C2 dans l'ontologie.
- MD = Représente la profondeur maximale des concepts dans l'ontologie, MD est utilisé pour éviter les valeurs négatives dans SDM.
- CL = Représente une fonction qui nous donne le niveau du concept.
- D = Représente le nombre d'arcs descendants entre deux concepts C1 et nœud commun.

La valeur de SDM (Semantic Distance Mesure) est égale à zéro s'il n'y a pas de relations (pas de chemin) entre les deux Concepts(en d'autres termes, si ils n'appartiennent pas à la même ontologie).

- Si C1 et C2 représentent le même concept, la valeur de SDM est égale à 100% (Où PL= 0 et D =0).

4.2. La formule de calcul de la distance sémantique entre deux listes de concepts

Si *L1* est la première liste de concepts et *L2* la deuxième, alors, pour calculer la distance sémantique entre les deux listes nous allons utiliser la formule suivante :

$$FSDM(L1, L2) = \sum_{x=0}^{m} \sum_{y=0}^{n} \frac{\max(\text{SDM}(x, y))}{m + n}$$

Tel que : $x \in L1$, $y \in L2$ et FSDM : Final DSM

4.3. La correspondance entre les services web sémantiques

Pour faire une correspondance entre un service web (S1) et un autre service web (S2) (dans le but de créer un arc entre S1 et S2) nous allons utiliser3 listes de concepts comme montré dans la figure III.8. Ces 3 listes sont :

- liste des sorties de S1 (O_S1),
- liste des entrés de S2 (I_S2),
- liste des ontologies (O).

Ensuite, une méthodologie générale pour faire une correspondance entre deux listes de concepts sera introduite. Dont l'objectif principal est d'obtenir la distance sémantique maximale.

La première étape consiste à calculer la distance sémantique entre les deux listes : O_S1 et I_S2, en utilisant la formule : FSDM *(O_S1, I_S2)*

La deuxième étape consiste à évaluer cette valeur :

- *FSDM (O_S1, I_S2)>= seuil* ➔La similarité est considérable, donc un arc entre S1 et S2 est créé dans le graphe des services.
- *FSDM (O_S1, I_S2) < seuil* ➔La similarité est considérable est négligeable, donc aucun ne sera créé

Un *seuil* : est une valeur fournit par l'utilisateur pour limiter ou élargir le résultat afin d'obtenir la précision qu'il veut.

Si le seuil = 30% alors ça veut dire que la liste L1 est similaire à la liste L2 par un pourcentage de 30%

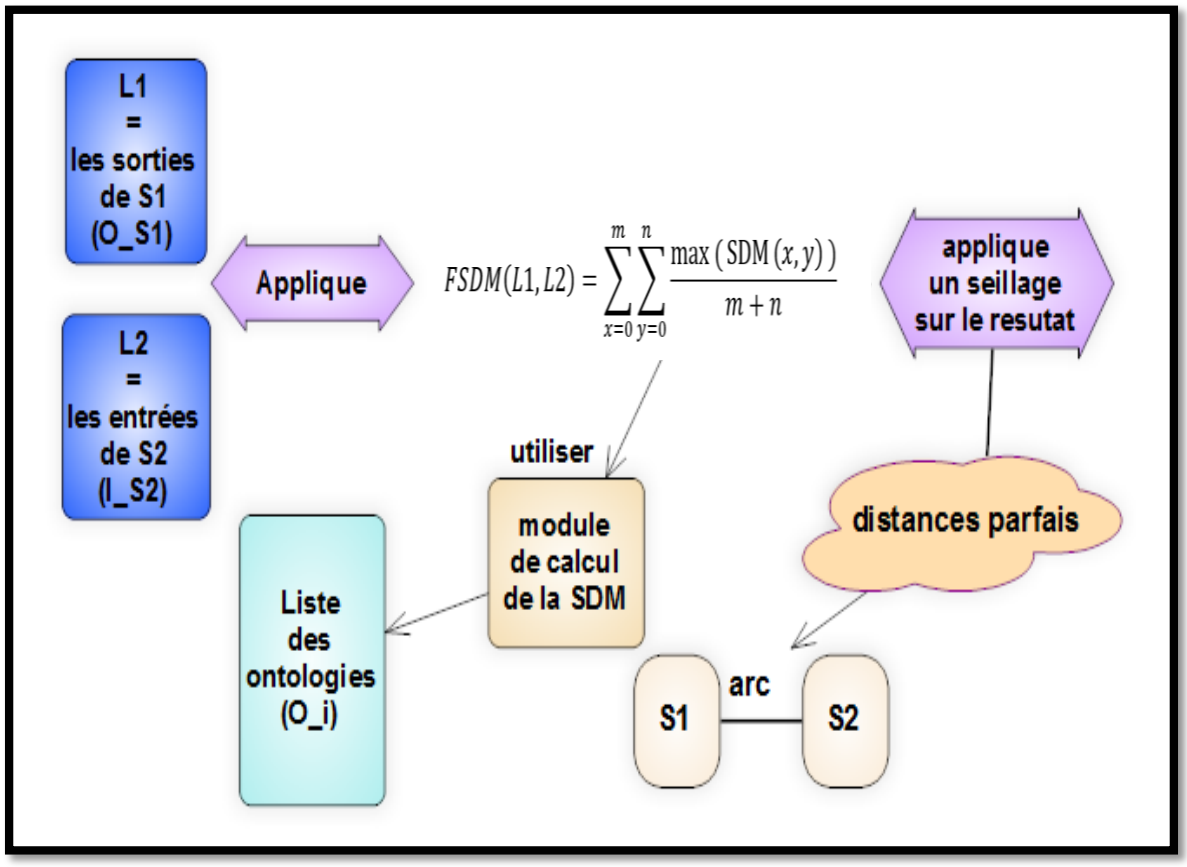

Figure III. 11 : Exemple d'une Correspondance entre deux services S1 et S2.

4.4. Algorithme de construction de graphe de services

```
Algorithme : construction de graphe de services
Entrée :
      Liste des informations des services (L_S) ;
      Liste d'ontologie(O) ;
      Seuil ;
Sortie :
      Graphs de service (Graph_serv)
Debut
      Créer_noeuds (L_S);
      Pour i allons de 0 à L_S.taille Faire
            Pour j allons de 0 à L_S.taille Faire
                  Si (i= !j) alors
                        Pour k allons de 0 à O.taille Faire
                              FSDM=clacul_distance (L_S (i), L_S (j),
                              O (k));
                              Si (FSDM >= seuil) alors
                                    Créer_arc (L_S (i), L_S (j)) ;
                              Fin si ;
                         Fin pour ;
                   Fin si ;
            Fin pour ;
      Fin pour ;
Fin.
```

5. Algorithme de découverte

Dans notre système, l'algorithme de découverte prend en entrée la requête de l'utilisateur ainsi que le graphe de services dans le but de déterminer un enchainement des services considéré comme réponse à l'utilisateur (figure III.12).

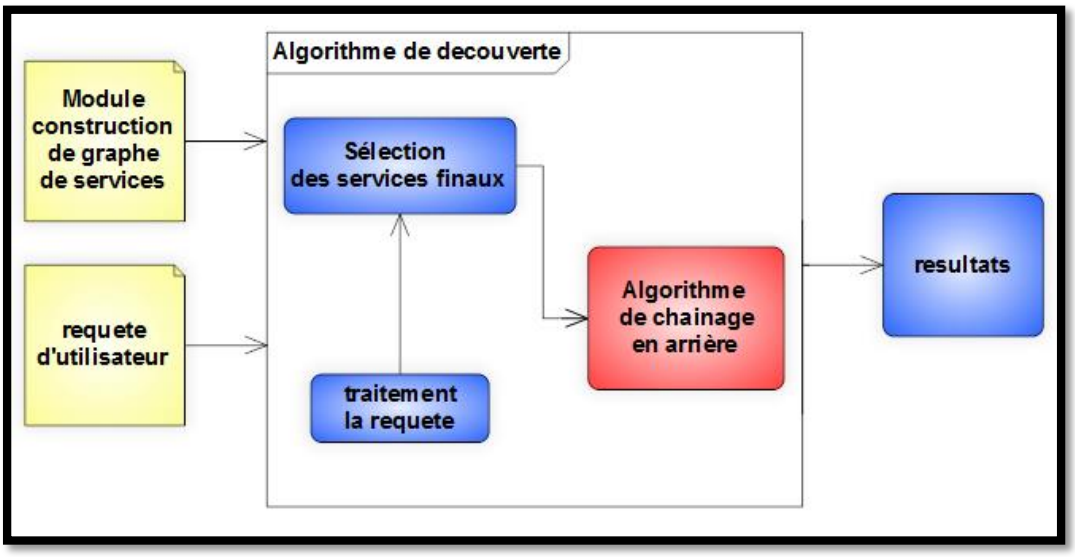

Figure III. 12 : Algorithme de découverte.

5.1. Le traitement la requête

L'objectif de cette étape est d'extraire les termes utiles dans une requête.

5.2. Sélection des services finaux

Un service final est déterminé en comparant les sorties de chaque service avec les sorties de la requête. La similarité est évaluée en calculant la distance sémantique entre les deux listes (les sorties du service et les sorties de la requête). Le but de construire la liste des services finaux est pour lancer l'algorithme de chainage en arrière à partir de ces services.

5.3. Algorithme de chainage en arrière

Cette approche est appelée planification régressive puisqu'elle se déplace vers l'arrière. Elle consiste à parcourir le graphe de services, non pas en partant de l'état initial (service initial), mais en partant du but final de la requête (qui doit donc être explicitement connu) et en appliquant l'inverse du parcoure pour produire des sous-buts jusqu'à arriver à l'état initial. Si l'algorithme permet de remonter à un état initial, alors la solution est trouvée.

Cet algorithme est présenté comme suit :

```
Algorithme : chainage en arrière
Entrée :
      Service web final (WS) ;
      Liste des entrées de la requête (L_E) ;
      Liste de solution (L_S) ;
Sortie :
      Affichage la solution ;
Debut
      List-input=slection les entrées de service(WS) ;
      D = Calculer la distance (List-input, L_E) ;
      SI (D >= seuil) alors
         Afficher la solution(L_S) ; //l'affiche est l'inverse.
      Fin si ;
      List_pere =Sélectionner père de service (WS) ;
      Tant que (List_pere non vide) Faire
          Si( ! L_S.contient(List_pere(i))) alore//éviter le cycle
                L_S.add (List_pere (i));
                Chainage_en_arriere (List_pere (i), L_E, L_S);
                L_S.remove (List_pere (i));
          Fin si ;
      Fin Tant que ;
Fin.
```

La figure suivant présente l'algorithme de chainage en arrière :

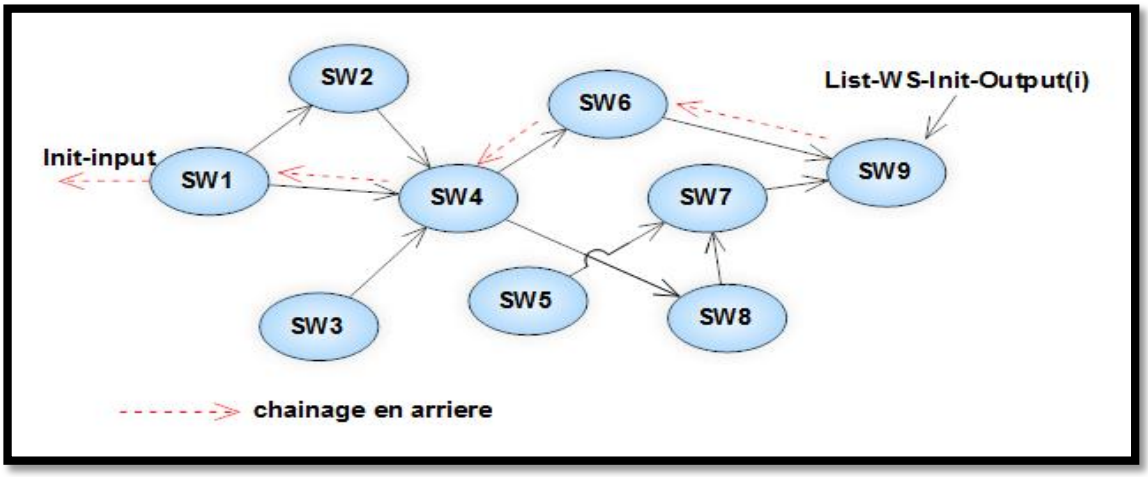

Figure III. 13 : Algorithme de chainage en arrière.

5.4.Algorithme découverte

L'algorithme principal de la découverte est un algorithme qui répond à certaines requêtes d'utilisateur. Chaque exécute cet algorithme lorsqu'il reçoit une requête de découverte des services. Il est définit comme suit :

```
Algorithme : Algorithme principale de découverte
Entrée :
     Liste des informations des services (L_Inf_S) ;
     Graph de service (graph_serv) ;
     Requête (R) ;
Sortie :
     Affichage la solution ;
Debut
  R_E=Traitement(Requête_E) ; //traitement le requête entrée
  R_S=Traitement(Requête_S) ; //traitement le requête sortie
  List-WS-Final-Output=selection_serv (R_S);//les services finaux
     Tant que (List_pere non vide) Faire
       L_S.add (List-WS-Final-Output (i));
       Chainage_en_arriere (List-WS-Final-Output (i), R_E, L_S);
       L_S.clear () ; //Vider la List de solution
     Fin tant que;
Fin.
```

Pour mieux expliquer le rôle de l'algorithme présenté précédemment, nous présentons un exemple d'un service web composite qui calcule le prix d'un livre en Dinar Algérien (DA). Ce Web service est composé de quatre services (figure III.14).

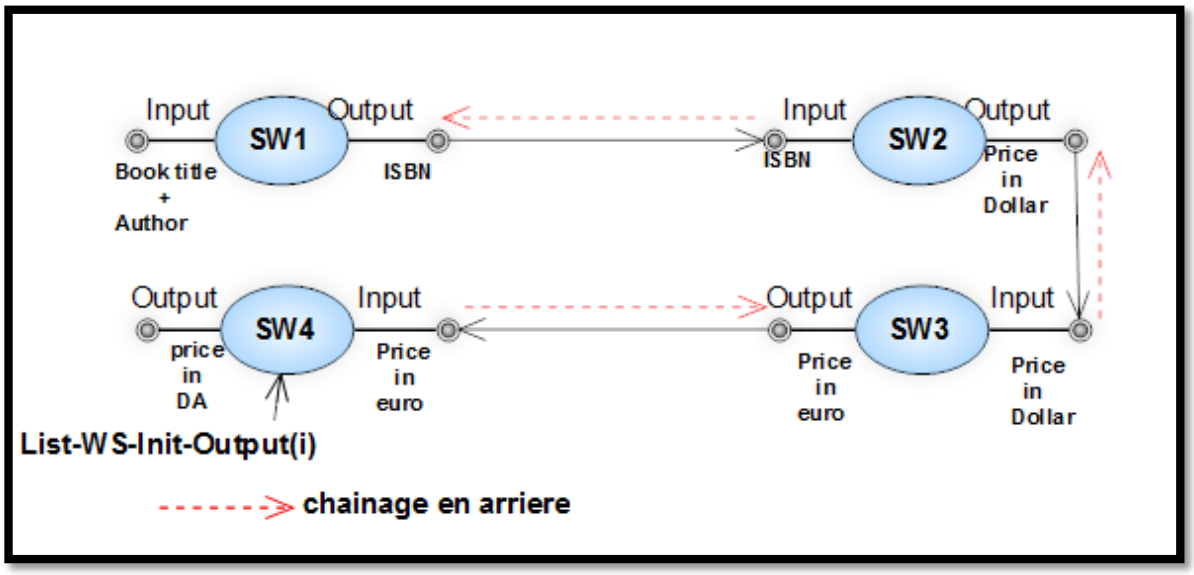

Figure III. 14 : Exemple d'un service web composite calculant le prix d'un livre en DA.

6. Module de présentation

Le module de présentation prend en charge le coté affichage des résultats dans notre système. Il présente à l'utilisateur des services classés, enrichis par des informations d'explication sur chaque page afin de lui permettre de comprendre plus facilement le contenu des documents qui lui sont retournés.

La (figure III.15) montre le principe de fonctionnement du module de présentation :

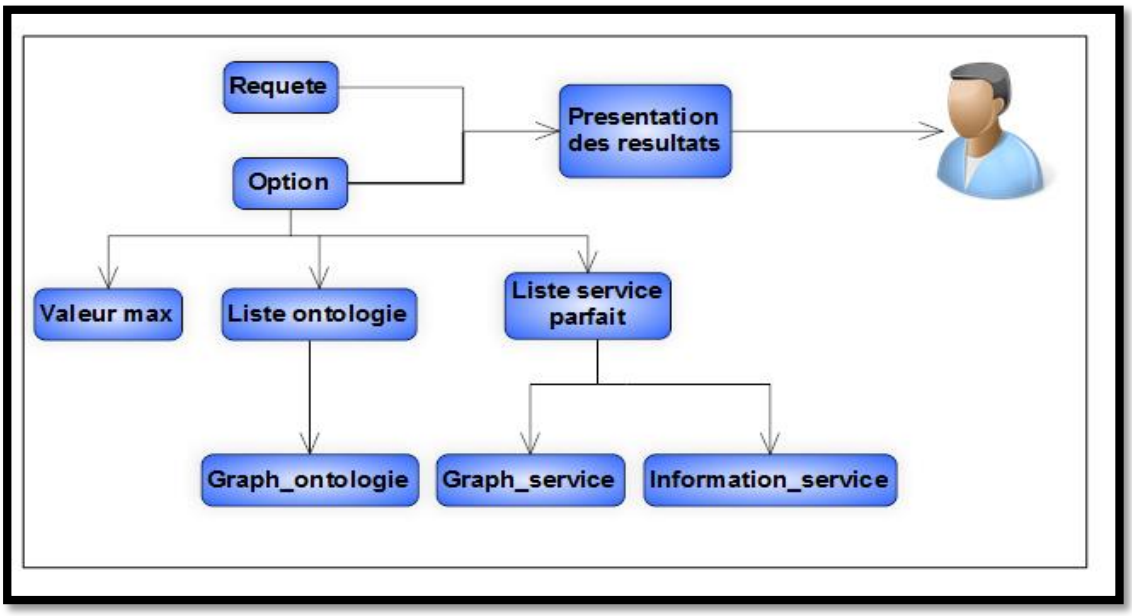

Figure III. 15 : Module de présentation.

Conclusion

Dans ce chapitre, nous avons introduit les modèles supports de l'approche que nous avons utilisée pour une meilleure découverte des services web. Notre modèle est fondé sur la distance sémantique et l'algorithme de chaînage en arrière.

Le système est composé d'un ensemble de modules : Un module de gestion des services web qui repose sur le parseur et le traitement du contenu d'un document OWLS afin d'extraire le vecteur de données pour chaque service Web.

Le module de gestion des ontologies, où la construction du graphe de concepts est nécessaire pour construire le graphe des services, et par conséquence améliorer les résultats de la découverte en intégrant l'aspect sémantique.

Ensuite, le module de construction d'un graphe global des services web (les services web sont associés les uns aux autres grâce au calcul de la distance sémantique).

Et nous avons terminé par la découverte des services web basée sur l'algorithme de chaînage en arrière.

Dans le chapitre suivant nous allons présenter l'implémentation de ce modèle et aussi expliquer les différentes interfaces de l'application réalisée.

Chapitre IV

Implémentation

Introduction

Nous avons vu dans le chapitre précédent l'architecture de notre système ainsi que la description et le rôle des modules qui le constituent. Ce chapitre est organisé de la façon suivante : nous commençons tout d'abord par une présentation des environnements utilisés pour le développement de notre système, à savoir : le langage de programmation, les outils utilisés pour l'implémentation, et les bibliothèques utilisées. Et par la suite, nous allons présenter la description et les fonctionnalités de l'application en décrivant sa structure.

1. Environnements utilisés pour le développement

Dans cette section nous présentons les principes d'implémentation de notre application, pour cela nous définissons les différents outils de programmation que nous avons utilisés, afin d'atteindre le but de la recherche sémantique des services web. L'environnement de développement est constitué de :

- ✓ Le langage de programmation java.
- ✓ L'environnement de programmation NeatBeans.
- ✓ La base de données :
  - Les ontologies.
  - Les services web.

- La base de données : est la troisième version de la collection de test de services web OWLS nommé OWLS-TC3. La collection est destinée à soutenir l'évaluation de la performance des algorithmes de découverte de services. La base de données fournit 1007 services Web sémantiques écrits en OWLS, et 32 ontologies OWL.

1.1. Langage de programmation

Notre choix du langage de programmation s'est porté sur le langage JAVA et cela pour diverses raisons :

- ✓ JAVA est un langage orienté objet simple ce qui réduit les risques d'incohérence.
- ✓ JAVA est portable et multi – plateforme.
- ✓ JAVA possède une bibliothèque riche de classes.

- ✓ Java est un langage "multi threade" : Le thread (processus léger) est une partie de code, un "flot d'instructions" qui s'exécute en concurrence avec d'autres threads dans un même processus, cela permet à un seul programme d'effectuer plusieurs activités simultanément.
- ✓ Le JDK (Java Development Kit), regroupe l'ensemble des éléments permettant le développement, la mise au point et l'exécution des programmes Java. Le JDK et la documentation sont librement téléchargeables sur le site web de Sun.

1.2. Environnent de programmation « NetBeans »

NetBeans est un environnement de développement intégré (EDI) pour Java, placé en open source par Sun en juin 2000 sous licence CDDL et GPLv2 (Common Development and Distribution License). En plus de Java, NetBeans permet également de supporter différents autres langages, comme Python, C, C++, XML, Ruby, PHP et HTML. Il comprend toutes les caractéristiques d'un IDE moderne (éditeur en couleur, projets multi-langage, refactoring, éditeur graphique d'interfaces et de pages Web).

Conçu en Java, NetBeans est disponible sous Windows, Linux, Solaris (sur x86 et SPARC), Mac OS X et Open VMS.

1.3. Les ontologies

L'objectif principal de notre travail consiste à prendre en compte la sémantique en se basant sur l'ontologie dans un système de recherche d'information. Le système utilise un ensemble des ontologies (OWLS-TC3).

1.4. Les services web (OWLS)

C'est un ensemble de documents OWLS où chaque document est un service web, ces services sont utilisés pour le test du bon déroulement de notre système. Nous avons utilisé les services d'OWLS-TC3.

2. Implémentation

Le système que nous avons développé est un système de recherche sémantique des services web composés basé sur un algorithme de découverte (nous l'avons appelé «découverte»). Découverte fait une correspondance entre ce qui est retournés par la gestion de composition (graph globale) par rapport à la requête utilisateur traité. Ces résultats sont obtenus par une projection sémantique de la requête sur l'ontologie. Dans ce qui suit nous présentons et nous décrivons les fonctionnalités offertes par l'application découverte.

2.1. Présentation de l'application découverte

Au lancement de l'application découverte, l'interface de l'utilisateur s'affiche, il s'agit d'une interface simple d'utilisation pour permettre à l'utilisateur d'effectuer des découvertes sémantique. La figure VI.1 présente la fenêtre de l'utilisateur :

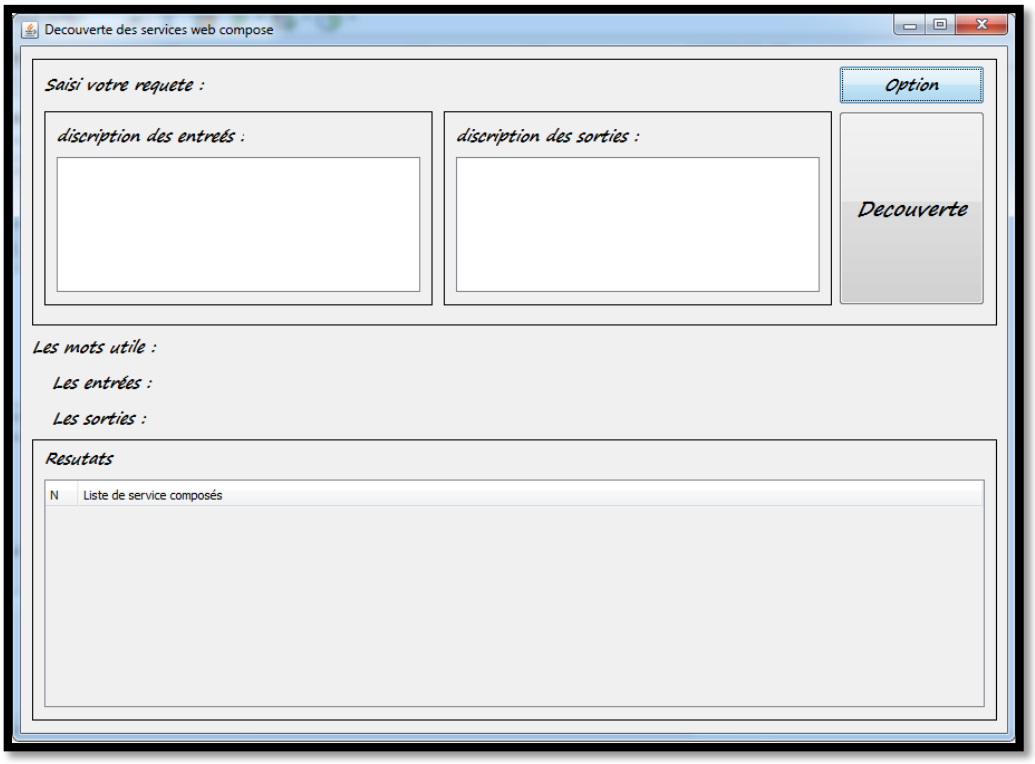

Figure IV. 1 : l'interface utilisateur.

L'interface utilisateur se compose de plusieurs zones :

- ✓ **Zone de recherche** : elle permet à l'utilisateur d'exprimer son besoin sous forme d'une requête et de faire passer la requête à « découverte ».

- ✓ **Zone de résultats** : c'est le volet dans lequel le système affiche les résultats rendus par le « découverte ».

Dans l'interface principale, l'interface option s'affiche en cliquant sur le bouton « Option ».

L'interface option se compose de plusieurs zones comme montré dans la figure VI.2 :

- ✓ **Zone d'ontologie** : elle représente la partie ou le système affiche le lien de répertoire qui contient les ontologies (avec possibilité de modifier le lien), l'affichage de l'ontologie est fait sous forme d'une table.
- ✓ **Zone de service web** : elle représente la partie où le système affiche le lien de répertoire qui contient les services web (avec possibilité de modifier le lien aussi), l'affichage des services est fait sous forme d'une table aussi.
- ✓ **Zone de seuil** : elle représente la partie ou le système affiche la valeur prise pour le calcul de la distance sémantique (avec possibilité de modifier cette valeur).

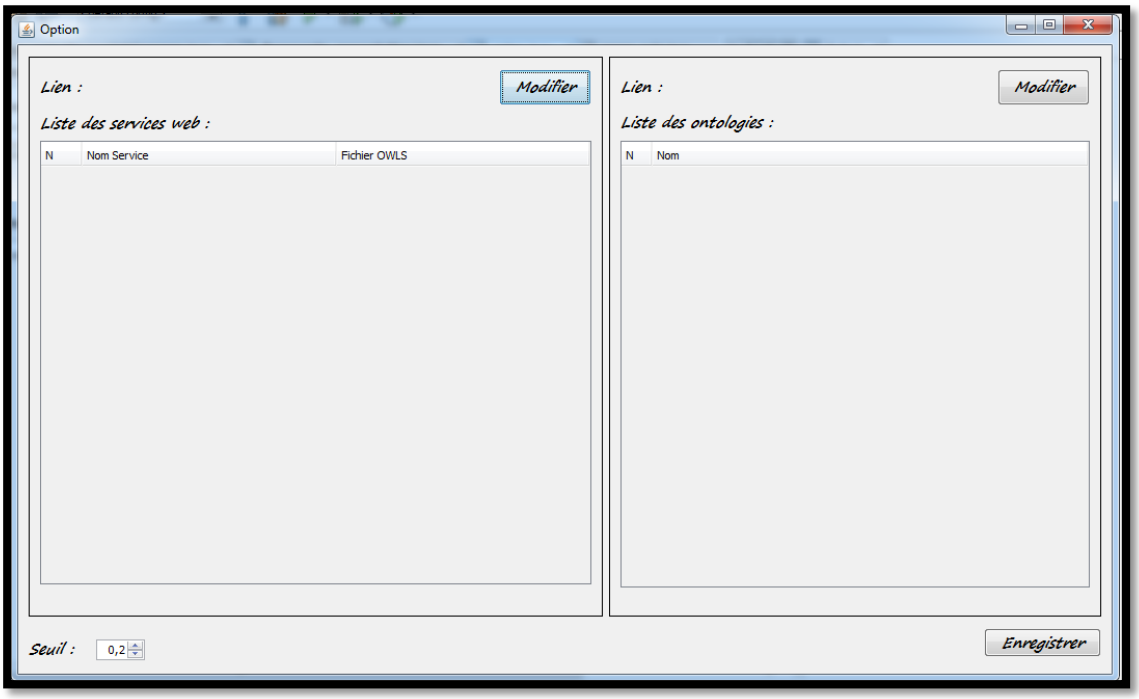

Figure IV. 2 : l'interface option.

Dans la zone service web (au niveau de l'interface option), l'interface graphique s'affiche en cliquant par le bouton gauche de la souris sur la ligne d'un service.

L'interface graphique se compose d'une zone comme montré dans la figure VI.3 :

- ✓ **Zone de dessin** : le système affiche le service web sélectionné avec d'autres services selon la valeur de calcul de la distance sémantique entre les d'entrées et les sorties de chaque service.

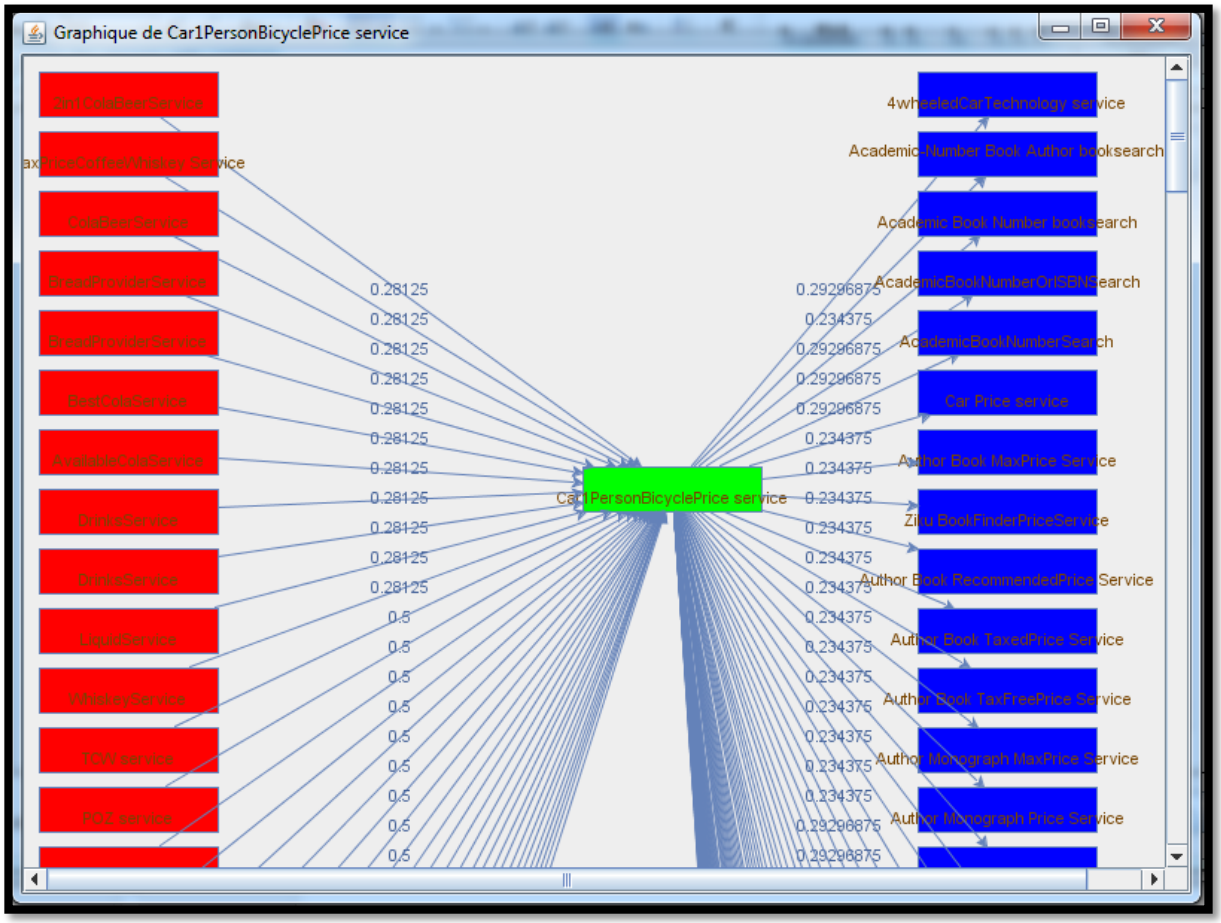

Figure IV. 3 : Exemple de l'interface graphique de Car1PersonBicyclePrice service.

Dans la zone service web (au niveau de l'interface option), l'interface information s'affiche en cliquant par le bouton droit de la souris sur la ligne d'un service.

L'interface information se compose d'une zone comme montré dans la figure VI.4 :

- ✓ **Zone information** : le système affiche les informations du service web sélectionné.

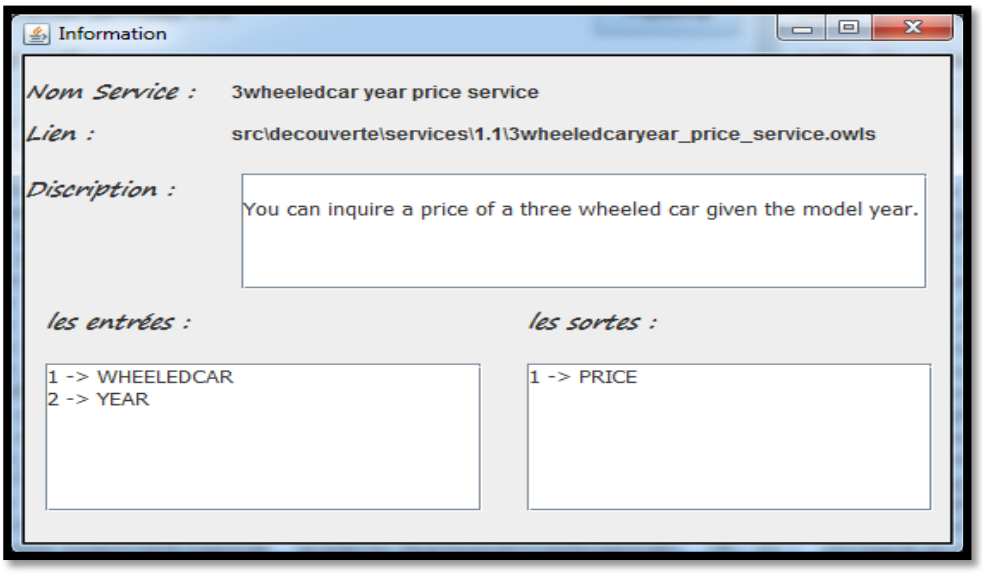

Figure IV. 4 : Exemple d'interface information.

Dans la zone d'ontologie (au niveau de l'interface option), l'interface graphique s'affiche en cliquant par le bouton gauche de la souris sur la ligne d'une ontologie.

L'interface graphique se compose d'une zone comme montré dans la figure VI.5 :

- ✓ **Zone dessin** : le système affiche le graphe d'une ontologie sélectionné qui contient l'ensemble des concepts associés les uns aux autres.

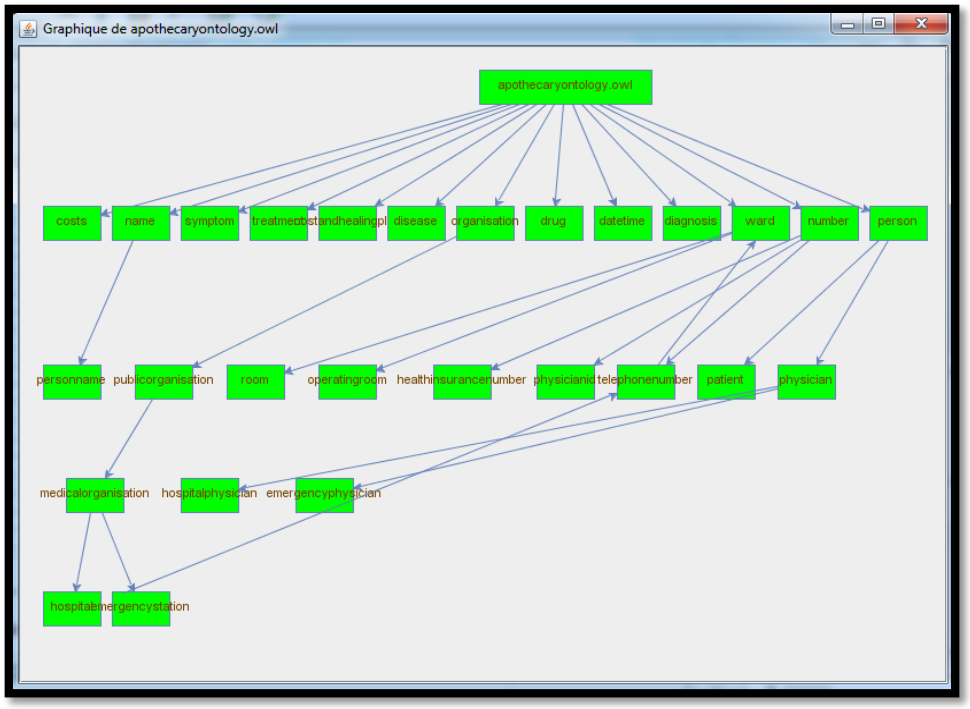

Figure IV. 5 : Exemple de l'interface graphique d'ApothecaryOntology.owl.

Un message de confirmation (au niveau de l'interface option) s'affiche en cliquant sur le bouton « enregistrer».

Message de confirmation comme montre la figure VI.6 :

- ✓ **Message de confirmation** : il représente la partie ou le système affiche un dialogue avec utilisateur lors d'un changement de : lien de répertoire des services, lien de répertoire des ontologies et valeur de composition.

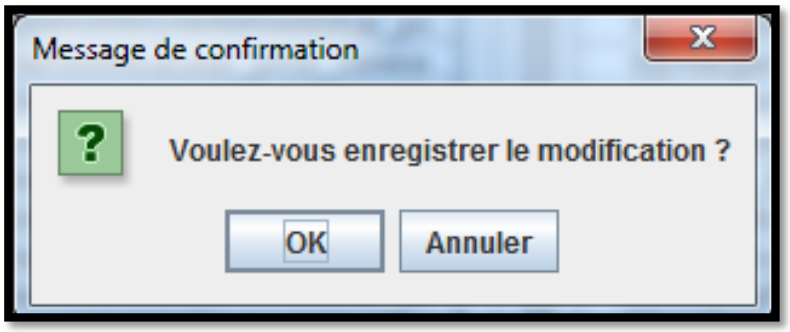

Figure IV. 6 : Message de confirmation.

2.2. Mécanisme et Principe de fonctionnement

L'objectif de l'utilisation de l'application découverte est d'accéder à des informations pertinentes par une recherche sémantique, guidée par les ontologies.

Lorsque l'utilisateur soumit son besoin des services sous forme d'une requête à «découverte», une clique sur le bouton «découverte», donnera lieu au déclenchement de processus de découverte qui coopère avec d'autre processus afin de répondre au besoin de l'utilisateur. Ces processus sont relatifs aux modules décrits dans la partie précédente. L'exécution se déroule de la manière suivante :

a. Construction de graphe de services

Le processus de construction de graphe de services est une suite d'étapes d'extraction des informations faites sur les fichiers OWLS et les fichiers OWL, en suite un calcul de la distance sémantique de chaque service avec tous les autres services par l'intermédiaire de l'ontologie. Enfin les services sont associés dans un graphe global et envoyé au processus de découverte.

b. Sélection des services

Le processus de sélection des services commence par la récupération de la requête client (entrées ou sorties), puis il lance le traitement de la requête. Le résultat est deux listes (entrée et sortie) des concepts utiles. La liste de sortie est utilisée pour déterminer les services initiaux.

c. Algorithme de chainage en arrière

L'algorithme de chainage en arrière récupère les résultats de la sélection des services et le graphe de services. Nous avons appliqué cet algorithme pour parcourir le graphe et construire une chaine de services à partir de chaque service initial.

d. Calcule de la distance sémantique

Le processus de calcul de la distance sémantique est un mécanisme permettant le calcul de la distance entre la requête et les entrées ou sortie d'un service web. La distance utilise des formules pour calculer la mesure de similarité.

Conclusion

Nous avons présenté dans ce chapitre une implémentation de notre application nommée « découverte ». Le travail réalisé est un système de découverte des services web guidé par des ontologies. Ces ontologies permettent d'ajouter une couche sémantique à notre système et d'effectuer une recherche par la sémantique des mots.

L'implémentation a été réalisée en utilisant le langage de programmation Java et l'environnement développement NeatBeans.

Conclusion générale

Comme toute technologie, les services web génèrent beaucoup d'intérêt et promettent de nombreux avantages.

L'architecture des services web fournit une infrastructure pour décrire (WSDL), découvrir (UDDI), et invoquer (SOAP) des services web. Ils sont basés sur XML qui constitue la technologie utilisée pour développer les services web.

Cependant, un seul service peut ne pas suffire à satisfaire tous les besoins. La plupart du temps il est nécessaire de composer plusieurs services web. Nous parlons donc, de la composition des services web qui a connu beaucoup d'intérêt ces dernières années.

L'idée de la composition de services web est de créer un nouveau service en combinant des services existants pour répondre aux besoins de l'utilisateur.

Au début, nous avons construit un graphe qui rassemble tous les services web en utilisant deux parseurs XML (le parseur des ontologies et le parseur des services web) et le calcul de la distance sémantique pour relier deux services dans le graphe.

Ensuite, Nous avons développé un algorithme de découverte qui répond aux requêtes des clients. Il est basé sur l'algorithme de chainage en arrière et le calcul de la distance sémantique pour découvrir les services au niveau du graphe construit au début.

Le travail ouvre de nombreuses perspectives aussi bien au niveau amélioration de certains composants qu'au niveau proposition d'une classification des services web de l'UDDI afin de minimiser le champ de recherche pour que la découverte vise seulement la classe concernée et ne pas la totalité des services.

Références bibliographique

- [01] : http://www.w3.org/TR/2003/WD-ws-gloss-20030514/# webservice.
- [02] : Jeremy Fierstone, les services web, SAR5– Novembre 2002.
- [03] : KHALFAOUI IMEN : Conception et réalisation d'un système pour la composition de services web, application au e_learning, 2007/2008.
- [04] : http://www.w3.org/TR/soap/.
- [05] : ANISS ALKAMARI : Composition de services web par appariement de signatures. Mémoire présenté comme exigence partielle de la maitrise en informatique, Janvier 2008.
- [06] : Samir Tata, Services Web, Département INFormatique .TELECOM SudParis, 2009-2010.
- [07] : Mickaël BARON, WSDL : Décrire et configurer SOA –Web Services, 2010.
- [08] : Ramdani Nabil, un système pour la composition automatique des services web basée sur leurs comportements, soutenu le 23.06.2010.
- [09] : Mohamed Gharzouli. Composition des Web Services Sémantiques dans les systèmes Peer-to-Peer ,25/09/2011.
- [10] : Céline LOPEZ-VELASCO, Sélection et composition de services Web pour la génération d'applications adaptées au contexte d'utilisation ,18 novembre 2008.
- [11] : Adel BOUKHADRA, La composition dynamique des services Web sémantiques à base d'alignement des ontologies owl-s, Ecole Doctorale en Science et Technologie de l'Information et de la Communication « STIC », Mémoire de Magister, 2011.
- [12] : G. Gardarin, Xml, des bases de donn_ees aux services web, Dunod, 2007.
- [13]: S. Dustdar and W. Schreiner, A survey on web services composition, International Journal of Web and Grid Services 1 (2005).
- [14] : Yasmine CHARIF, Choregraphie dynamique de services basee sur la coordination d'agents introspectifs UNIVERSITE PIERRE ET MARIE CURIE, decembre 2007
- [15] : Composition semi-automatique de Services Web, Nerea Arenaza Fevrier 2006
- [16] : S Jamal (épouse Sanlaville), Environnement de procédé extensible pour l'orchestration : Application aux services Web, Doctorat de l'Université Joseph Fourier de Grenoble I ,13/12/2005.
- [17] : H. D. Rojas, Orchestration à haut niveau et BPEL, Master Mathématiques Informatique 2e année, Recherche Spécialité Systèmes et Logiciels, Université Joseph Fourier de Grenoble, Soutenu le 21 juin 2006.

- [18] : J. Guitton, Planification multi-agent pour la composition dynamique de services Web, Université Joseph Fourier, Grenoble I, Informatique et Mathématiques Appliquées, Rapport de stage Master 2 Recherche, Juin 2006.
- [19] : N. Arenaza, Composition semi-automatique de Services Web, Projet de Master, Ecole Polytechnique Fédérale de Lausanne (EPFL), Février 2006.
- [20] : S. Rampacek, Sémantique, interactions et langages de description des services Web complexes, Doctorat de l'Université Reims Champagne Ardenne ,10 /10/ 2006.
- [21] : H. Duarte Amaya, Tcows : Canevas pour la composition de services web avec propriétés transactionnelles, Doctorat de l'Université Joseph Fourier ,13/10/2007.
- [22] : Farhat BENSEDDIK et Nabil RAMDANI, Un système pour la composition automatique des Services Web basée sur leurs comportements, Ecole Nationale Supérieure d'Informatique, Mémoire d'ingénieur, Alger, 2010.
- [23] : Wohed P. et al , Pattern Based Analysis of BPEL4WS. 2002.
- [24]: Rada R, Mili H, Bicknell E, Blettner M. Development and application of a metric on semantic nets. IEEE Trans Syst Man Cybern, 1989.
- [25]: Leacock C, Chodorow M. Combining local context and WordNet similarity for wordsense identification. In: Fellbaum C, editor. WordNet an electron, Lex database. MIT Press; 1998.
- [26]: Tversky A. Features of similarity. Psychol Rev, 1977.
- [27]: Couto FM, Silva M, Coutinho PM. Implementation of a functional semantic similarity measure between gene-products. Department of Informatics, University of Lisbon; 2003.
- [28]: Semantic web services matchmaking: Semantic distance-based approach.