Université Abou Bekr Belkaid
Tlemcen Algérie
جامعة أبي بكر بلقايد
تلمسان الجزائر

République Algérienne Démocratique et Populaire
Université Abou Bakr Belkaid– Tlemcen
Faculté des Sciences
Département d'Informatique

Mémoire de fin d'études

pour l'obtention du diplôme de Master en Informatique

Option: *Modèle Intelligent et Décision(M.I.D)*

Thème

Recherche des Services Web à base de mesures de similarités

Réalisé par :

Mlle. MEZIANE Wissame Soulaf

Présenté le 04 Juillet 2011 devant la commission composé de MM.

Président : - *Mr. BENAMAR. A*

Encadreur : - *Mr. HADJILA.F*

Examinateurs : - *Mr. BENAZZOUZ .M* / - *Mm. ILES.N*

- *Mr. BENTAALLAH .A* / - *Mr. SMAHI.M.I*

- *Mr. BENZIANE .Y*

Année Universitaire : 2010-2011

Table des matières

Table des matières

Chapitre II : Les Ontologies

Chapitre III : Les mesures de similarités

Chapitre IV : Conception et réalisation du prototype

Liste des figures

Chapitre I : les Services Web

Chapitre II : les Ontologies

Chapitre III : les Mesures de Similarités

Chapitre IV : Conception et réalisation du prototype

Introduction générale

I. Contexte

L'architecture orientée services est une réponse très efficace aux problématiques que rencontrent les entreprises en termes de réutilisabilité, d'interopérabilité et de réduction de couplage entre les différents systèmes qui implémentent leurs systèmes d'information. SOA est un paradigme fondée sur la description de services et de leurs interactions.

Les caractéristiques principales d'une SOA sont :

- *Le couplage faible entre les services* implique qu'un service n'appelle pas directement un autre service. En effet, les interactions sont gérées par une fonction d'orchestration. La réutilisation d'un service est alors plus facile, du fait qu'il n'est pas directement lié aux autres services de l'architecture dans laquelle il évolue.
- *L'indépendance par rapport aux aspects technologiques* est obtenue grâce aux contrats d'utilisation associés à chaque service. En effet, ces contrats sont indépendants de la plate-forme technique utilisée par le fournisseur du service.
- *La mise à l'échelle* est rendue possible grâce à la découverte et à l'invocation de nouveaux services lors de l'exécution.

Les architectures orientés services ont été popularisées avec l'apparition de standards comme les **Web Services** dans l'e-commerce (commerce électronique) (B2B, inter-entreprise, ou B2C, d'entreprise à consommateur), basés sur des plates-formes comme J2EE ou .NET.

Les Services Web représentent une instance particulière du paradigme SOA, Ils permettent l'interopérabilité des applications ainsi que l'intégration des applications sur le Web, C'est grâce à ces avantages qu'ils sont devenus le point de convergence technologique de l'industrie logiciel.

En plus, la croissance rapide et continue du volume d'information complique de plus en plus les taches de recherches, d'organisation, d'accessibilité et de la maintenance d'information, pour résoudre ces problèmes, les informations doivent être comprises et traitées automatiquement par la machine, d'où l'apparition du Web sémantique qui permet aux machines de mieux exploiter les contenus des sources d'information afin de proposer un niveau de services qualitativement meilleur.

II. Problématique

Les grands acteurs industriels ont crée le standard UDDI pour faciliter la recherche des Services Web. Cependant, cette recherche supporte uniquement la syntaxe des entrées/Sorties, elle ne donne pas l'importance au contenu sémantique qui peut être interprété différemment chez le client et le Service Web.

D'un autre coté, on constate une augmentation exponentielle du nombre des Services disponibles sur le Web. Cette situation nous oblige de créer une solution efficace pour la recherche des Services Web, tout en prenant compte de la sémantique de la requête.

III. Contribution

L'objectif de ce mémoire est de proposer deux approches de recherches approximatives des Services Web qui se basent sur la notion de similarité sémantique.

IV. Plan du mémoire

Le reste du mémoire est structuré comme suit :

Chapitre I

Dans ce chapitre on va présenter les technologies des services web et les principaux standards qu'il supporte, et parmi ces derniers, on cite les protocoles SOAP, WSDL, UDDI, ainsi que les avantages et les inconvénients des Services Web.

Chapitre II

Ce chapitre expose la notion d'ontologie en présentant ces différents constituants, ces types, son cycle de vie, ces critères d'évaluation ainsi que ces méthodologies et ces outils de construction.

Chapitre III

Le troisième chapitre présente quelques mesures de similarités syntaxiques et sémantiques, ainsi que la solution proposée qui aide à améliorer la recherche sémantique des Services Web.

Chapitre IV

Ce chapitre présente le prototype destiné à la recherche sémantique des Services Web.

Conclusion générale

La conclusion générale résume les résultats de notre travail, et présente les perspectives que nous souhaiterons réaliser dans le futur.

CHAPITRE I

LES SERVICES WEB

I. Introduction

Avec l'interconnexion des ordinateurs en réseau et en particulier à travers Internet, il devient possible de faire fonctionner des applications sur des machines distantes. L'intérêt d'une application fonctionnant à distance peut à première vue sembler inutile dans la mesure où les applications fonctionnent fort bien en local néanmoins une application distante peut répondre aux problématiques suivantes :

- Les données peuvent être présentes uniquement sur le serveur distant.
- Le serveur distant peut disposer d'une puissance de calcul ou de capacités de stockage dont l'utilisateur local ne dispose pas.
- L'application distante peut être utilisée simultanément par un grand nombre d'utilisateurs et sa mise à jour n'intervient qu'à un seul endroit.

Pour ses raisons, une interaction entre des programmes distants peut être utile. Les Services Web apportent une solution à ce problème en définissant une manière standard d'invoquer une application distante et d'en récupérer les résultats à travers le web.

Les Services Web sont un sujet complexe, intégrant plusieurs domaines informatiques (documents, sécurité, programmation, etc.) Le nombre de normes est impressionnant, mais il est dépassé par la multitude de solutions d'implémentation.

L'objectif de ces derniers est de faciliter l'accès aux applications entre les entreprises et ainsi de simplifier les échanges de données. Ils poursuivent un vieux rêve de l'informatique distribuée où les applications pourraient inter-opérer à travers le réseau, indépendamment de leur plate-forme et de leur langage d'implémentation.

Cette technologie, initiée par IBM et Microsoft, puis en partie normalisée par le W3C, est maintenant acceptée par l'ensemble des acteurs de l'industrie informatique sans exception. C'est surtout ce point qui fait des Services Web une technologie révolutionnaire et qui les rend aussi populaire.

Lors de cette présentation, nous essaierons d'améliorer notre compréhension des Services Web, nous ferons un survol des normes les plus importantes de l'industrie et nous regarderons qu'elles sont les alternatives d'implémentations.

II. Historique

Le World Wide Web s'accroît de manière exponentielle depuis les années suivantes **[21]** :

- ***En 1990*** : Ce phénomène est dû en partie aux entreprises qui ont vu dans l'Internet l'outil par excellence pour se faire connaître. Mais également aux organismes ayant vocation à fournir un service public d'information, de proximité ou non.
- ***En 1993*** : Les entreprises et les états ont exploité l'Internet et en particulier les serveurs Web pour faire le commerce électroniques ou bien les gouvernements électroniques.
- ***En 1998*** : Les Services Web prennent leur origine dans l'informatique distribuée et dans l'évènement du Web.

III. Définitions

Définition 01

Le consortium W3C définit un Service Web comme étant une application ou un composant logiciel qui vérifie les propriétés suivantes :

- Il est identifié par une URI.
- Ses interfaces et ses liens (binding) peuvent être décrits en XML.
- Sa définition peut être découverte par d'autres Services Web.
- Il peut interagir directement avec d'autre Services Web à travers le langage XML et en utilisant les protocoles d'Internet. **[32]**

Définition 02

Selon IBM : A Web Service is a Web Application

« Web Services are a new breed of Web application. They are self-contained, self-describing, modular applications that can be published, located, and invoked across the Web. Web Services perform functions, which can be anything from simple requests to complicated business processes... Once a Web Service is deployed, other applications (and other Web Services) can discover and invoke the deployed Service. » **[33]**

IV. Les caractéristiques des Services Web

Web based : les Services Web sont basés sur les protocoles et les langages du Web, en particulier HTTP et XML (tout comme le Web lui-même s'appuie sur les protocoles d'Internet en particulier TCP/IP : c'est une « couche » supplémentaire).

Self-described, self-contained : le cadre des Services Web contient en lui-même toutes les informations nécessaires à l'utilisation des applications sous la forme de trois fonctions : trouver, décrire et exécuter. Il est donc nécessaire pour faire fonctionner un cadre de Service Web de disposer d'un annuaire des applications disponibles, d'une description du fonctionnement de l'application, et d'avoir accès à l'application.

Modular **:** les Services Web fonctionnent de manière modulaire et cela signifie qu'au lieu d'intégrer dans une seule application globale toutes les fonctionnalités, on crée (ou on récupère) plusieurs applications spécifiques qu'on fait interpeler entre elles, et qui remplissent chacune une de ces fonctionnalités. **[34]**

- Les Services Web sont indépendants de toute plate-forme d'exécution et de tout langage de programmation, ceci offre donc un intérêt majeur et ce principe de fonctionnement permet à quiconque désirant utiliser ou développer un Service Web de le faire dans son propre langage.
- Ils servent principalement pour le développement d'applications distribuées.
- Leur fonctionnement repose sur un modèle en couches, dont les trois couches fondamentales sont les suivantes **[35] :**

  - **Invocation** : visant à décrire la structure des messages échangés par les applications.
  - **Découverte** : pour permettre de rechercher et de localiser un Service Web particulier dans un annuaire de services décrivant le nom de la société, l'objectif de chaque service, etc.
  - **Description :** dont l'objectif est la description des interfaces (paramètres des fonctions, types de données) des Services Web.

- Les communications des Services Web s'effectuent sur un support universel, maîtrisé et généralement non filtré par les pare-feux employant une syntaxe basée sur la notation XML pour décrire les appels de fonctions distantes et les données échangées organisant les mécanismes d'appel et de réponse.

On distingue au moins deux styles de communications :

- Le style RPC (Remote Procedure Call : appel de procédure à distance)
- Le style Document : communication sous la forme de documents XML auto-descriptifs.

Ainsi que trois modes de communication :

- Le mode RPC ou mode requête-réponse.
- Le mode "one-way messaging" ou mode requête simple.
- Le mode "asynchronous callback" ou mode requête-réponse asynchrone.

V. Cycle d'utilisation des Services Web

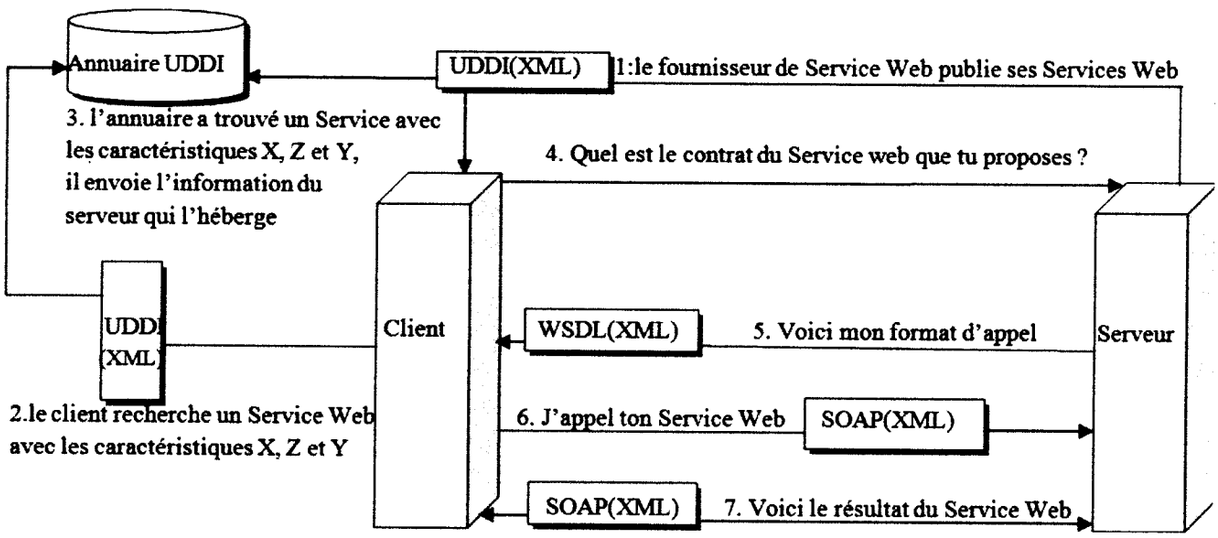

Figure I.1: Cycle d'utilisation des Services Web

- **Le fournisseur de Service :** crée le Service Web puis publie son interface ainsi que les informations d'accès au service dans un annuaire de Service Web.
- **L'annuaire de Service :** rend disponible l'interface du Service ainsi que ses informations d'accès pour n'importe quel demandeur potentiel de Service.
- **Le consommateur de Service :** accède à l'annuaire de Service pour effectuer une recherche afin de trouver les Services désirés, ensuite il se lie au fournisseur pour invoquer le Service.

- **Langage XML** (eXtend Mark-up Language) : décrit les informations.
- **Protocole SOAP** (Simple Object Access Protocol) : exécute les Services à distance.
- **Langage WSDL** (Services Web Description Language) : décrit l'interface des Services.
- **La Norme UDDI** (Universal Description, Discovery and Integration)**:** trouve les services dont nous avons besoin.

VI. L'intérêt des Services Web

Les technologies des Services Web peuvent être appliquées à toute sorte d'applications auxquelles elles offrent de considérables avantages en comparaison aux ancienne API propriétaires, aux implémentations spécifiques à une plate-forme et à quelque autres restrictions classiques que l'on peut rencontrer (multi plate-forme, multi-langage, disponible sur internet avec une information actualisée disponible en temps réel,...).

Les entreprises qui mettent à disposition leurs Services Web permettent aux développeurs intéressés par ses fonctionnalités de les réutiliser sans avoir à les recoder.

Le principe des Services Web permet d'avoir un partage des fonctionnalités et facilite grandement le développement,

Leur intérêt majeur est l'instauration :

- L'interopérabilité des applications
- L'intégration des applications sur le Web

VII. Les concepts de base des Services Web

Le concept de Web-Service veut simplement dire qu'un serveur propose des fonctions (des "**services**") que vous pouvez invoquer à distance à travers le **web** afin de déclencher un traitement sur le serveur distant et/ou obtenir une information de ce serveur. Ce concept n'est pas nouveau. Avant on appelait ça des RPC pour "Remote Procedure Calls". Outre les RPC du monde UNIX, il existe pour cela des standards ouverts en particulier CORBA et la réponse propriétaire de Microsoft: DCOM.

La seule nouveauté avec les Services Web, c'est que ces appels de procédures à distance se font comme des appels de page web, c'est à dire sur le protocole HTTP. "Je demande une URL, je reçois une page" devient "Je demande un service, je reçois une réponse". L'unique avantage de passer par HTTP est de ne pas nécessiter l'ouverture de ports spécifiques sur les firewalls.

Bien évidemment, les Services Web ont leurs propres standards, SOAP (Simple Object Access Protocol), WSDL (Simple Object Access Protocol), UDDI (Universal Description, Discovery and Intégration), etc.

Le protocole SOAP met à la disposition des services Web, un moyen standard de structuration et d'échange de messages XML. Il ne fournit en aucun cas une indication sur la structure que le message doit respecter vis à vis du service Web sollicité.

Le protocole WSDL offre une grammaire qui décrit l'interface des services Web de manière générique. Ces deux standards, SOAP et WSDL, définissent ensemble l'aspect le plus basique du développement de l'infrastructure des services Web.

Toutefois, dans un environnement ouvert comme Internet, le modèle de description des Services Web n'est d'aucune utilité s'il n'existe pas un moyen de localiser aussi bien les services que leurs descriptions WSDL. Un troisième standard a été conçu pour réduire l'écart entre les applications clientes et les services Web, appelé UDDI. **[01]**

VIII. Le protocole SOAP (Simple Object Access Protocol)

VIII.1. Définition

- Le protocole SOAP (Simple Object Access Protocol) **[36],** est le protocole qui permet le transport des messages générés par les Services Web.
- Il est indépendant du contenu du message et laisse la responsabilité de l'interprétation aux couches d'abstractions supérieures. Il est techniquement très proche de XML-RPC. La bibliothèque client encode les paramètres en XML et la bibliothèque serveur les décode. Le programmeur ne voit jamais de XML.
- Le programmeur SOAP dispose d'un grand nombre de bibliothèques : Perl, C, C#, Python, Ruby, Java, VisualBasic/.NET, PHP, Ada.
- Dans la pratique, le transfert est le plus souvent assuré via le protocole HTTP, cependant il peut aussi reposer sur d'autres protocoles.
- Il a l'intérêt de pouvoir être employé dans tous les types de communication.
- Les erreurs sont signalées par des exceptions (*faults*).
- **SOAP** constitue un standard simple et neutre puisqu'il n'impose pas l'utilisation d'une API particulière, ainsi qu'aucun modèle de programmation.

VIII.2. Généralité

La communication avec les Services Web s'effectue via le protocole SOAP qui peut être utilisé :

- Pour l'appel de méthodes (SOAP RPC).
- Pour l'échange de messages (SOAP Messaging).

SOAP permet l'interopérabilité entre différents systèmes d'exploitation et différentes plates-formes (J2EE, .NET, ...), Il est donc important d'avoir des règles de codage des types de données, afin que ces dernières puissent être encodées/décodées sans difficultés.

On distingue deux types de données :

- Les données de types **simples :** une chaîne de caractère par exemple.
- Les types **composés :** structures ou tableaux.

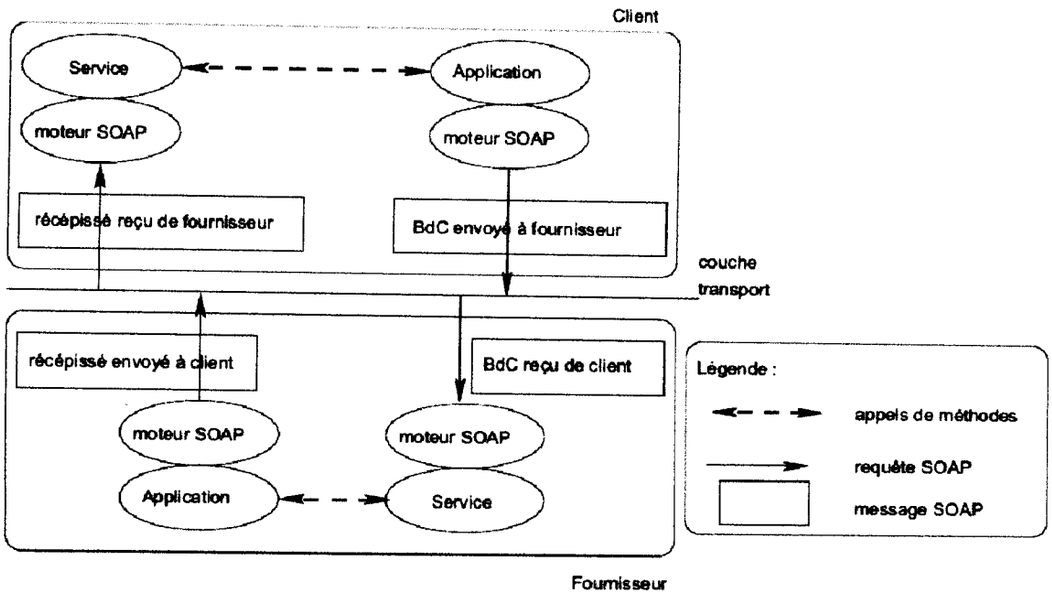

Figure I.2 : Interactions entre les partenaires Client et Fournisseur

Explication

La figure schématise un scénario d'interaction entre les partenaires Client et Fournisseur (les interactions avec l'entrepôt ne sont pas montrées).

- Le client transmet une commande au fournisseur en lui envoyant un bon de commande, ce dernier est transmis par une application du client, sous forme d'une requête SOAP.
- Chez le fournisseur, le serveur d'application reçcoit la requête (sous forme d'une requête HTTP) et la transmet au moteur SOAP.
- Ce dernier décode le message reçu et effectue les appels aux procédures correspondantes de l'application locale.
- Lorsque les vérifications de disponibilité des articles commandés et les interactions avec l'entrepôt sont terminées, l'application du fournisseur transmet,

La partie entête contient l'entête du protocole de transport (par exemple HTTP) ainsi que les métadonnées qui portent sur d'éventuelles propriétés non fonctionnelles du service (jeton de sécurité, contexte de transaction, certificat de livraison, etc.).

La partie corps regroupe quant à elle, les éléments métier tels que :

- Les appels de méthode, avec transferts de données spécifiques, dans le cadre d'une requête (le nom de la méthode ainsi que la valeur de ses paramètres).
- Seulement, les transferts de données spécifiques dans le cadre d'une réponse (la

- Les règles d'encodages SOAP qui définissent le mécanisme de sérialisation utilisé pour échanger les objets.
- SOAP RPC qui définit pour les utilisations synchrones une convention de représentation des appels et des réponses des procédures distantes.

VIII.3.1.1. Le HTTP Header

- Le protocole HTTP envoie une requête POST. L'entête HTTP se trouve juste avant le message SOAP, et définit le destinataire du message, les règles d'encodages HTTP,...
- Le champ SOAP Action peut être utilisé pour indiquer l'intention de la requête SOAP.
- Cette information peut être utilisée par un firewall pour filtrer les messages, ce champ est obligatoire mais peut être vide si on n'indique pas l'intention de la requête.

VIII.3.1.2. L'enveloppe SOAP

L'enveloppe contient l'espace de nommage définissant la version SOAP utilisée, et les règles de sérialisation et d'encodage.

VIII.3.1.3. Le Header SOAP

Cette partie du message est optionnelle, elle sert à transmettre des informations nécessaires pour l'exécution de la requête SOAP aux intermédiaires qui recevront le message. On y précise, généralement, des informations liées aux transactions, à l'authentification, etc.

Le Header est composé d'un ou plusieurs champs :

- L'attribut actor : désignant le destinataire du Header, il permet de préciser l'application à laquelle est destinée l'information contenue dans le Header.
- L'attribut mustUnderstand : indique si le processus est optionnel.

L'URI http://schemas.xmlsoap.org/soap/actor/next: précise, en particulier, que ces informations sont destinées à la première application qui reçoit le message. Dans le cas où l'actor n'est pas précisé, le Header est analysé par le destinataire final du message.

VIII.3.1.4. Le Body SOAP

Le body SOAP contient toutes les informations que l'on veut transmettre à l'application distante.

Le contenu du Body est normalisé dans SOAP RPC, pour modéliser une requête et sa réponse. Le Body de la requête contient l'identifiant de l'objet distant, le nom de la méthode à exécuter et les éventuelles paramétrés, le Body de la réponse contient le résultat de l'exécution de la requête.

VIII.3.2. SOAP RPC

Ce dernier définit les conventions permettant d'utiliser SOAP comme un RPC, et le format des messages pour effectuer une requête ou envoyer une réponse. SOAP RPC se base sur les spécifications de XML RPC.

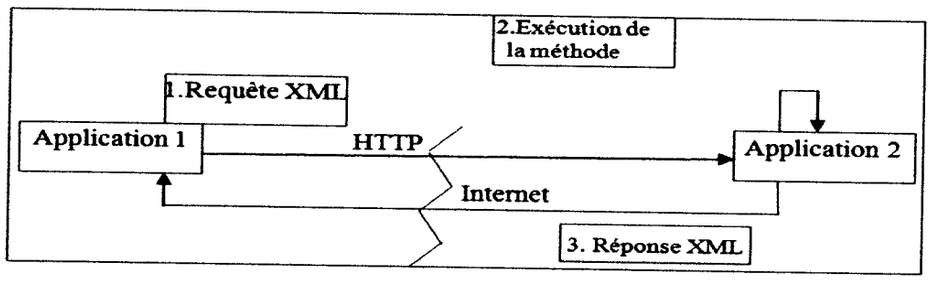

Figure I.4 : SOAP RPC

VIII.3.2.1. Préambule

On appelle service SOAP, une application dont les méthodes sont accessibles via SOAP. Ce service peut être développé dans n'importe quel langage de programmation. Un service est accessible via un identifiant unique de type URN.

En java, toute classe peut être rendue accessible via SOAP (en utilisant Apache SOAP par exemple). Lors de l'appel d'une méthode sur le service déployé, c'est la méthode correspondante de la classe Java qui est exécutée. Nous donnons un exemple de classe Java : « **calculatrice.Java**» que l'on désire rendre accessible via SOAP avec l'identifiant « urn » : **calculatrice**.

```
Imort java.*;
Public class calculatrice {
Public int plus (int a, int b)
{Return a+b ;}
Public int moins (int a, int b)
{Return a-b ;} }
```

VIII.3.2.2. Requête/réponse SOAP

Pour appeler l'opération « plus », on précise dans le body SOAP de la requête l'identifiant de l'objet distant, l'opération à exécuter et les éventuels paramètres:

<SOAP-ENV: Body>

<m: plus xmlns:m="urn.Calculatrice">

7

9

</m: plus>

<SOAP-ENV: Body>

Explication de la requête:

- <m/ plus xmlns:m="urn:Calculatrice"> : l'espace de nommage défini, à pour URN l'identifiant du service («urn : Calculatrice») le nom de l'élément (plus) correspond au nom de la méthode à exécuter.
- Les paramètres d'appel de la méthode sont ensuite ajoutés les uns à la suite des autres.

Dans notre exemple, on a deux paramètres a=7 et b=9.

Le message SOAP est alors envoyé au service SOAP, ce dernier exécute la méthode précisée dans la requête. Dans notre exemple, c'est la méthode «plus », ensuite il retourne un message SOAP dont le Body contient le résultat de l'opération.

<SOAP-ENV : Body>

<m :plus xmlns :m= "urn :Calculatrice">

<return type="int">16</return>

</m : plus>

</SOAP-ENV : Body>

Le résultat est contenu dans un élément **<return>** :

- L'attribut type précise le type de retour de la méthode exécutée.
- Le contenu de l'élément est le résultat de l'exécution de la méthode.

VIII.2.2.3. La gestion des erreurs

En cas d'erreur lors du traitement de la requête, le serveur renvoie un message SOAP donnant les raisons de l'erreur, dans un message HTTP dont le header commence par : http /1.1 500 Server Error.

L'erreur est détaillée dans le Body SOAP, dans un élément FAULT, donnant :

- **Faultcode :** le code de l'erreur, destiné à un traitement informatique.
- **Faultstring :** une explication textuelle, à destination des opérateurs humains.
- **Faultactor (optionnel):** en cas d'erreur dans le transport, l'acteur mis en cause peut être précisé : firewall, serveur, proxy, etc.
- **Détail (optionnel) :** un détail de l'erreur (par exemple en java la trace de l'exception est renvoyée).

Par exemple, dans le cas où la signature de la méthode de la requête ne correspond pas à la signature de la méthode du service, c'est-à-dire au lieu de la méthode «**plus** », on met par exemple «**plu**», la réponse SOAP sera comme suit :

<SOAP-ENV: Body>

<SOAP-ENV: Fault>

<faultcode>SOAP-ENV : Client< /faultcode>

<faultstring>

METHODE SIGNATURE DOES NOT MATCH

</faultstring>

</SOAP-ENV: Fault>

</SOAP-ENV: Body>

IX. L'Interface WSDL (Web Service Definition Language)

IX.1. Définition

Pour utiliser un Services Web, il est nécessaire d'en connaître la définition. Le langage WSDL (Web Services Definition Language) **[37]** décrit l'interface au service. En utilisant XML Schéma, WSDL défini les paramètres d'entrée et de retour d'un appel au service Web.

- La spécification WSDL présente les services comme des boîtes noires et s'intéresse à fournir une abstraction fonctionnelle du service. Elle joue un rôle important dans l'interopérabilité des Services Web et permet de définir ce qui est nécessaire à leur invocation.
- En réalité, la notion d'invocation de service est un abus de langage car ce n'est pas le service lui-même qui est invoqué mais bien une opération de ce service.
- La spécification WSDL est définie selon une sémantique totalement indépendante du modèle de programmation de l'application. Elle sépare clairement la définition abstraite du service (échange de messages) de ses mécanismes de liaison (définition des protocoles applicatifs). Cette dernière caractéristique permet d'interagir avec un service même si ce dernier a été modifié, ce qui est un point important pour assurer l'interopérabilité des services.
- La complétude de la spécification WSDL permet l'automatisation du processus d'invocation. En effet, elle contient toutes les informations nécessaires pour la mise au point d'un ensemble d'interfaces (API) qui génèrent automatiquement un programme client pour l'invocation d'un service Web.

IX.2. Structure d'un document WSDL

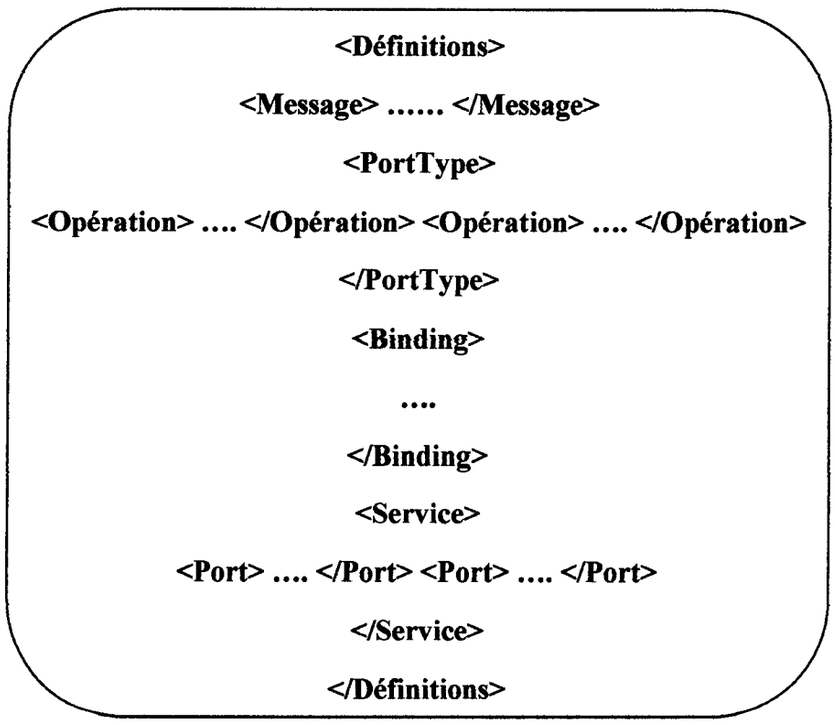

Figure I.5 : Structure d'un document WSDL.

- **<définitions> :** cet élément contient la définition du service, c'est la racine de tout document WSDL. Cette balise peut contenir les attributs précisant le nom du service, et les espaces de nommage.
- **<définitions>** contient trois types d'éléments :

➢ **<message>** et **<portType> :** ces éléments définissent les opérations offertes par le service, leurs paramètres d'entrée et de sortie, etc.

En particulier, un **<message>** correspond à un paramètre d'entrée ou de sortie d'une **<opération>**.

Un **<portType>** définit un ensemble d'opérations, une <**opération**> définit un couple message-entrée/message-sortie. Par exemple : dans le monde Java, une opération est une méthode et un portType est une interface.

➢ **<binding> :** cet élément associe les **<portType>** à un protocole particulier. Les binding possibles sont SOAP, CORBA ou DCOM, actuellement seul SOAP est utilisé. Il est possible de définir un Binding pour chaque protocole supporté.

➢ **<service> :** cet élément précise les informations complémentaires nécessaires pour invoquer le service, et en particulier l'URI du destinataire. Un **<service>** est modélisé comme une collection de ports, un <**port**> étant l'association d'un <Binding> à un URI.

Il est aussi possible de définir des types de données complexes dans une balise **<types>** juste avant la balise **<message>.**

Enfin, chaque élément WSDL peut être documenté à l'aide de l'élément <documentation>. Cet élément contient des informations liées à la compréhension du document par les utilisateurs humains du service.

WSDL est un système de communication « point à point ». Un consommateur du Service Web interroge le serveur, sur lequel celui-ci est disponible, et celui-ci retourne la description du Services Web demandé. **[38]**

L'exemple suivant est le fichier XML qui décrit le service «**Calculatrice**» précédemment défini :

```
1.  <? xml version= "1.0" encoding="UTF-8" ?>
2.  <wsdl : definitions targetNamespace="http://localhost:8080/axis/Calculatrice.jws">
3.  Xmlns : apachesoap="http://xml.apache.org/xml-soap"
4.  Xmlns : impl="http://localhost:8080/axis/calculatrice.jws"
5.  Xmlns : intf="http://localhost:8080/axis/calculatrice.jws "
6.  Xmlns : soapenc="http://schemas.xmlsoap.org/soap/encoding/"
7.  Xmlns : wsdl="http://schemas.xmlsoap.org/wsdl/"
8.  Xmlns : wsdlsoap="http://schemas.xmlsoap.org/wsdl/soap/"
9.  Xmlns : xsd="http://www.w3.org/2001/XML.Schema">
10. <wsdl : message name="moinsRequest">
11. <wsdl : part name="a" type="xsd :int"/>
12. <wsdl : part name="b" type="xsd :int"/>
13. </wsdl : message>
14. <wsdl : message name="plusRequest">
15. <wsdl : part name="a" type="xsd :int"/>
16. <wsdl : part name="b" type="xsd :int"/>
17. </wsdl : message>
18. <wsdl : message name="moinsRequest">
19. <wsdl : part name="moinsReturn" type="xsd :int">
20. </wsdl : message>
21. <wsdl : message name="plusRequest">
22. <wsdl : part name="plusReturn" type="xsd :int">
23. </wsdl : message>
24. <wsdl portType name="Calculatrice">
25. <wsdl : operation name="plus" parametrerOrder="a b">
26. <wsdl : input message="impl : plusRequest" name="plusRequest"/>
27. <wsdl : output message="impl : plusResponse" name="plusResponse"/>
28. </wsdl : operation>
29. <wsdl : operation name="moins" parametrerOrder="a b">
30. <wsdl : input message="impl : moinsRequest" name="moinsRequest"/>
31. <wsdl : output message="impl : moinsResponse" name="moinsResponse"/>
32. </wsdl : operation>
33. </wsdl : portType>
34. <wsdl : binding name="CalculatriceSoapBinding" type="impl:Calculatrice">
35. <wsdlsoap : binding style="rpc"transport="http://schemas.xmlsoap.org/soap/http"/>
36. <Wsdl : operation name="plus">
37. <wsdlsoap : operation soapAction=" "/>
38. <wsdl : input name="plusRequest">
39. <wsdlsoap : body encodingStyle="http://schemas.xmlsoap.org/soap/encoding"
40. namespace="http://DefaultNamespace" use="encoded"/>
41. </wsdl : input>
42. <wsdl : output name="plusResponse">
43. <wsdlsoap : body encodingStyle="http://schemas.xmlsoap.org/soap/encoding"
44. Namespace="http://localhost:8080/axis/Calculatrice.jws" use="encoded"/>
45. </wsdl : output>
46. </wsdl : operation>
47. <Wsdl : operation name="moins">
48. <wsdlsoap : operation soapAction=" "/>
49. <wsdl : input name="moinsRequest">
```

```
50. <wsdlsoap : body encodingStyle="http://schemas.xmlsoap.org/soap/encoding"
51. Namespace="http://DefaultNamespace" use="encoded"/>
52. </wsdl : input>
53. <wsdl : output name="moinsResponse">
54. <wsdlsoap : body encodingStyle="http://schemas.xmlsoap.org/soap/encoding"
55. Namespace="http://localhost:8080/axis/Calculatrice.jws" use="encoded"/>
56. </wsdl : output>
57. </wsdl : operation>
58. </wsdl : binding>
59. <wsdl : service name="CalculatriceService">
60. <wsdl : port binding="impl:CalculatriceSoapBinding" name="Calculatrice">
61. <wsdlsoap : address location="http://localhost:8080/axis/Calculatrice.jws"/>
62. </wsdl : port>
63. </wsdl : service>
64. </wsdl : definitions>
```

Expliquons chaque partie du document WSDL précédent :

- **(10-23) :** la première partie du fichier : définit les paramètres d'entrée (Request) et de sortie (Response) de toutes opérations du service :

  - **moinsRequest** et **moinsResponse** pour l'opération moins.
  - **plusRequest** et **plusResponse** pour l'opération plus.

- **(24-33) :** la définition abstraite du Service Web est faite par la définition du portType, qui encapsule les définitions des opérations **plus** et **moins**.

On fait ici référence aux messages définis précédemment (paramètres d'entrée et de sortie de l'opération). On obtient une description abstraite du service, indépendante de tout protocole de communication. C'est l'interface du service définissant les méthodes exportées, et leurs paramètres d'entrée et de sortie.

- **(34-58) :** il est ensuite possible d'associer ce service à un protocole existant par la définition d'un *Binding* (dans ce cas le protocole SOAP). Le *Binding* définit les paramètres d'invocation du service spécifique au protocole utilisé.

On définit ici les paramètres nécessaires à l'utilisation du service via SOAP (lien vers les spécifications du transport utilisé, règles d'encodage pour la sérialisation des messages échangés, etc.).

- **(59-64) :** la définition du service se termine par la définition des paramètres restants.

Par exemple pour un Binding SOAP, il reste à définir l'adresse URL du service à invoquer. Notons qu'il est possible de définir plusieurs Bindings, et d'associer ces Bindings à plusieurs URL, en utilisant la même définition abstraite du service.

X. L'annuaire UDDI (Universal Description, Discovery and Integration Service)

X.1. Définition

Pour utiliser un Service Web, il faut premièrement savoir qu'il existe. UDDI (Universal Description, Discovery and Integration Service) **[39]** est la norme qui définit le mécanisme pour découvrir dynamiquement des Services Web.

- La spécification UDDI constitue une norme pour les annuaires de Services Web: les registres UDDI. Les fournisseurs disposent d'un schéma de description permettant de publier des données concernant leurs activités, la liste des services qu'ils offrent et les détails techniques sur chaque service.
- La spécification UDDI offre aussi une API aux applications clientes, pour consulter et extraire des données concernant un service et/ou son fournisseur.
- En effet, les registres UDDI sont eux-mêmes exposés comme des Services Web. La mise en place d'un registre UDDI suit un processus uniforme imposé par la spécification. Chaque organisation qui veut mettre en place un registre UDDI doit suivre ce processus pour devenir un opérateur UDDI. Les registres UDDI crées sont organisés en réseaux, ils partagent ainsi les différentes informations publiées.
- La publication d'un service chez un opérateur donne automatiquement lieu à un processus de propagation des informations aux différents registres UDDI. L'accès à l'ensemble des informations des registres peut se faire de n'importe quel opérateur UDDI.

X.2. Le principe d'annuaire

UDDI (*Universal Description,Discovery and Integration*) permet de classer et de rechercher des Services Web. Si l'accès a l'information est très élevé, le recours aux Services Web ne devient plus intéressant. C'est pourquoi le principe d'annuaire universel a été mis en place.

Les annuaires UDDI ne répondent pas aux mêmes besoins que les annuaires de type LDAP (Lightweight Directory Access Protocol) dont la vocation est de référencer aussi bien des personnes que des ressources matérielles ou logicielles et de gérer les droits d'accès à ces ressources. **[38]**

L'API (Application Programming Interface) UDDI est divisée en une interface de programmation pour l'enregistrement de Services Web dans un annuaire UDDI et une interface de programmation pour la recherche d'informations.

Cette API est composée de deux grandes bibliothèques :

- API de requête
- API de publication

X.3. La recherche d'un Service Web

La recherche d'un Service Web se fait grâce à un moteur de recherche intégré au site de l'operateur UDDI choisi. Ce moteur de recherche vous permettra d'affiner votre recherche selon plusieurs critères :

- Nom de l'entreprise
- Localisation de l'entreprise
- Identifiant de l'entreprise
- Nom du Service Web

L'API de requête regroupe les appels aux sites operateurs qui n'ont pas besoin d'authentification particulière.

Les annuaires UDDI ont pour but de localiser virtuellement des Services Web hébergés sur les serveurs du monde entier. Lors de votre recherche vous pouvez ainsi vous renseigner sur tous les services mis à disposition d'une entreprise, avoir des informations sur l'entreprise. Les operateurs UDDI vous certifient la sécurité et l'intégrité des Services Web contenus dans leurs annuaires. Les appels aux sites des operateurs donnent accès aux différents éléments de l'enregistrement d'un Service Web dans un annuaire UDDI **[38]** :

- ***Find_binding*** : récupère la liaison du service considéré.
- ***Find_business*** : récupère l'identité de l'entreprise productrice du Service Web.

- ***Find_relatedbusiness*** : récupère la liste des entreprises étant reliées (filiale, département, partenaire,...) à l'entreprise productrice du Service Web.
- ***Find_service*** : récupère la définition du service.
- ***Find_tmodel*** : récupère le model de données associé.
- ***Get_bindingDetail*** : récupère par une liaison précédemment établie par ***Find_binding*** les champs individuels.
- ***Get_businessDetail, get_bisinessDetailExt*** : récupère une entité précédemment établie par ***find_business*** les attributs individuels.
- ***Get_serviceDetail*** : récupère un service précédemment établi par ***find_service*** les attributs individuels du service (prototypes des méthodes).
- ***Get_tmodelDetail*** : récupère un modèle établie par ***find_tmodel*** les champs individuels

X.4. La publication d'un Service Web

Comme dans WSDL, la liaison UDDI regroupe, pour un protocole de communication donné, les données techniques nécessaires à l'exploitation du Service Web (adresse IP, noms de domaines, les informations sur les modalités d'usage du service,...).

La publication d'un Service Web par une entreprise requière que celle-ci s'authentifie auprès du site de l'operateur UDDI. L'entreprise doit s'enregistrer chez l'operateur si cela n'est pas déjà le cas.

Une fois le site de l'operateur choisi, les modifications ultérieures ou la mise à jour de cet enregistrement doivent être faites apurées du même operateur. Lors de l'enregistrement, vous pouvez enregistrer simultanément un ensemble d'entreprises affiliées, des relations entres entreprises indépendantes décrivant des accords croisés.

L'API se décompose en trois groupes :

- La manipulation (save et delete).
- L'authentification des commandes par jeton (*get_authToken* et *discard_authToken*).
- L'ajout de relations inter entreprise (*joint_ventures)* **[38]**

X.5. Structure UDDI

UDDI est une spécification définissant la manière de publier et de découvrir les Services Web sur un réseau. Ainsi, lorsqu'on veut mettre à disposition un nouveau

service, on crée un fichier appelé **Business Registry** qui décrit le service en utilisant un langage dérivé de XML suivant les spécifications UDDI.

Les informations qu'il contient peuvent être séparées en trois types :

- **Les pages blanches :** incluant l'adresse, le contact et les identifiants relatifs aux Services Web.
- **Les pages jaunes :** identifiant les secteurs d'affaires relatifs aux Services Web.
- **Les pages vertes :** donnant les informations techniques.

En utilisant l'API UDDI l'utilisateur peut alors stocker ces informations dans un nœud (node) UDDI. Elles sont ensuite répliquées de nœud en nœud (principe assez proche dans l'idée des réplications de DND). Une fois ceci est fait, le Service Web peut alors être connu de tous ceux qui le recherchent. **[30]**

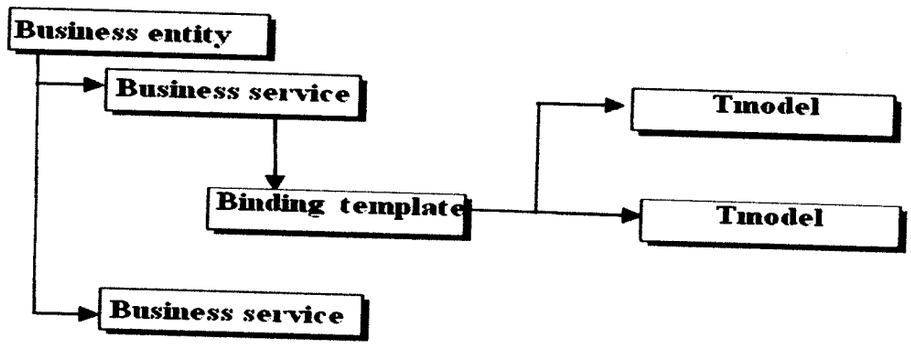

Figure I.6 : Une synthèse sur les informations constituant l'annuaire

- **BusinessEntity :** chaque service appartient à une organisation qui est identifiée par un nom, une description, des personnes de contact, une ou plusieurs catégories, etc.
- **BusinessService :** inclut les informations non techniques relatives au service, comme le nom et une brève description.
- **BindingTemplate :** définit ou (URL) et comment (protocole) accéder au service.
- **TModel :** comprend des liens vers les informations techniques du service, comme par exemple un fichier WSDL (Services Web Description Language) décrivant les différentes méthodes disponibles.
- **Publisher Assertion :** ensemble d'informations contractuelles entre les partenaires

X.6. Six types d'annuaires UDDI

Steve Graham **[40],** différencie six types d'UDDI :

1. **L'operateur UDDI** : c'est ce dont on parle dans la presse quand on parle d'UDDI aujourd'hui : un annuaire globale distribué sur internet, et accessible par tous.

Parmi les opérateurs UDDI on compte IBM, Microsoft et HP.

2. **L'UDDI e_marketplace** : cet annuaire UDDI regroupe des informations spécifiques à un corps de métier. Les entreprises d'un même corps de métier s'inscrivent et enregistrent leurs services. On y retrouve aussi les taxinomies spécifiques (code produit, catégories,...). Cette spécification de l'annuaire facilite sa mise en place pour des entreprises ayant des besoins spécifiques.

3. **L'UDDI portail** : cet annuaire se situe au niveau du firewall d'une entreprise et est accessible de l'extérieur. C'est un moyen pour l'entreprise de publier ses services à destination de ses partenaires, tout en gardant un contrôle total des informations publiées.

4. **L'UDDI catalogue partenaire ou (UDDI B2B)** : cet annuaire UDDI est interne à une entreprise, et n'est accessible que par les applications de cette entreprise. On y ajoute toutes les informations contenant ses partenaires et les services qu'ils proposent. Lorsqu'une entreprise désire travailler avec un nouveau partenaire, elle examine les services proposés dans un UDDI portail et ajoute les informations les plus utiles dans son UDDI catalogue partenaire. Ces informations sont utilisées par les applications de B2B de l'entreprise.

5. **L'UDDI EAI** : cet annuaire UDDI est lui aussi interne à une entreprise, mais les informations contenues servent avant tout à faire de l'intégration d'application en interne. On y retrouve les services publiés par les applications de l'entreprise, et des informations sur les rôles de l'organisation. Les informations contenues dans cet annuaire permettent aux applications d'une même entreprise de se découvrir et de collaborer.

6. **L'UDDI de test** : cet annuaire UDDI est coupé des autres car les informations contenues ne sont pas consistantes avec celles des autres annuaires. Il est utile lors des phases de tests qui précédent le déploiement des services.

Plus qu'un annuaire global, UDDI devient alors un outil incontournable pour faciliter l'intégration que ce soit pour des problématiques EAI (UDDI EAI) ou B2B (UDDI partenaire, UDDI e_marketplace, operateur UDDI).

XI. Les Services Web, CORBA et RMI

CORBA, acronyme de **Common Object Request Broker Architecture,** est une architecture logicielle pour le développement de composants et d'***Object Request Broker*** ou ORB. Ces composants qui sont assemblés afin de construire des applications complètes, peuvent être écrits dans des langages de programmation distincts, être exécutés dans des processus séparés, voire être déployés sur des machines distinctes.

CORBA est un standard maintenu par l'Objet Management Groupe. **[41]**

Remote Method Invocation, plus connu sous l'acronyme **RMI,** est une interface de programmation (API) pour le langage Java qui permet d'appeler des méthodes distantes.

L'utilisation de cette API nécessite l'emploi d'un registre RMI sur la machine distante hébergeant ces objets que l'on désire appeler au niveau duquel ils ont été enregistrés. Cette interface de programmation est très souvent utilisée en parallèle avec l'API d'annuaire JNDI ou encore avec la spécification de composants distribués transactionnels EJB du langage Java. **[42]**

Voici un tableau comparatif entre les Services Web, CORBA et RMI

	Service Web	CORBA	RMI
Qui	3WC	OMG	SUN
Description	WSDL	IDL	Interface Java
Nommage	UDDI	Naming Service	Remote Registry
Proxy	SOAP	Stubs	Stubs/Skeleton
Données	XML	CDR	Sérialisation Java
Transport	HTTP	GIOP/IIOP	JRMP
Réseau	TCP/IP	TCP/IP	TCP/IP
Sécurité	Pas fini	Oui	Possible SSL
Interopérabilité	Oui	Oui	Java
Spécificité	Web++	Service++	Classloading

XII. Quelques domaines d'application des Services Web

- Les Services Web peuvent être utiles dans la plupart des scénarios applicatifs lorsque la communication peut être établie sur un modèle bidirectionnel (requête/réponse).
- L'application des Services Web est multiple, autant dans les domaines du B2B, B2C que pour les domaines de gestion de stock, etc.
- B2C (Business to Cosumer) : qualifie une application, un site internet destiné au grand public.
- B2B (Business to Business) : qualifie une application, un site internet destiné au commerce professionnel à professionnel.

XIII. Avantages et inconvénients

XIII.1. Avantages

- Les Services Web comportent de nombreux avantages ils sont utilisables à distance via n'importe quel type de plate-forme, ils peuvent servir au développement d'applications distribuées et sont accessibles depuis n'importe quel type de client.
- Les services web fournissent l'interopérabilité entre divers logiciels fonctionnant sur diverses plates-formes.
- Ils utilisent des standards et protocoles ouverts, ils engendrent un faible couplage entre applications.
- Les protocoles et les formats des données sont au format texte dans la mesure du possible, facilitant ainsi la compréhension du fonctionnement global des échanges.
- Basés sur le protocole HTTP, les Services Web peuvent fonctionner à travers de nombreux pare-feux sans nécessiter des changements sur les règles de filtrage.
- Les outils de développement, s'appuyant sur ces standards permettant la création automatique de programmes utilisant les Services Web existants. **[43]**
- Les Service Web permettent de réduire la complexité d'une application car le développeur peut se focaliser sur un service, indépendamment du reste de l'application.
- Les Services Web sont réutilisables.

XIII.2. Inconvénients

- Les normes de Services Web dans certains domaines sont actuellement récentes. Bien que s'appuyant sur des bases solides (SOAP et WSDL), la normalisation complète d'une architecture distribuée fondée sur les Services Web n'est pour le moment qu'un rêve annoncé chaque année.
- Les Services Web souffrent des performances faibles comparées à d'autres approches de l'informatique répartie telles que le RMI (Remote method invocation), CORBA (Common Object Request Broker Architecture), ou DCOM (Distributed Component Object Model).
- Par l'utilisation du protocole HTTP, les Services Web peuvent contourner les mesures de sécurité mises en place au travers des pare-feux. **[43]**
- Les transferts reposent sur le XML, ce qui pose un problème sur la taille des fichiers échangés. Les fichiers XML sont le plus souvent de très gros fichiers et ceci entrainera une lourdeur considérable.
- Les Services Web ont une faible adaptabilité aux changements de contexte.
- Ils ne sont pas sécurisés à 100%.

XIV. Conclusion

Le Service n'est autre qu'une transaction accessible par l'échange de documents XML entre deux URL. Son principal avantage réside dans l'instauration d'un dialogue direct entre applications.

Effectivement, ce dernier a été conçu pour faciliter les échanges de données ainsi que l'accès aux applications au sein des entreprises et entre les entreprises elles-mêmes.

l'objectif de ce paradigme est de transformer le Web en un dispositif distribué de calcul où les programmes peuvent interagir de manière intelligente en étant capables de se découvrir automatiquement, de négocier entre eux, et de se composer en des services plus complexes.

En effet, On doit prendre en compte aussi lors de la description de ses Services Web des aspects sémantiques, le chapitre suivant présente une solution idéale pour assurer cet objectif

CHAPITRE II

LES ONTOLOGIES

I. Introduction

Au fur et à mesure des expérimentations, des méthodologies de construction d'ontologie et des outils de développement adéquats sont apparus. Emergeant des pratiques artisanales initiales, une véritable ingénierie se constitue autour des ontologies, ayant pour but la construction mais plus largement leur gestion tout au long d'un cycle de vie.

Les ontologies proposent une compréhension commune et partagée d'un domaine, tant au niveau des utilisateurs humains qu'au niveau des applications logicielles. Elles sont devenues un outil clé dans la représentation des connaissances, leurs applications sont nombreuses et font l'objet d'intenses travaux dans différents domaines dont les suivants :

1. La description de l'information dans différentes spécialités (Biologie, Médecine,...) Par exemple : The Gene Ontology (GO).
2. La recherche d'information.
3. Le commerce électronique afin de permettre, par exemple, la communication entre les fournisseurs et les acheteurs.
4. Les environnements de formation à distance

L'enjeu de l'effort engagé est de rendre les machines suffisamment sophistiquées pour qu'elles puissent intégrer le sens des informations, qu'à l'heure actuelle, elles ne font que manipuler formellement.

Mais en attendant que des ordinateurs « bourrés » d'ontologies et de connaissances nous déchargent en partie du travail de plus en plus lourd de gestion des informations dont le flot a tendance à nous submerger, de nombreux problèmes théoriques et pratiques restent a résoudre.

Ce chapitre a pour but de présenter sans exhaustivité l'état de l'art en ingénierie otologique tout en mettant en lumière certaines des principales difficultés rencontrées dans cette discipline.

II. Définition

Il est difficile de définir ce qu'est une ontologie d'une façon définitive. Le mot est en effet employé dans des contextes très différents touchant la philosophie, la linguistique ou l'intelligence artificielle.

- Gruber **[10]** définit l'ontologie comme suit **:** « une ontologie est une spécification explicite d'une conceptualisation d'un domaine »
- Guarino **[13],** part de sept interprétations possibles pour chercher à clarifier ce qu'est une ontologie. Outre le sens philosophique originel, une ontologie désigne en effet ***une modélisation conceptuelle,*** *ou* ***une représentation de cette modélisation.*** Dans les deux cas, on parle **d'ontologie formelle** pour désigner aussi bien la rigueur de la modélisation que la structure de sa représentation.

Cela nous amène à la notion de convention ontologique, qui est une modélisation conceptuelle du sens du formalisme de représentation. A la différence d'un vocabulaire, une ontologie cherche à représenter le sens des concepts et des relations qui les lient.

- Le premier objectif d'une ontologie est de définir quelles primitives (avec leur sémantique associée) pour la représentation des connaissances dans un contexte donné.
- Les ontologies décrivent généralement :
  1. **Individus** : les objets de base.
  2. **Classe** : ensemble, collection, ou types d'objets.
  3. **Attributs** : propriétés, fonctionnalités, caractéristiques ou paramètres que les objets peuvent posséder et partager.
  4. **Relations** : les liens que les objets peuvent avoir entre eux.
  5. **Evénement** : changement subis par des attributs ou des relations.
- L'ontologie en tant que discipline se donne pour objectif de caractériser les différents modes d'existence des objets, suivant les espèces d'objet (Naturels, Artificiels, Esthétiques, etc.). On trouve bien ce souci de caractériser les concepts dans différents domaines. **[44]**

III. L'intérêt des ontologies

Selon les chercheurs de l'université de Stanford (USA), les ontologies sont créées pour :

- Partager une compréhension commune de la structure d'informations entre les personnes ou agents logiciels.
- Permettre la réutilisation des connaissances de domaine.
- Faire des hypothèses de domaine explicite.
- Séparer les connaissances du domaine de la connaissance opérationnelle.
- Analyser la connaissance de domaine.

IV. Les constituants d'une ontologie

IV.1. *Connaissances et domaines de connaissance*

Une ontologie ne peut être construite que dans le cadre d'un domaine précis de la connaissance, ne serait-ce que parce que beaucoup de termes n'ont pas le même sens d'un domaine à l'autre, et qu'une sémantique non ambigüe doit être intégrée à l'ontologie. **[02]**

Délimiter rigoureusement un domaine de connaissance peut par contre se révéler ardu, à cause de la nature holistique de la connaissance. Certaines connaissances, qui peuvent constituer en elles-mêmes un domaine, sont utilisées dans tous les autres domaines.

De plus, les connaissances humaines se déploient suivant plusieurs dimensions : des connaissances peuvent être développées non seulement sur la réalité mais également sur un domaine de connaissance. On parle alors de méta-connaissances, ou connaissances sur les connaissances. Dans le domaine de l'ingénierie des connaissances, le terme de connaissance a un sens forcement restreint : ne sont considérées que les connaissances (au sens large) susceptible d'être formalisées, c'est-à-dire les connaissances peu ou beaucoup techniques **[03] :** « *For knowledge-based systems, what exist is exactly that wich can be represented* » **[10].** En outre, la manipulation automatique de connaissances ne parait pertinente que dans le cas des connaissances objectives, c'est-à-dire dont le sens est l'objet d'un consensus large.

En conclusion, la construction d'une ontologie doit se faire à partir d'un champ de connaissances bien délimité par un objectif opérationnel clair, et portant sur des connaissances objectives dont la sémantique puisse être exprimée rigoureusement et formellement, les ontologies vont permettre de spécifier les connaissances d'un domaine, de façon aussi indépendante que possible du type de manipulation qui vont être opérées sur ces connaissances. Ces ontologies sont appelées **des ontologies de domaine,** puisqu'elles sont construites sur un domaine particulier de la connaissance. De nombreuses ontologies de domaine existent déjà, telles que MENELAS dans le domaine médical **[31],** ENGMATH pour les mathématiques **[11],** TOVE dans le domaine de la gestion des entreprises **[12],** etc.

IV.2. *Les concepts et les relations*

IV.2.1. Concept

Un concept peut représenter un **objet matériel, une notion, une idée [26].**

Un concept peut être divisé en trois parties : **un terme** (ou plusieurs)**, une notion** et **un ensemble d'objets.**

La notion : également appelée *intention* du concept, contient la sémantique du concept exprimée en termes de propriétés et d'attributs, de règles et de contraintes.

L'ensemble d'objets : également appelé *extension* du concept, regroupe les objets manipulés à travers le concept, ces objets sont appelés instances du concept.

Par exemple le terme « table » renvoie à la fois à la notion de table comme objet de type « meuble » possédant un plateau et des pieds, et à l'ensemble des objets de ce type. Les deux aspects d'un concept sont assez différents, en particulier par le fait que deux extensions peuvent ne pas être disjointes, alors que deux intentions s'excluent mutuellement par au moins une propriété. B.BACHIMONT utilise le terme de *concept formel* pour designer l'extension d'un concept et de *concept sémantique* pour désigner l'intention du concept. **[03].** Un concept est ainsi doté d'une sémantique référentielle (celle imposée par son extension) et d'une sémantique différentielle (celle imposée par son intention).

IV.2.2. Relation

Une relation permet de lier des instances de concepts, ou des concepts génériques, elles sont caractérisées par un terme (voir plusieurs) et une signature qui précise le nombre d'instances de concepts que la relation lie, leurs types et l'ordre des concepts, c'est-à-dire la façon dont la relation doit être lue. Par exemple, la relation « écrit » lie une instance du concept « personne » et une instance du concept « texte », dans cet ordre. Les relations sont organisées de manier hiérarchisée à l'aide de la propriété de subsomption.

IV.3. Les connaissances inférentielles

Décrire les connaissances en termes de concepts, de relations entre ces concepts et de propriétés sur ces concepts et relations ne suffit généralement pas pour atteindre

l'objectif opérationnel d'un Système à Base de Connaissances, car le modélisateur s'intéresse à la mise en œuvre de ces connaissances pour atteindre un but.

Il s'agit également de tirer au maximum parti de ce qui fait la spécificité du support informatique par rapport au support écrit traditionnel, c'est-à-dire son aspect dynamique. Comparé à l'oral qui n'offre pas de support, et à l'écrit traditionnel, un système informatique peut manipuler les connaissances pour en inférer de nouvelles. Ces connaissances opérationnelles peuvent être des faits, des règles, ou des contraintes. **[03]**

V. Les types d'ontologies

Indépendamment des formalismes retenus, on distingue différents niveaux d'ontologies selon le domaine modélisé et éventuellement les tâches pour lesquelles elles sont conçues **[05]**.

- **Les ontologies de domaine** ont un faisceau plus large, une bonne précision et ne sont pas propres à une tâche particulière. Elles sont réutilisables dans un domaine donné. Elles fournissent des vocabulaires au sujet des concepts dans un domaine et leurs relations au sujet des activités qui ont lieu dans ce domaine, et au sujet des théories et des principes élémentaires régissant ce domaine.
- **Les ontologies de tâches** fournissent un vocabulaire systématisé des termes employés pour résoudre des problèmes liés aux tâches qui peuvent être ou non du même domaine. Ces ontologies fournissent un ensemble de termes au moyen desquelles nous pouvons décrire généralement comment résoudre un type de problèmes. Elles incluent des noms génériques, des verbes génériques, des adjectifs génériques et d'autres dans les descriptions de tâches.
- **Les ontologies d'application** ont un domaine de validité restreint et correspondent à l'exécution d'une tâche. Ce type d'ontologie décrit des concepts qui dépendent à la fois d'un domaine particulier et d'une tache particulière. Elles seraient souvent des spécialisations à la fois des ontologies de domaine et des ontologies de taches (double spécialisation) et correspondraient aux rôles par les entités de domaine lorsqu'elles effectuent certaines activités.
- **Les ontologies supérieures (upper level ontologies)** représentent des concepts généraux comme l'espace, le temps ou la matière. Elles sont universelles. Les

concepts des trois autres types d'ontologie peuvent y faire référence. Exemples : SUMO, BFO

- **les ontologies de représentation** conceptualisent des primitives des langages de représentation des connaissances.

Voici un schéma qui représente les différents types d'ontologies selon leurs degrés d'abstraction vis-à-vis d'une tâche particulière ou d'un point de vue. Tel que les flèches représentent des relations de spécialisation

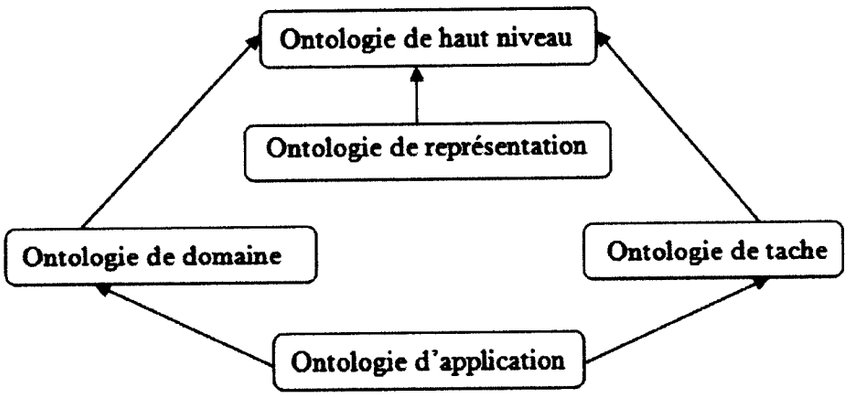

Figure II.1 : Différents types d'ontologie selon leur degré de d'abstraction [18]

VI. Langages de développement d'ontologie

Plusieurs langages d'ontologies ont été développés pendant les dernières années, et ils deviendront sûrement des langages d'ontologie dans le contexte du Web sémantique.

Certains d'entre eux sont basés sur la syntaxe de XML, tels que SHOE (Simple HTML Ontology Extension qui a été précédemment basé sur le HTML), RDF (Resource Description Framework), RDF Schéma. Les 2 derniers sont des langages créés par des groupes de travail du World Wide Web Consortium (W3C). Ainsi, que trois langages additionnels sont établis sur RDF(S) pour améliorer ses caractéristiques: OIL (Ontology Inference Layer), DAML+OIL et OWL (Web Ontology Language).

La figure ci-dessous présente des langages de spécification d'ontologie, ainsi que les rapports principaux entre tous ces langages sous la forme d'une pyramide des langages du Web sémantique. **[25]**

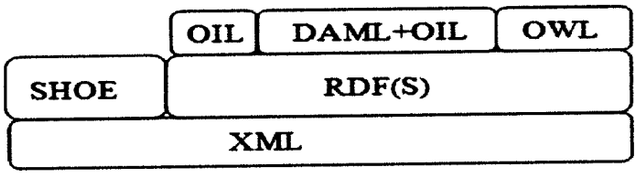

Figure II.2: La pyramide des langages basés Web.

VI.1. SHOE (Simple HTML Ontology Extension)

SHOE a été développé à l'Université de Maryland. Il a été créé comme prolongation d'HTML, incorporant la connaissance sémantique compréhensible par une machine dans des documents HTML ou d'autres documents Web. SHOE permet que les agents ramassent des informations significatives sur des pages Web et des documents ce qui améliore les mécanismes de recherche et de rassemblement de la connaissance. Ce processus se compose des trois phases: définir une ontologie, annoter les pages HTML avec l'information ontologique et avoir un agent pour rechercher sémantiquement l'information en recherchant toutes les pages existantes et en maintenant l'information mise à jour. **[47]**

VI.2. RDF(S) (Resource Description Framework and SDF Schema)

RDF a été développé par le W3C pour décrire des ressources Web. Il permet les spécifications des données sémantiques basées sur XML d'une façon normalisée et interopérable. Il fournit également des mécanismes pour représenter explicitement des services, processus, et des modèles d'affaires, tout en permettant l'identification d'information non explicite. Le modèle de données de RDF est équivalent au formalisme sémantique de réseaux. Il se compose de trois types d'objet: des ressources, des propriétés et des instructions.

Le modèle de données de RDF ne fournit pas des mécanismes pour définir les rapports entre les propriétés (attributs) et les ressources ceci est le rôle de RDFS. Ce dernier offre des primitives pour définir les modèles de la connaissance, il est largement utilisé comme format de représentation dans beaucoup d'outils et projets, comme Amaya, Protégé, Mozilla SilRI, etc. **[47]**

VI.3. OIL (Ontology Interchange Language)

OIL a été développé dans le projet d'OntoKnowledge**[46],** Il permet l'interopérabilité sémantique entre les ressources Web, sa syntaxe et sa sémantique sont basées sur les propositions existantes (OKBC, XOL, et RDF(S)).

OIL est construit sur RDF(S), il a les couches suivantes: OIL Core, OIL standard, OIL Instance, OIL Heavy. La syntaxe d'OIL est non seulement exprimée en XML mais peut également être présentée dans une forme textuelle.

VI.4. DAML+OIL (DARPA Agent Markup Language + OIL)

DAML+OIL a été développé par un Comité mixte des US et de l'union européenne (IST), DAML+OIL partage le même objectif que OIL. Il est construit sur RDF(S). Les spécifications initiales se sont appelées DAML-ONT et ont été également basées sur le langage d'OIL. OILed, OntoEdit, Protégé2000, et WebODE sont des outils qui peuvent rédiger des ontologies en DAML+OIL. **[47]**

VI.5. OWL (Web Ontology Language)

OWL peut être utilisé pour représenter explicitement des vocabulaires de termes et des rapports entre les entités dans ces vocabulaires. C'est une révision de DAML+OIL qui utilise la conception et l'application de DAML+OIL. Il n'existe pas d'outil qui soutienne OWL, mais on peut également utiliser des éditeurs de texte pour rédiger des ontologies en OWL. **[47]**

VII. Construction d'une ontologie

VII.1. Cycle de vie des ontologies

Les ontologies étant destinées à être utilisées comme des composants logiciels dans des systèmes répondant à des objectifs opérationnels différents, leur développement doit s'appuyer sur les mêmes principes que ceux appliquées en géni logiciel.

En particulier, les ontologies doivent être considérées comme des objets techniques évolutifs et possédants un cycle de vie qui nécessite d'être spécifié.

Les activités liées aux ontologies sont d'une part des activités de gestion de projet (planification, contrôle, assurance qualifiée), et d'autre part des activités de développement (spécification, conceptualisation, formalisation), s'y ajoutent des activités transversales de support telle que l'évaluation, la documentation, la gestion de la configuration.

Le cycle de vie inspiré du géni logiciel comprend une étape initiale d'évaluation des besoins, une étape de construction, une étape de diffusion, et une étape d'utilisation. Après chaque utilisation significative, l'ontologie et les besoins sont réévalués et l'ontologie peut être étendue et si nécessaire en partie reconstruite.

La phase de conceptualisation peut être décomposée en 3 étapes :

- ✓ **Conceptualisation**
- ✓ **Ontologisation**
- ✓ **Opérationnalisation**

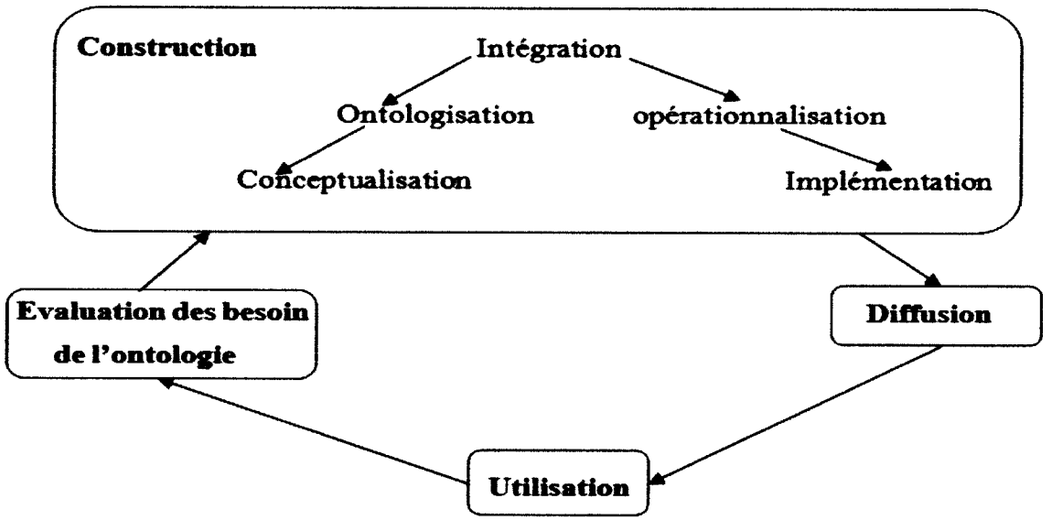

Figure II.3 : Cycle de vie d'une ontologie

VII.2. Les méthodologies de construction d'ontologies

Bien qu'aucune méthodologie générale n'ait pour l'instant réussi à s'imposer, de nombreux principes et critères de construction d'ontologies ont été proposés. Ces méthodologies peuvent porter sur l'ensemble du processus et guider l'otologiste à toutes les étapes de la construction.

VII.2.1. l'évaluation des besoins

Le but visé par la construction d'une ontologie se décline en 3 aspects

- ➢ *L'objectif opérationnel :* il est indispensable de bien préciser l'objectif opérationnel de l'ontologie, en particulier à travers des scénarios d'usage.
- ➢ *Le domaine de connaissance :* il doit être délimité aussi précisément que possible, et découpé si besoin en termes de connaissances du domaine, connaissance de raisonnement, connaissances de haut niveau (communes à plusieurs domaines).
- ➢ *Les utilisateurs :* ils doivent être identifiés autant que faire se peut, ce qui permet de choisir en accord avec l'objectif opérationnel, le degré de formalisme de l'ontologie, et sa gradualité.

Une fois le but défini, le processus de construction de l'ontologie peut démarrer en commençant par la phase de conceptualisation.

VII.2.2. la conceptualisation

La conceptualisation consiste à identifier, dans un corpus les connaissances du domaine.

Cette conceptualisation peut se décomposer en deux étapes :

- *Le tri :* cette opération s'effectue entre connaissances spécifiques au domaine et celles qui, ne participent qu'a l'expression des connaissances du domaine. **[09]**
- *La description sémantique :* en indiquant, a priori en langage naturel, leurs instances connues, les liens qu'ils entretiennent entre eux, leurs propriétés. La description d'une primitive conceptuelle doit contenir des liens vers les parties du corpus qui mettent en évidence sa sémantique, ce qui permet au cas où une ambigüité sémantique demeure, de revenir au corpus. **[06]**

Une fois les ressources cognitives passées au travers du tamis de la conceptualisation, il convient de formaliser, au cours de la phase d'ontologisation le modèle conceptuel obtenu.

VII.2.3. l'ontologisation

Une formalisation partielle, respectant l'intégrité du modèle conceptuel, va permettre à cette étape de construire une ontologie proprement dite. Afin de respecter les objectifs généraux des ontologies, T.GRUBER propose 5 critères permettant de guider le processus d'ontologisation. **[10]**

De plus il faut bien voir que l'ontologisation est une traduction dans certain formalisme de connaissances exprimées a priori en langage naturel.

VII.2.4. l'opérationnalisation

L'opérationnalisation consiste à outiller une ontologie pour permettre à une machine, via cette ontologie de manipuler des connaissances du domaine. La machine doit donc pouvoir utiliser des mécanismes opérant sur les représentations de l'ontologie. Dans le cas où le langage d'ontologisation n'est pas opérationnel, il est nécessaire soit d'outiller ce langage dans la mesure du possible, soit de transcrire l'ontologie dans un langage opérationnel.

L'ontologie opérationnalisée est intégrée en machine au sein d'un système manipulant le modèle de connaissance utilisé via le langage opérationnel choisi. Mais avant d'être livrée aux utilisateurs, l'ontologie doit bien sur être testée par rapport au contexte d'usage pour lequel elle a été bâtie.

VIII. Critère d'évaluation d'une ontologie

D'après Gruber, 5 critères permettent de mettre en évidence des aspects importants d'une ontologie :

- **La clarté :** la définition d'un concept doit faire passer le sens voulu du terme, de manière aussi *objective* que possible (indépendante du contexte). Une définition doit de plus être complète (c'est-à-dire définie par des conditions à la fois nécessaires et suffisantes) et documentée en langage naturel.
- **La cohérence :** rien qui ne puisse être inféré de l'ontologie ne doit entrer en contradiction avec les définitions des concepts (y compris celles qui sont exprimées en langage naturel).
- **L'extensibilité :** les extensions qui pourront être ajoutées à l'ontologie doivent être anticipées. Il doit être possible d'ajouter de nouveaux concepts sans avoir à toucher aux fondations de l'ontologie.
- **Une déformation d'encodage minimale :** une déformation d'encodage a lieu lorsque la spécification influe la conceptualisation (un concept donné peut être plus simple à définir d'une certaine façon pour un langage d'ontologie donné, bien que cette définition ne corresponde pas exactement au sens initial). Ces déformations doivent être évitées autant que possible.
- **Un engagement ontologique minimal :** le but d'une ontologie est de définir un vocabulaire pour décrire un domaine, si possible de manière *complète* ni plus ni moins. Contrairement aux bases de connaissances par exemple, on n'attend pas d'une ontologie qu'elle soit en mesure de fournir systématiquement une réponse à une question arbitraire sur le domaine. Une ontologie est la théorie la plus faible couvrant un domaine ; elle ne définit que les termes nécessaires pour partager la connaissance liée à ce domaine. **[10]**

IX. Quelques outils de construction d'ontologies

De nombreux outils de construction d'ontologies utilisent des formalismes variés et offrant différentes fonctionnalités ont été développés. On va citer juste les plus significatifs, essentiellement ceux qui constituent des implémentations de méthodologies. Tous ces outils offrent des supports pour le processus d'ontologisation,

mais peu offrent une aide à la conceptualisation ils sont plus ou moins indépendants des formalismes de représentation.

IX.1. DOE (DIFFERENTIAL ONTOLOGY EDITOR)

Cet éditeur offre la possibilité de construire les hiérarchies de concepts et relations en utilisant les principes différentiels énoncés par B.BACHIMONT. Puis en ajoutant les concepts référentiels, la sémantique des relations est ensuite précisée par des contraintes. Ce n'est qu'une fois l'ontologie ainsi structurée qu'elle est formalisée en utilisant la syntaxe XML.

IX.2. ODE (ONTOLOGY DESIGN ENVIRONMENT)

Cet environnement permet de construire des ontologies au niveau connaissances, comme le préconise la méthodologie METHONTOLOGY. L'utilisateur construit son ontologie dans un modèle de type frame, en spécifiant les concepts du domaine, les termes associés, les attributs et leurs valeurs, les relations de subsomption. L'ontologie opérationnelle est alors générée en utilisant les formalismes ONTOLINGUA ou FLOGIC. WEBODE est l'application de ODE pour le Web.

IX.3. ONTOEDIT (ONTOLOGY EDITOR)

C'est un environnement de construction d'ontologies indépendant de tout formalisme. Il permet l'édition des hiérarchies de concepts et de relations et l'expression d'axiomes algébriques portant sur les relations, et de propriétés telles que la généricité d'un concept. Des outils graphiques dédiés à la visualisation d'ontologies sont inclus dans l'environnement. ONTOEDIT intègre un serveur destiné à l'édition d'une ontologie par plusieurs utilisateurs. Un contrôle de la cohérence de l'ontologie est assuré à travers la gestion des ordres d'éditions. Enfin, un plug-in nommé ONTOKICK offre la possibilité de générer les spécifications de l'ontologie par l'intermédiaire de questions de compétences.

IX.4. PROTEGE

PROTEGE est une interface modulaire permettant l'édition, la visualisation, le contrôle (vérification des contraintes) d'ontologies, l'extraction d'ontologies à partir de sources textuelles, et la fusion semi-automatique d'ontologies. Le modèle de connaissances sous-jacent à PROTEGE est issu du modèle des frames et contient des

[illegible] ontologies

Plusieurs ontologies ont été développées pour décrire des concepts généraux ou des domaines particuliers. Parmi les ontologies générales nous citons :

I. Introduction

La similarité sémantique, c'est-à-dire l'appréhension de la liaison entre deux concepts, est une capacité de l'homme que les machines ne savent que très mal reproduire. Ainsi, pour un humain, il est évident que les concepts de crayon et de papier sont liés, beaucoup plus que ceux de parapluie et fer à repasser en tout cas. Mais il est très difficile de le formaliser car rien, en surface, ne permet de le décider.

Pour ce faire, il faut utiliser des ressources sémantiques : les ontologies, c'est-à-dire des bases de connaissances. Elles seules permettent de montrer les liens (hypéronymie, antonymie, métonymie, etc.) entre des concepts.

Les recherches sur ce sujet se font sur plusieurs domaines : intelligence artificielle, psychologie, sciences cognitives. Les modèles de calcul de la similarité sémantique se retrouvent dans de multiples applications, avec pour but de donner à ces dernières des connaissances supplémentaires pour raisonner sur leurs données. On peut distinguer trois grandes familles d'approches pour l'identification de la similarité sémantique.

- Les approches basées sur les nœuds utilisant typiquement des mesures du contenu informationnel pour déterminer la similarité conceptuelle. En plus, la similarité entre les concepts est déterminée par le degré de partage de l'information.
- L'autre famille d'approches repose uniquement sur la hiérarchie ou sur les distances des arcs. Le principe de calcul de similarité avec cette approche est basé sur l'idée suivante : plus le chemin entre deux nœuds est court plus ils sont plus semblables. L'autre notion qui caractérise cette deuxième approche est que les arcs d'une taxonomie représentent des distances uniformes, par conséquent, cette approche présente l'inconvénient que tous les liens sémantiques possèdent le même poids ce qui impose des difficultés au niveau de la définition et du contrôle des distances des liens.
- Finalement l'approche hybride qui combine entre les deux premières approches. Avec cette approche, il y a plusieurs manières de déterminer la similarité conceptuelle de deux mots dans un réseau sémantique hiérarchique.

Il existe aussi d'autres mesures de similarités syntaxiques dont on va citer dans la suite mais notre travail a pour objectif de proposer une nouvelle mesure de similarité sémantique en se basant sur la mesure de *Wu et Palmer* pour pallier au problème cité plus bas afin d'obtenir des résultats réalistes pour des concepts non situés dans le même chemin.

II. Quelques mesures de similarités syntaxiques

II.1. Distance de Hamming

La distance de Hamming, doit son nom à Richard Hamming (1915 1998) **[14].** Elle est utilisée en *télécommunication* pour compter le nombre de bits altérés dans la transmission d'un message d'une longueur donnée. Elle est utilisée aussi en *informatique*, en *traitement du signal.* Elle joue un rôle important en théorie algébrique des codes correcteurs. Elle permet de quantifier la différence entre deux séquences de symboles. Cette dernière, est une distance au sens mathématique du terme. À deux suites de symboles de même longueur, elle associe le nombre de positions où les deux suites diffèrent.

Soit A un alphabet et F l'ensemble des suites de longueur n à valeur dans A. La distance de Hamming entre deux éléments a et b de F est le nombre d'éléments de l'ensemble des images de a qui diffèrent de celle de b.

Formellement, si $d(.,.)$ désigne la distance de Hamming :

$$\forall a, b \in F \quad a = (a_i)_{i\in[0,n-1]} \; et \; b = (b_i)_{i\in[0,n-1]} \quad d(a,b) = \#\{i : a_i \neq b_i\}$$

La notation $\#E$ désigne le *cardinal* de l'ensemble E.

Un cas important dans la pratique est celui des symboles binaires. Autrement dit A= {0,1}, On peut alors écrire, si $\oplus$ désigne le *ou exclusif.*

$$d(a,b) = \sum_{i=0}^{n-1} (a_i \oplus b_i)$$

Exemples

Considérons les suites binaires suivantes : a= (0 0 0 1 1 1 1) et b= (1 1 0 1 0 1 1)

La distance entre a et b est égale à 3 car 3 bits diffèrent.

II.2. Distance d'édition

La distance d'édition (ou distance de Levenshtein) entre deux chaines de caractères X=[x1...xT] et Y=[y1...yV], est donnée par le nombre minimum d'opérations qui nous à permis de passer de X à Y (ou de Y à X...).

Ces opérations peuvent être : la suppression (S), l'insertion (I), le changement (C).

Au lieu de compter le nombre d'opérations, on peut affecter un coût "C" différent à chacune, ce qui nous donne :

- C(xi,yj) est le coût de la substitution du caractère xi par yj.
- C(xi,λ) est le coût de délétion du caractère xi. λ représente un symbole vide.
- C(λ,yj) est le coût d'insertion du caractère yj.

Exemple :

Voici un exemple qui permet la transformation de aabcb en ababd. La figure suivante donne quelques chemins permettant de passer de aabcb à ababd. En prenant un coût de 1 pour chacune des transformations, la distance d'édition de aabcb à ababd est de trois.

Elle serait différente si le coût d'une suppression était de 0,5, celui d'une insertion de 0,75 et celui d'un changement de 0,25. Sa valeur serait alors de 1,5.

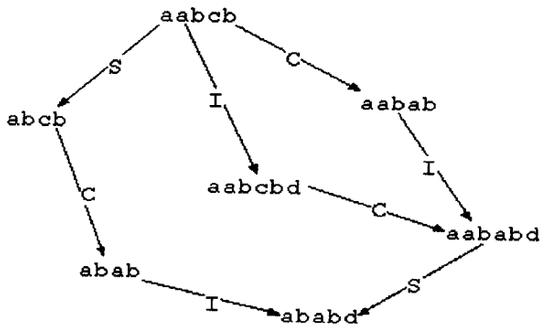

Figure III.1 : les chemins les plus courts permettant de passer de aabcb à ababd

La distance ***d*** entre deux chaînes *a* et *b* permet facilement de calculer le taux de similarité ***s(a,b)*** en fonction aussi de leurs longueurs (*l* (*a*) et *l* (*b*)) :

$$s(a,b) = \frac{1 - d(a,b)}{MAX(l(a), l(b))}$$

II.3. Distance de Jaro-Winkler

La distance de Jaro-Winkler mesure la similarité entre deux chaînes de caractères. Il s'agit d'une variante proposée en 1999 par William E. Winkler **[28]**, découlant de la distance de Jaro **[16]** qui est principalement utilisée dans la détection de doublons.

Plus la distance de Jaro-Winkler entre deux chaînes est élevée, plus elles sont similaires. Cette mesure est particulièrement adaptée au traitement de chaînes courtes comme des noms ou des mots de passe. Le résultat est normalisé de façon à avoir une mesure entre 0 et 1, le zéro représentant l'absence de similarité.

Distance de Jaro

La distance de Jaro entre chaînes s1 et s2 est définie par :

$$d_j = \frac{1}{3}\left(\frac{m}{|s_1|} + \frac{m}{|s_2|} + \frac{m-t}{m}\right)$$

Où **m** est le nombre de caractères correspondants et **t** est le nombre de transpositions.

Deux caractères identiques de s1 et de s2 sont considérés comme correspondants si leur éloignement (la différence entre leurs positions dans leurs chaînes respectives) ne dépasse pas :

$$\left\lfloor \frac{\max(|s_1|, |s_2|)}{2} \right\rfloor - 1$$

Le nombre de transpositions est obtenu en comparant le i-ème caractère correspondant de s1 avec le i-ème caractère correspondant de s2. Le nombre de fois où ces caractères sont différents, divisé par deux, donne le nombre de transpositions.

Distance de Jaro-Winkler

La méthode introduite par Winkler utilise un coefficient de préfixe p qui favorise les chaînes commençant par un préfixe de longueur ℓ(avec $\ell \leq 4$).

En considérant deux chaînes s1 et s2, leur distance de Jaro-Winkler dw est :

$$d_w = d_j + (\ell p(1 - d_j))$$

Où **dj** est la distance de Jaro entre s1 et s2, ℓest la longueur du préfixe commun (maximum 4 caractères) et **p** est un coefficient qui permet de favoriser les chaînes avec un préfixe commun. Winkler propose pour valeur p = 0.1

Exemple

Soit deux chaînes s1 MARTHA et s2 MARHTA. La table de correspondance est :

	M	A	R	T	H	A
M	1	0	0	0	0	0
A	0	1	0	0	0	0
R	0	0	1	0	0	0
H	0	0	0	0	1	0
T	0	0	0	1	0	0
A	0	0	0	0	0	1

$m = 6$ (nombre de 1 dans la table), $| s_1 | = 6, | s_2 | = 6$

Les caractères *correspondants* sont {M,A,R,T,H,A} pour s_1 et {M,A,R,H,T,A} pour s_2. En considérant ces ensembles ordonnés, on a donc 2 couples (T/H et H/T) de caractères *correspondants* différents, soit deux demi-transpositions. D'où

$$t = \frac{2}{2} = 1$$

La distance de Jaro est :

$$d_j = \frac{1}{3}\left(\frac{6}{6} + \frac{6}{6} + \frac{6-1}{6}\right) = 0.944$$

La distance de Jaro-Winkler devient avec : $p = 0.1$ et $\ell = 3$

$$d_w = 0.944 + (3 * 0.1(1 - 0.944)) = 0.961$$

II.4. Mesure de Cosinus

La similarité cosinus (ou mesure cosinus) permet de calculer la similarité entre deux vecteurs à n dimensions en déterminant l'angle entre eux. Cette métrique est fréquemment utilisée en fouille de textes. Soit deux vecteurs V1 et V3.

α : Est l'angle qui s'obtient par le produit scalaire et la norme des vecteurs :

$$\cos\alpha = \cos(\mathbf{V_1}, \mathbf{V_3}) = \frac{\mathbf{V_1} \cdot \mathbf{V_3}}{||\mathbf{V_1}|| \, ||\mathbf{V_3}||}$$

Tel que **||V1||** : la longueur de **V1** (Pythagore). Cette formule se généralise dans un espace à deux dimensions à L : V = {a1, ..., aj , ..., aL} V' = {b1, ..., bj , ..., bL}

$$\cos\alpha = \cos(\mathbf{V}, \mathbf{V'}) = \frac{\sum_{j=1}^{L} a_j b_j}{\sqrt{\sum_{j=1}^{L} a_j^2} \sqrt{\sum_{j=1}^{L} b_j^2}}$$

Comme l'angle est compris dans l'intervalle [0,π], la valeur π indiquera des vecteurs résolument opposés, π/2 des vecteurs indépendants (orthogonaux) et 0 des vecteurs **colinéaires** (vecteurs similaires). Les valeurs intermédiaires permettent d'évaluer le degré de similarité.

Exemple

La figure suivante illustre la représentation vectorielle avec les fréquences de mots des documents D1, D2 et D3. Tel que : D1 = je je vais, D2 = je je je je vais vais , D3 = je vais vais

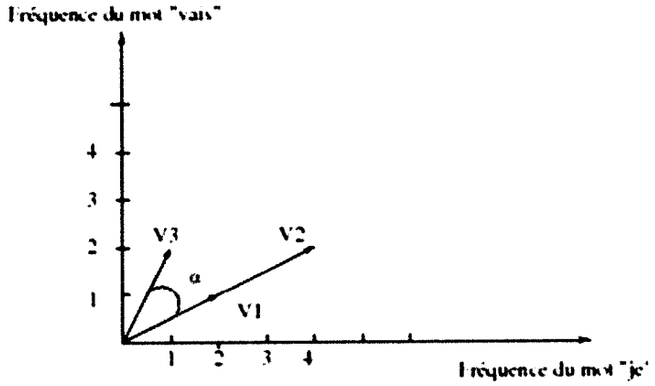

Figure III.2 : Représentation vectorielle complète des documents D1, D2 et D3.

II.5. Mesure de Jaccard

La mesure de similarité de Jaccard est définie par le nombre des objets communs divisé par le nombre total des objets moins le nombre d'objets communs :

$$s_j = \frac{m_c}{m1 + m2 - m_c}$$

Les vecteurs utilisés dans ce calcul de la similarité avec l'indice de Jaccard se fondent sur la présence/absence des mots. Ils n'utilisent pas les valeurs numériques et les fréquences, seulement l'absence ou la présence des mots, qu'on peut caractériser par un 0 (absence) ou par un 1 (présence). Les vecteurs D1, D2 et D3 correspondent au même vecteur booléen Vb1= Vb2 = Vb3 = {1,1}.

II.6. Similarité euclidienne

La similarité euclidienne est basée sur le ratio de la distance euclidienne augmenté de 1. La mesure de similarité est donc définie par :

$$Sime(X, Y) = \frac{1}{1+dE}$$

II.7. Similarité de Dice

La similarité de Dice **[20],** est définie par le nombre des objets communs multipliés par 2 sur le nombre total d'objets. La mesure de Dice est donc définie comme suit :

$$Simd(X, Y) = \frac{2 \cdot m_c}{m1 + m2}$$

III. Quelques mesures de similarités sémantiques

III.1. La classification des approches de mesure de similarité sémantique

Dans ce qui suit nous allons classifier les approches principales de la mesure de similarité en affectant des exemples pour chacune de ces approches. **[24]**

III.1.1. Approches basées sur les arcs

La mesure de similarité la plus intuitive des objets dans une ontologie est leurs distances **[27].** Evidemment, un objet X est plus similaire à un objet Y qu'un objet Z. Cette similarité est évaluée par la distance qui sépare les objets dans l'ontologie.

Ces mesures se servent de la structure hiérarchique de l'ontologie pour déterminer la similarité sémantique entre les concepts. Le calcul des distances dans l'ontologie est basé sur un graphe de spécialisation des objets. Dans chaque graphe, la distance de l'ontologie doit être caractérisée par le plus court chemin qui fait intervenir un ancêtre commun ou le plus petit généralisant, connectant potentiellement deux objets à travers des descendants communs.

Parmi les travaux classifiés sous cette bannière on peut citer:

III.1.1.1. Mesure de Wu & Palmer

La mesure de similarité de Wu et Palmer **[27],** est basée sur le principe suivant :

Etant donnée une ontologie W formée par un ensemble de nœuds et un nœud racine R (Figure3). Soit X et Y deux éléments de l'ontologie dont nous allons calculer la similarité. Le principe de calcul de similarité est basé sur les distances (N1 et N2) qui séparent les nœuds X et Y du nœud racine et la distance qui sépare le concept subsumant(CS) de X et de Y du nœud R.

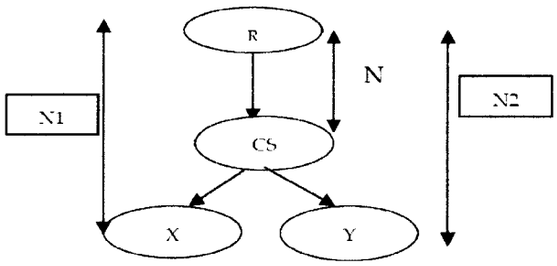

Figure III.3 : Le principe de calcul de Wu & Palmer.

La mesure de Wu et Palmer est définie par la formule suivante :

$$Simwp(X, Y) = \frac{2 * N}{N1 + N2}$$

Cette méthode a l'avantage d'être simple à calculer en plus des performances qu'elle présente, tout en restant aussi expressive que les autres, c'est pour cette raison qu'on a adopté cette mesure comme fondement de notre travail.

Cette mesure est intéressante mais présente un inconvénient car dans certaines situations, la valeur de similarité de deux éléments d'une ontologie contenus dans le voisinage dépasse la valeur de similarité de deux éléments contenus dans la même hiérarchie. Cette situation est inadéquate dans le cadre de la recherche de l'information

III.1.1.2. Mesure de Rada et al

Cette mesure **[22],** est adoptée dans un réseau sémantique et elle est fondée sur le fait qu'on peut calculer la similarité en se basant sur les liens hiérarchiques «is-a». Pour calculer la similarité de deux concepts dans une ontologie, on doit calculer le nombre des arcs minimums qui les séparent. Cette mesure, basée sur le calcul de la distance entre les nœuds par le chemin le plus court, présente un moyen des plus évidents pour évaluer la similarité sémantique dans une ontologie hiérarchique.

Cette meure est définie par la formule suivante :

$$Simrada(X, Y) = \frac{1}{1 + dist(X, Y)}$$

Tel que **dist(X, Y)** indique le nombre d'arcs minimum à parcourir pour aller d'un concept X à un concept Y

III.1.1.3. Mesure d'Ehrig et al.

Un travail de mesure de similarité pour les ontologies a été introduit par Ehrig et al **[08].** Ce travail introduit trois couches : les données, l'ontologie et le contexte. La similarité des entités est mesurée au niveau des données en considérant les valeurs de données de type simple ou complexe (entiers, caractères). Les relations sémantiques entre les entités sont mesurées au niveau de la couche de l'ontologie. Finalement la couche du contexte spécifie comment les entités de l'ontologie sont utilisées dans un certain contexte externe, plus spécifiquement, le contexte de l'application.

Ces mesures basées sur les arcs ont l'avantage d'être faciles à implémenter et peuvent donner une idée sur le lien sémantique entre les concepts. Cependant, elles ne prennent pas en compte le contenu du concept lui-même, ce qui peut conduire, dans certains cas, à une marginalisation de l'apport du concept en termes d'information.

III.1.2. Approches basées sur les nœuds

Ces approches adoptent une nouvelle mesure en termes de la mesure entropique de la théorie de l'information **[20]**. La probabilité P(.) pour l'identification de l'utilisation d'une classe ou de ses descendants dans un corpus désigne l'information de la classe. On définit l'entropie d'une classe par la formule suivante :

$$\mathbf{CI(x) = -log\ (P(x))}$$

P est la probabilité de trouver une instance du concept c. La probabilité d'un concept c est calculée en divisant le nombre des instances de c par le nombre total des instances. En associant des probabilités aux concepts d'une taxonomie, il est possible d'éviter le manque de fiabilité des distances des arcs. Cette caractéristique quantitative de l'information fournit une nouvelle façon de mesurer la similarité sémantique. Plus l'information est partagée par deux concepts, plus ils sont similaires.

Parmi les travaux, recensés dans la littérature, sous cette bannière on peut citer :

III.1.2.1. Mesure de Resnik

La notion du contenu informationnel (CI) a été initialement introduite par Resnik **[23]** qui a prouvé qu'un objet (mot) est défini par le nombre des classes spécifiées et que la similarité sémantique entre deux concepts est mesuré par la quantité de l'information qu'ils partagent.

Pour évaluer la pertinence d'un objet il faut calculer le contenu informationnel. Ce dernier est obtenu en calculant la fréquence de l'objet dans le corpus (Wordnet).

La formule proposée par Resnik est définie par:

$$\mathbf{Sim(X, Y)=Max\ [CI\ (CS(X, Y))]=Max\ [-log\ (P(CS(X, Y)))]}$$

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

III.1.2.2. Mesure de Lin

Lin a défini une mesure de similarité légèrement différente de celle de Resnik :

$$\mathbf{Siml} = \frac{\mathbf{2*log\ (P(CS(X,Y)))}}{\mathbf{log\ (P(X))+log(P(Y))}}$$

Cette mesure représente la similarité comme degré probabiliste de chevauchement des concepts descendants de X et Y.

III.1.2.3. Mesure de Jiang et Conrath

Pour remédier au problème présenté au niveau de la mesure de Resnik, **[17]** Jiang a apporté une nouvelle formule qui consiste à combiner l'entropie (contenu informationnel) du concept spécifique à ceux des concepts dont on cherche la similarité. La mesure adoptant cette méthode est basée sur la combinaison d'une source de connaissance riche (thesaurus) avec une source de connaissance pauvre (corpus).

Notons que cette formule est définie par l'inverse de la distance sémantique.

$$Sim(X, Y) = \frac{1}{distance(X, Y)}$$

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

$$distance(X, Y) = E(X) + E(Y) - (2.E(CS(X, Y)))$$

III.1.2.4. Mesure de Hirst et Onge

L'idée de cette mesure est que deux concepts lexicalisés sont sémantiquement étroits si leurs ensembles synonymes (synsets) de WordNet sont reliés par un chemin qui n'est pas trop long et qui "ne change pas la direction trop souvent". Avec cette mesure, toutes les relations contenues dans un réseau Wordnet sont prises en considération.

Dans le travail de Hirst **[15],** les auteurs ont classé la direction des liens en lien haut (super-classe), lien bas (sous-classe) et lien horizontal (antonyme).

Le calcul de la similarité, avec cette méthode, s'effectue entre objets (mots) par le poids du plus court chemin allant d'un terme à un autre, en plus des classifications qui indiquent les changements de direction.

La force du rapport est donnée par :

Simht=T-PCC-K*nd

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

III.1.3. Approches hybrides

Ces approches sont fondées sur un modèle qui combine entre les approches basées sur les arcs (distances) en plus du contenu informationnel qui est considéré comme facteur de décision.

Parmi les travaux, recensés dans la littérature, sous cette bannière on peut citer :

III.1.3.1. Mesure de Leacock et Chodorow

Une méthode présentée par Leacock et Chodorow **[19]** qui combine entre la méthode de comptage des arcs et la méthode du contenu informationnel. La mesure proposée par est basée sur la longueur du plus court chemin entre deux synsets de Wordnet.

Les auteurs ont limité leur attention à des liens hiérarchiques «is-a » ainsi que la longueur de chemin par la profondeur globale M de la taxonomie.

La formule est définie par :

$$Simlc(X, Y) = -\log\left(\frac{dist(X, Y)}{2 * M}\right)$$

M est la longueur du chemin le plus long qui sépare le concept racine, de l'ontologie, du concept le plus en bas. On dénote par dist (X, Y) la longueur du chemin le plus court qui sépare X de Y.

III.1.4. Approches basées sur l'espace vectoriel

Dans le domaine de la recherche de l'information, les modèles de l'espace vectoriel sont largement adoptés **[04].** Ces approches utilisent un vecteur caractéristique, dans un espace dimensionnel, pour représenter chaque objet et calculent la similarité en se basant sur la mesure de cosine ou la distance euclidienne.

Le modèle de l'espace vectoriel est employé pour un arrangement des objets complexes en les représentants comme des vecteurs de k-dimensions. La définition de la similarité entre deux vecteurs d'objets est obtenue par leurs contenus internes. Sachant que ses deux vecteurs doivent impérativement prévenir d'une ontologie.

Parmi les approches citées dans la littérature on peut citer :

- ✓ la similarité de cosine
- ✓ la similarité euclidienne
- ✓ la similarité de Jaccard
- ✓ la similarité de Dice

Comme nous avons déjà défini toutes ces mesures en haut, on peut dire que ces dernières peuvent être considérées comme des mesures de similarités syntaxiques et sémantiques, la différence réside dont la nature des vecteurs. S'ils sont extraits d'une ontologie on parle des mesures de similarités sémantiques sinon, on parle des mesures de similarités syntaxiques

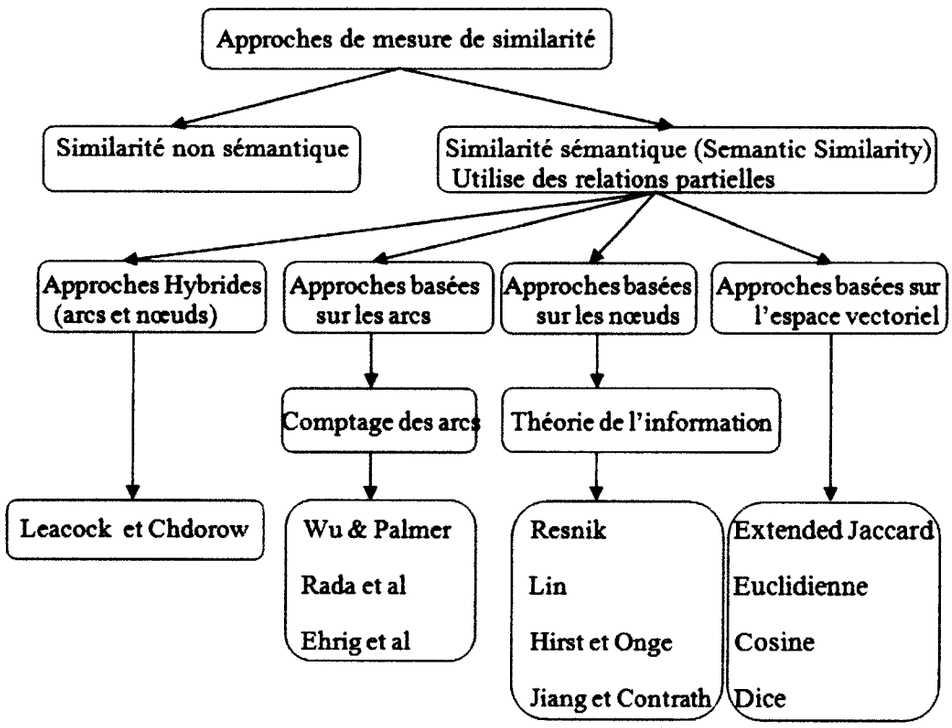

Figure III.4 : Taxonomie des approches de mesure de similarité.

IV. La mesure de similarité proposée

La mesure de Wup et Palmer est intéressante mais présente une limite car elle vise essentiellement à détecter la similarité entre deux concepts par rapport à leur distance de leur plus petit généralisant. Cette mesure présente l'avantage de la rapidité du temps d'exécution, mais l'inconvénient de la production d'une valeur de similarité de deux concepts voisins qui dépassent la valeur de deux concepts dans la même hiérarchie.

Exemple : La figure 5 représente un extrait de notre ontologie (SUMO), on dénote par C1, C2 et C3 les concepts «Physical», «Organisation» et «Agent».

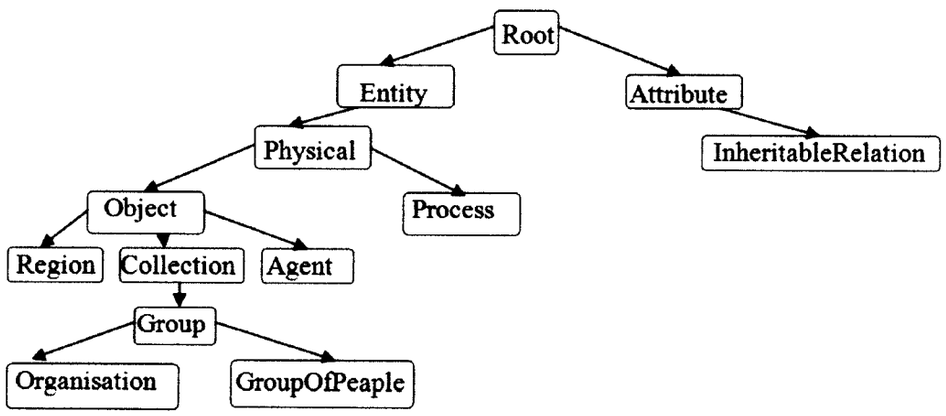

Figure III.5 : Graphe représentant une hiérarchie des concepts.

En appliquant la mesure de Wu et Palmer, la valeur de similarité est calculée comme suit :

Simwp(C1, C2)=2*2/(2+6)=0.5. Simwp(C2,C3) = 2*3/(4+6)=0.6.

Les valeurs de similarités obtenues par Wu et Palmer montrent que les concepts voisins C2 et C3 sont plus similaires que les concepts C1 et C2 situés dans une même hiérarchie ce qui est inadéquat dans le cadre de la recherche des informations sémantiques.

Nous proposons une nouvelle mesure qui inspire des avantages du travail de Wu et Palmer, dont l'expression est représentée par la formule suivante :

$$Sim(C1, C2) = \frac{2 \cdot Pi\ ofondeur(C)}{Pi\ ofondeur(C1) - Pi\ ofondeur(C2) - fp(C1, C2)}$$

fp(C1, C2) est la fonction de pénalisation des concepts C1 et C2. L'utilité de cette fonction est de pénaliser la similarité de deux concepts éloignés qui ne sont pas situés dans une même hiérarchie.

fp(C1, C2)= Prof(C1) + Prof(C2)

Tel que ***Prof(C1)*** est la distance en nombre d'arcs entre **C1** et le concept subsumant commun (CS).

Si on calcule l'exemple précédent avec la formule introduite :

SimHM(C1, C2)=2*2/(2+6)=0.5. SimHM(C2,C3) = 2*3/(4+6+4)=0.42.

Notre formule assure une valeur de similarité SimHM(C1, CV) inférieure à SimHM(C1,CF), tels que CV est un concept voisin (non inclus dans les nœuds fils de C1) et CF un nœud fils subsumé par C1.

De plus en plus que la chaine reliant deux concepts voisins est grande de plus en plus que la fonction de pénalité augmente et de ce fait la valeur de similarité diminue.

V. Conclusion

Dans ce travail nous avons présenté une extension de la mesure de Wu et Palmer. Notre mesure assure la pertinence des valeurs produites et favorise les concepts appartenant à la même hiérarchie par rapport aux concepts voisins.

Ceci peut être adopté dans le domaine de l'identification des associations sémantiques où les approches actuelles portent sur les associations ne donnant pas une précision sur le degré de justesse d'une association.

CHAPITRE IV

CONCEPTION ET RÉALISATION DU PROTOTYPE

I. Introduction

Les ressources sémantiques (thésaurus, ontologies, etc.) ont un apport considérable pour le traitement des documents et des applications.

En général les données ou les applications retournées par un système de recherche sont ordonnées à l'aide d'une mesure de similarité calculée entre les éléments de la base et la requête.

L'intérêt d'utiliser des ressources sémantiques en recherche d'application (Service Web) est de pouvoir retourner, lors d'une recherche par similarité, les services qui partagent avec la requête le maximum de concepts plutôt que le maximum de mots-clés.

II. Objectif

Notre travail a deux objectifs principaux:

- Aider l'utilisateur à trouver un service parmi un ensemble de services, en se basant sur une recherche sémantique.

  Cette recherche sémantique se fait à l'aide de calcul de similarité entre la requête (service demandé), et l'ensemble de services disponibles.

- Pour le calcul de similarité on a proposé deux mesures : Wu et Palmer et la mesure produite SimHM. le deuxième objectif est d'implémenter et comparer les performances de ces deux mesures de similarités dans un algorithme qui permet de vérifier si deux concepts sont sur la même hiérarchie ou pas.

III. Description du corpus

Les données utilisées dans nos expérimentations sont issues d'un échantillon du corpus owls-tc version 2.2.1. Cette base qui est déjà développée par le centre allemand pour la recherche en intelligence artificielle DFKI **[48].**

Les Services Web de la base sont décrits à travers les documents owls.

La base contient plus de 1007 Services Web, distribués sur 07 catégories, ces derniers sont :

1. Education
2. Medical care
3. Food
4. Travel
5. Communication
6. Economy
7. Weapon

OWLS-tc contient 29 requêtes associées avec leurs réponses prédéfinies. 06 Services Web qui ont une réponse de format WSDL.

Par souci de simplicité nous avons choisi uniquement :

- 2 requêtes (hospital_investigating_service.owls et governmentmissile_funding_service.owls).
- 2 catégories des Services Web qui sont : Medical care et Weapon.

Notre ontologie (SUMO), est une ontologie de domaine qui fusionne les deux dernières sous-ontologies de domaine: médical et Weapon (domaine de catégorisation d'entreprises)

IV. Les outils utilisés

IV.1. Le langage JAVA

Dans notre travail nous avons utilisé un langage orienté objet qui est le Java plus précisément nous avons utilisé la version 6.8.

- Java est portable sur la plupart des plates-formes.
- C'est un langage généraliste ayant un très vaste champ d'application (réseau, base de données..). Il permet de développer des applications professionnelles de grande taille.
- Il intègre une interface graphique de haut niveau.
- Il existe de nombreuses bibliothèques de programmes dans des domaines très variés, de sorte que ce langage devient professionnel de premier plan.
- Le code produit est indépendant de la plate-forme utilisée.

IV.2. NetBeans

NetBeans est un environnement de développement intégré (IDE) pour Java, placé en open source par Sun en juin 2000 sous licence CDDL (Common Development and Distribution License).

C'est un IDE offrant un éditeur avec des codes couleurs et un ensemble de signes, des modèles de projets multi-langage et de différents types (application indépendante, distribuée, plugin, mobiles, ...), le refactoring, l'éditeur graphique d'interfaces et de pages Web pour supporter le programmeur dans son travail. Il permet d'accéder rapidement à la documentation détaillée, de naviguer dans les sources et de faire des recherches d'usage des classes, méthodes et propriétés. Netbeans indique à l'utilisateur les erreurs et fait des propositions pour y remédier.

IV.3. API Jena

Jena est une API Java pour les applications du web sémantique. L'API a été définie en termes d'interfaces afin que le code d'application puisse travailler avec différentes implémentations sans changement. Ce paquet contient les interfaces pour représenter des modèles, des ressources, des biens, des littéraux, des déclarations et tous les autres concepts clés de RDF, et un ModelFactory pour créer des modèles.

IV.4. API Axis

Axis est un package Java libre qui fournit :

- Une API pour développer des services web SOAP RPC ou à base de messages SOAP
- Le support de différentes couches de transport : HTTP, FTP, SMTP, POP et IMAP.
- La sérialisation/désérialisation automatique d'objets Java dans des messages SOAP
- Des outils pour créer automatiquement les WSDL correspondant à des classes Java ou inversement pour créer les classes Java sur la base d'un WSDL (classe proxy en quelque sorte, qui fait le lien entre l'application Java cliente et le service distant).
- Des outils pour déployer, tester et monitorer des web-services.

IV.5. API JFreeChart

JFreeChart est une bibliothèque 100% gratuite des graphiques Java qui rend facile pour les développeurs d'afficher des graphiques de qualité professionnelle dans leurs applications. C'est une bibliothèque de classe pour l'utilisation par les développeurs, et non pas une application utilisateur final. Elle comprend:

- Une API cohérente et bien documentée, en soutenant un large éventail de types graphiques
- Une conception flexible qui est facile à étendre, et vise à la fois côté serveur et côté client des applications.
- Support pour les types de sortie beaucoup, y compris les composants Swing, les fichiers image, et les formats vectoriels de fichier graphique (PDF, EPS,...)
- Free software . JFreeChart est "open source" ou, plus précisément, le logiciel libre.

V. L'algorithme proposé

1. Parcourir la base des Services Web

2. Créer la matrice **M1**, (qui contient les ***Sorties*** des requêtes(OR) et des Services Web(OS)) tel que :

- Lignes ⟶ Les Output des Requêtes.
- Colonnes ⟶ Les Output des Services Web de la base.

i. Calculer chaque case de la matrice **M1** à travers les deux mesures de similarités *Wu et Palmer* ainsi que la mesure qu'on a proposée *SimHM.*

ii. Trier les résultats dans le sens décroissant des scores.

iii. Tant que le nombre de lignes n'est pas atteint.

- Choisir les paires ayant les scores les plus grands.

iv. Calculer le score global :

$$\text{Score global} = \frac{\text{Somme des sous scores}}{\text{Nombre des lignes}}$$

3. Refaire les mêmes étapes précédentes pour la comparaison des ***Entrées*** des requêtes (ER) avec celles des Services Web (ES) mais en permettant :

- Lignes ⟶ Les Input des Services Web de la base.
- Colonnes ⟶ Les Input des Requêtes.

4. Calculer la moyenne :

$$\text{Sim} = \frac{\text{Score Entrées} + \text{Score Sorties}}{2}$$

5. Retourner uniquement les Services qui ont un score >= seuil (ce dernier est fixé par l'utilisateur).

VI. Modélisation UML

Dans ce qui suit nous présentons les principaux diagrammes qui reflètent notre conception.

VI.1. Diagramme de cas d'utilisation

Ce model est composé de trois cas d'utilisations :

- L'invocation
- La consultation
- La recherche

Chacun d'entre eux est spécialisé dans certain nombre de sous cas d'utilisations.

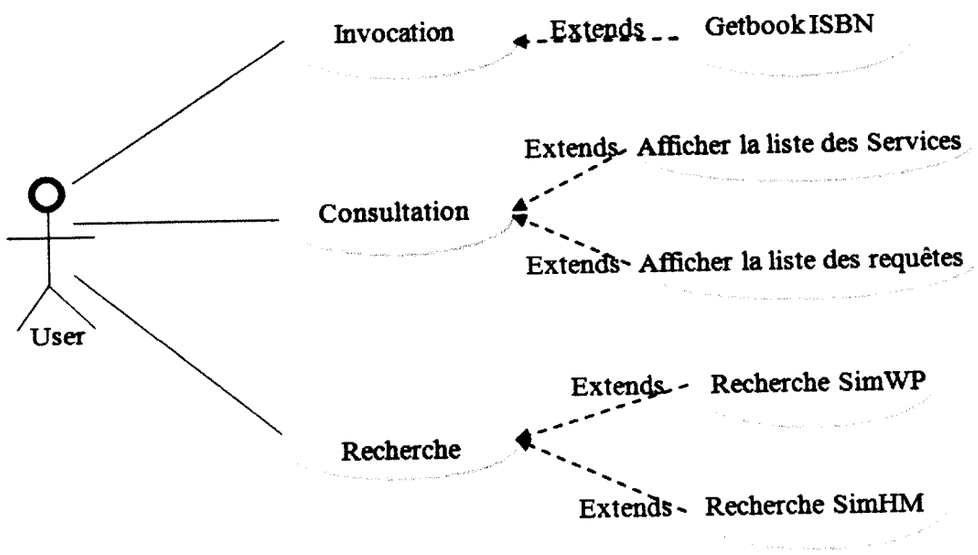

Figure IV.1 : Diagramme de cas d'utilisation de notre application

VI.2. Diagramme de classe

Dans cette section, et pour plus de compréhension, nous proposons, notre système de recherche. Ce modèle est composé d'une classe principale « Main» et des classes secondaires « Sim Complexe », « Sim Simple », « Traitement Ontologie », « PPS».

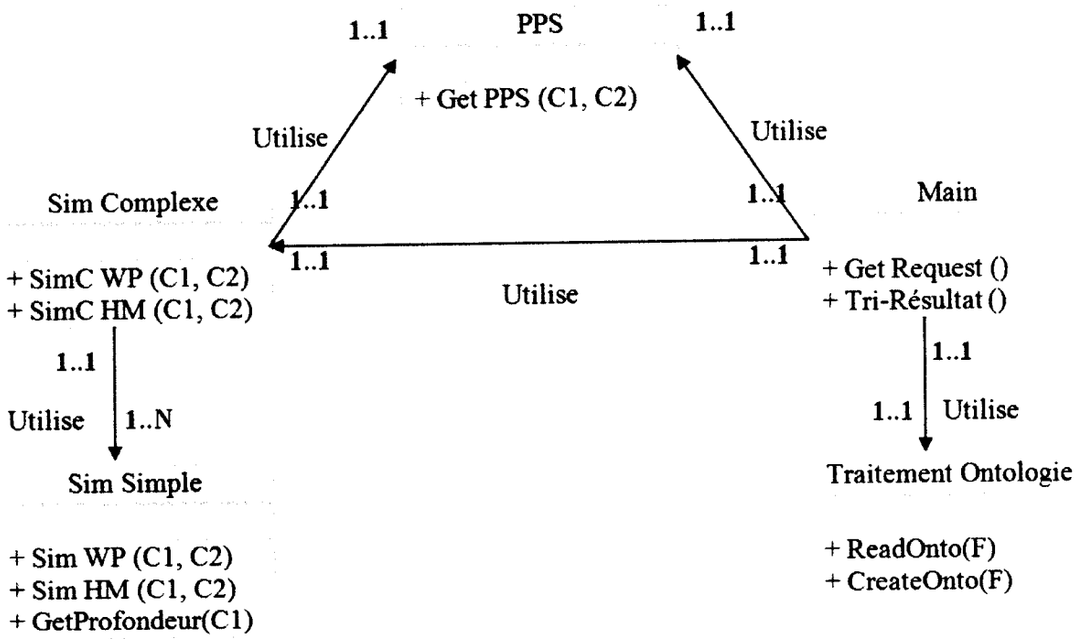

Figure IV.2 : Diagramme de classe de notre application

VI.3. Un extrait de notre ontologie (SUMO)

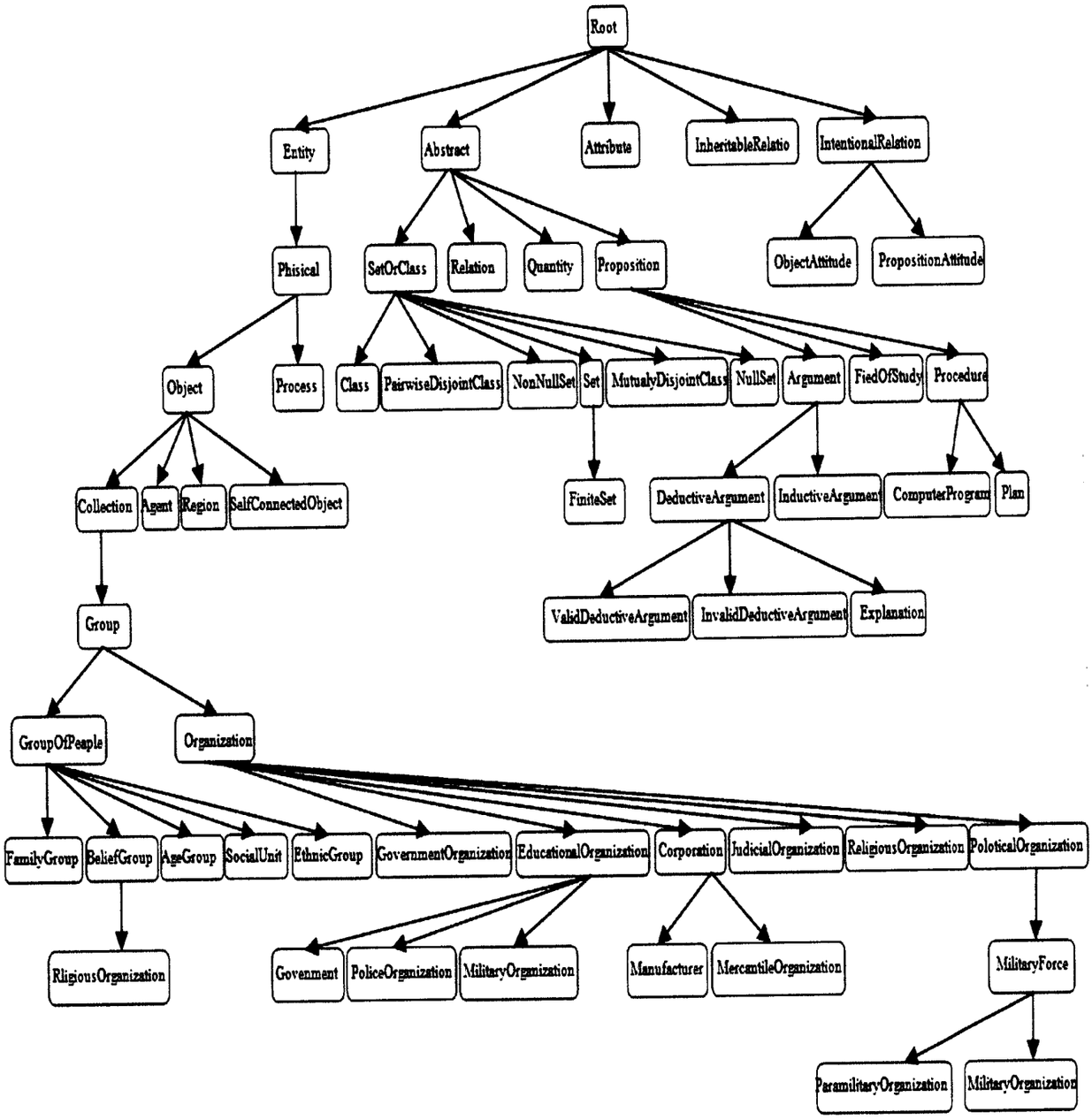

Figure IV.3 : Un extrait de l'ontologie (SUMO)

VII. Présentation du Prototype

L'IHM suivante consiste à aider l'utilisateur de choisir une des deux requêtes : hospital_investigating_service.owls ou bien governmentmissile_funding_service.owls

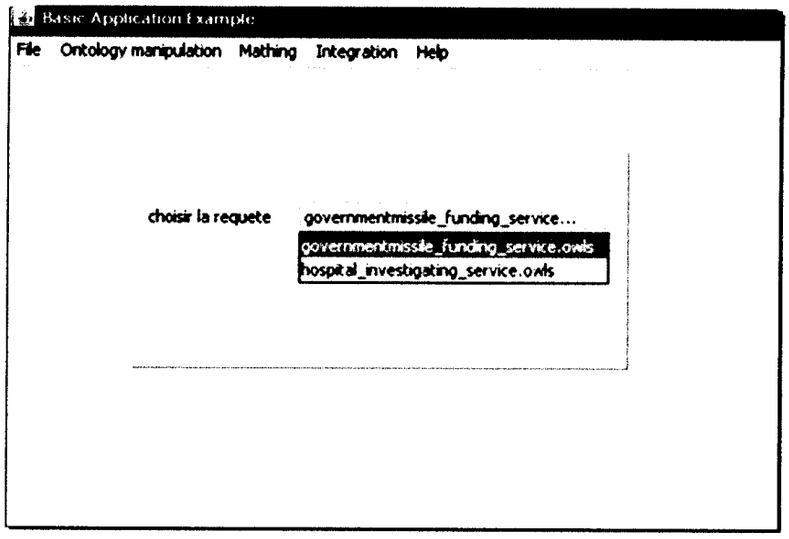

Figure IV.4 : Choix de la requête

L'IHM suivante fait le calcule de la similarité entre la requête choisi et tous les Services Web de la base à travers la mesure de **Wu et Palmer**.

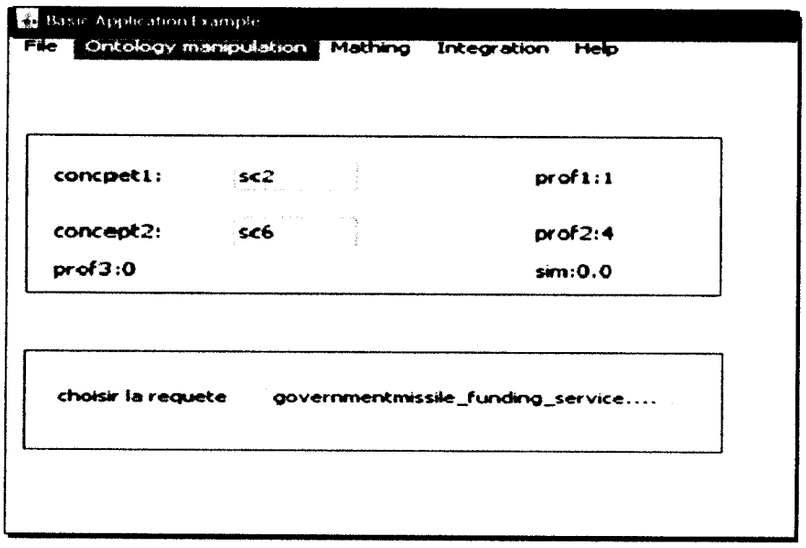

Figure IV.5 : Mesure de similarité de Wu & Palmer

L'IHM suivante permet à l'utilisateur de choisir l'une des deux méthodes pour afficher les résultats de comparaison. En cliquons sur la méthode 1 par exemple (celle de Wu et Palmer) on obtient les résultats suivants.

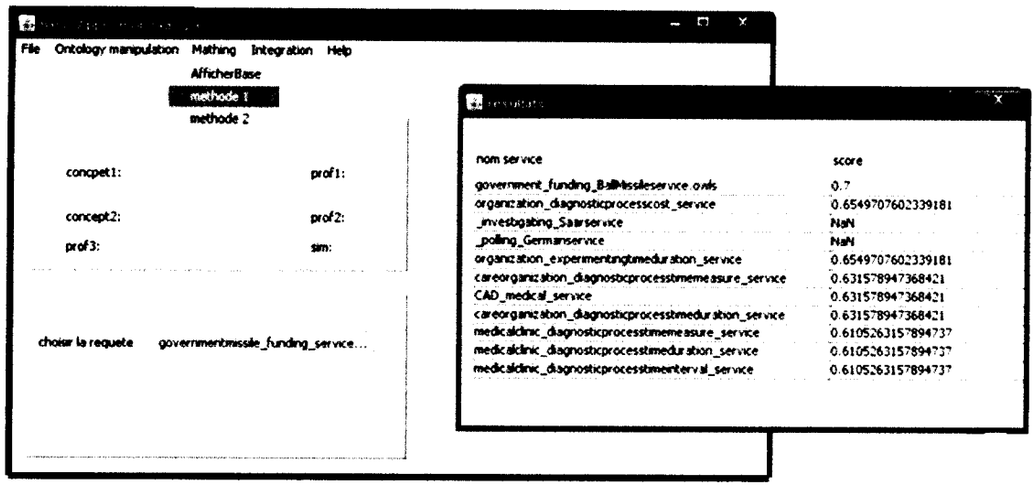

Figure IV.6 : Algorithme de recherche basé sur Wu et Palmer

La dernière IHM fait la même chose que la précédente mais les résultats affichés sont de l'algorithme qu'on a proposé (Choisir la méthode 2).

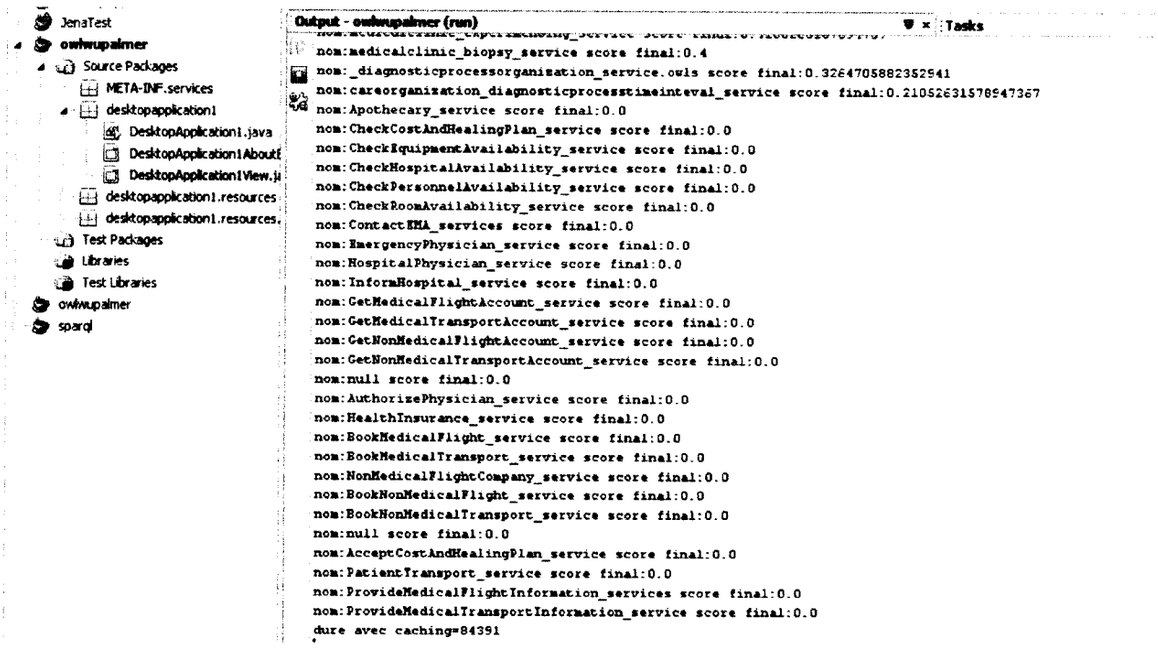

Figure IV.7 : Les résultats de l'algorithme de recherche proposé (Méthode 2)

VIII. Résultats

Nous avons réalisé des expérimentations à l'aide d'un laptop ayant :

- Un processeur dual-core, T4400 2.20GHZ,
- 4 G.O de RAM,
- Un système d'exploitation de type windows SP3

Nous avons présenté dans ce chapitre deux approches de recherche sémantique des Services Web.

La première adopte la mesure de Wu et Palmer et la deuxième adopte notre mesure SimHM. Elles ont données les performances suivantes :

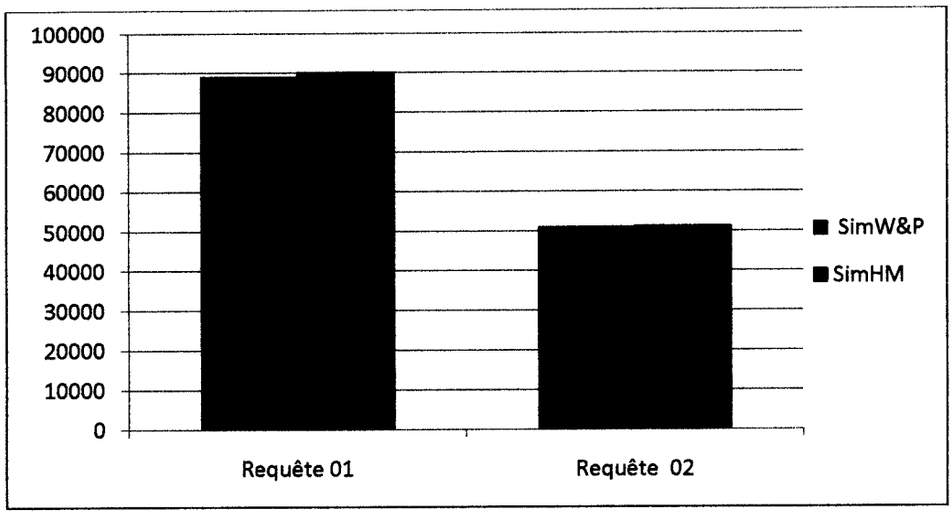

Figure IV.8 : Temps d'exécution moyen des deux mesures W&P et SimHM

On remarque qu'il ya pas une très grande différence entre les deux mesures de similarités en terme de temps d'exécution.

La mesure de W&P est légèrement plus rapide que notre mesure SimHM, car cette dernière nécessite de tester en plus si deux concepts sont sur la même hiérarchie ou pas.

La mesure produite SimHM assure une pertinence en termes de qualité.

IX. Conclusion

Nous avons présenté dans ce chapitre un système de recherche des Services Web qui se base sur deux mesures de similarités sémantiques : la première est celle de Wu et Palmer et la deuxième est une amélioration de la première en termes de qualité.

Les mesures de similarités possèdent une complexité polynomiale. Les résultats préliminaires obtenus sont acceptables et encouragent des futurs travaux sur cet axe de recherche.

Conclusion

Au cours de l'élaboration de ce mémoire, nous avons montré deux approches de recherche sémantique des Services Web à base de deux mesures de similarités sémantiques.

La première est la mesure de Wu et Palmer, elle est articulé sur le nombre d'arcs intermédiaires séparent les nœuds à comparer (Concepts) en plus de la profondeur du plus petit généralisant de ces nœuds.

La deuxième est notre mesure de similarité SimHM dérivée de la première afin d'améliorer la qualité et la pertinence de la recherche des Services Web.

Les deux mesures de similarités employées possèdent une complexité polynomiale.

Ce travail a été fait pour aider l'utilisateur à trouver un service parmi un ensemble des services.

Comme perspectives à notre travail, nous proposons quelques types d'améliorations à apporter :

- Implémenter d'autres mesures de similarités par exemple des mesures basées sur les nœuds, basées sur l'espace vectoriel ou bien hybride.
- Pondérer les entrées et les sorties.
- Introduire les *post* et les *pré-conditions* comme filtres de recherche.

Bibliographie

Bibliographie

[01] ANDRE .B, Thèse Pour obtenir le grade de DOCTEUR, UNIVERSITE JOSEPH FOURIER – GRENOBLE I, 1994.

[02] BACHIMONT .B, *l'intelligence artificielle comme écriture dynamique : de la raison graphique à la raison computationnelle,* Grasset, Paris, 1999.

[03] BACHIMONT .B, Engagement sémantique et engagement ontologique : conception et réalisation d'ontologies en ingénierie des connaissances, in CHARLET J, ZACKLAD M, KASSEL G. & BOURIGAULT D, eds, *ingénierie des connaissances : évolution récentes et nouveaux défis, Eyrolles,*2000.

[04] BAEZA-YATES .R, et RIBEIRO-NETO .B, Modern Information Retrieval. ACM Press; Addison-Wesley: New York; Harlow, England; Reading, Mass., 1999.

[05] BURGUN .A, mappingUMLS. BURGUN .A, and BODENREIDER .A, Mapping the UMLS Semantic Network into general ontologies. Proceedings of the AMIA Annual Symposium. 2001.

[06] DECHILLY .T, & BACHIMONT .B, Une ontologie pour éditer des schémas de description audiovisuels, extension pour l'inférence sur les descriptions, in Actes des journées francophones d'ingénierie des connaissances (IC'2000), 2000.

[07] DUMAS .M, Intergiciel et Construction d'Applications Réparties. Licence Creative Commons, version du 12 juin 2008.

[08] EHRIG .M, HAASE .P, HEFKE .M, et STOJANOVIC .N, Similarity for ontology-a comprehensive framework. In Workshop Enterprise Modelling and Ontology: Ingredients for Interoperability, 2004.

[09] FERNANDER .M, COMEZ-PEREZ .A & JURISTO .N, METHONTOLOGY : from ontological art towards ontological engineering, in proceedings of the Spring Symposium Series on Ontological Engineering, AAAI Press, 1997.

[10] GRUBER .T, Toward principals for the design of anthologies used for knowledge sharing. Revision of paper presented at the international Workshop on Formal Ontology, Padova, Italy, in Special Issue of International Journal on Human-Computer Studies, March1993.

[11] GRUBER .T, & OLSEN .G, An ontology for engineering mathematics, in DOYLE .F, & TORANO .P, eds, Proceeding of the 4th International Conference on Principals of Knowledge Representation and Reasoning, Morgan-Kauffmann, 1994.

[12] GRUNINGER .M, & FOX .M, Methodology for the design and evaluation of ontology, in proceedings of the Workshop on Basic Ontological Issue on Knowledge Sharing, IJCAI'95, 1995.

[13] GUARINO .N, & GIARETTA .P, Ontologies and Knowledge bases, towards a terminological clarification, in Mars N., eds, Towards very large Knowledge building and Knowledge sharing.ISO Press, 1995.

[14] HAMMING, R. Error-detecting and error-correcting codes Bell System Technical Journal 29(2), 1950

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

[16] JARO .M,« *Advances in record linking methodology as applied to the 1985 census of Tampa Florida* », In Journal of the American Statistical Society, 1989.

[17] JIANG .J, et CONRATH .D, Semantic similarity based on corpus statistics and lexical taxonomy. In Proceedings of International Conference on Research in Computational Linguistics, Taiwan, 1997.

[18] KAVEH .B, Le rôle des ontologies de domaine dans la conception des interfaces de navigation pour des collections en ligne de musées: évaluations et proposition, 2004.

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

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

[21] MONFORT .V, & KADIMA .H, Les Services Web : techniques, démarches et outils, Dunod, 2003.

[22] RADA .R, MILI .H, Bichnell .E, et BLETTNER .M, Development and application of a metric on semantic nets. IEEE Transaction on Systems, Man, and Cybernetics,1989.

[23] RESNIK .P, Using information content to evaluate semantic similarity in taxonomy. In Proceedings of 14th International Joint Conference on Artificial Intelligence, Montreal, 1995.

[24] THABET .S, MELLOULI .K, BEN YAGHLANE .B. Une extension de mesure de similarité entre les concepts d'une ontologie. 4th International Conference: Sciences of Electronic, Technologies of Information and Telecommunications March, 2007 – TUNISIA

[25] USCHOLD, M. and Gruninger, M. Ontologies : Principles, Methods and Applications. Knowledge Engineering Review 11(2),1996.

[26] USCHOLD M. & KING M., Towards a methodology for building ontologies, in proceedings of the Workshop on Basic Ontological Issues in Knowledge Sharing, IJCAI'95,1995.

[27] WU .Z, et PALMER .M, Verb semantics and lexical selection. In Proceedings of the 32nd Annual Meeting of the Associations for Computational Linguistics, 1994.

[28] Winkler .W.E, The State of record linkage and current research problems statistics of Income Division, Internal Revenue Service Publication 2004.

[29] W3C Recommendation SOAP Version 1.2 Part 0 :Primer.24 June 2003.

[30] W3C UDDI Technical White Paper. W3C September 6, 2000.

[31] ZWEIGENBAUM .P Encoder l'information médicale : des termonologies aux systèmes de représentation des connaissances, in Innovation stratégique en information de santé (ISIS) (2-3), 1999.

Webographie

[32] W3C http://www.w3.org/2002/ws/ ,24 Avril 2008

[33] http://www.ibm.com/webservices

[34] http://fplanque.net/Blog/devblog/2004/03/08/webservices_pour_quoi_faire

[35] http://www.w3c.org/2002/ws/arch/2/08/wd-wsa-arch-20020821.html#webservice

[36] http://soap.weblogs.com.

[37] http://www.W3.org/TR/WSDL

[38] http: //www.labo-sun.com

[39] http://www.uddi.org.

[40]Type d'UDDI http://www-106.ibm.com/developerworks/library/ws-rpul.html

[41] http://www.omg.org/corba

[42] http://fr.wikipedia.org/wiki/Remote_method_invocation_%28Java%29

[43] Wikipedia, http://fr.Wikipedia.org/wiki/service_web, 21 Avril 2008

[44] http://liris.cnrs.fr/~amille/enseignement/DEA-ECD/ontologies/notion_ontologie.htm

[45] http://www.nlm.nih.gov/research/umls

[46] http://www.ontoknowledge.org/OIL

[47] http://www.livrespourtous.com

[48] http://www.dfki.de/scallops