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: *Système d'information et de Connaissance (S.I.C)*

Thème

Composition des Services Web Sémantiques à base d'Algorithmes Génétiques

Réalisé par :

- *DAOUD Zahèra*
- *MAALACHE Nassima*

Présenté le : *04 Juillet 2013 devant le jury composé de MM.*

- Mr SMAHI Ismail	*(Président)*
- Mr MERZOUG Mohamed	*(Encadreur)*
- Mr HADJILA Fethallah	*(Examinateur)*
- Mr BELABED Amine	*(Examinateur)*

Année universitaire: 2012-2013

REMERCIEMENTS

Nous tenons à remercier notre encadreur

M MERZOUG Mohamed

pour ses conseils, sa disponibilité et son encouragement qui nous ont permis de réaliser ce travail dans les meilleures conditions.

Aux honorables membres de mon jury

qui ont bien voulu sacrifier de leur temps à la lecture de ce modeste travail et à son évaluation. Qu'ils en soient remerciés.

Enfin, à tous nos enseignants

qui nous ont fait profiter de leur savoir, sans lequel, un tel travail n'aurait pu voir le jour.

Dédicace

A ma mère et à mon père pour leur aides et encouragement

Spécialement à mon mari Ali pour son encouragement, sa patience et son aide.

Et surtout a mes deux chères filles Safaa et Sarah

A mes frères, ma sœur et leurs familles

A mes collègues et mes amies

Nassima.

DEDICACES

Ma profonde reconnaissance va d'abord

A MES PARENTS

pour toute l'affection dont ils m'ont entouré
et pour l'encouragement moral qu'ils m'ont témoigné

*A mon mari **LAKHDAR***
*et notre adorable fille **AYA** que j'adore*

A mes frères
YOUCEF et ZAKARIA

A mes sœurs
DJAZIA et RABIHA

A
Inès, Chaimaa et Djewed

A tous mes amis et collègues

ZAHERA

Résumé

Les services web sont des applications accessibles sur Internet réalisant chacune une tâche spécifique. Pour fournir une solution à une tâche complexe, on peut regrouper des services web pour n'en former qu'un seul ; on parle alors de composition de services web. De ce fait le besoin d'une composition dynamique et automatique se fait sentir. Dans ce mémoire, nous proposons une approche pour la composition dynamique des services web sémantiques basée sur un algorithme génétique, en prenant en compte les propriétés fonctionnelles des services Web -(spécifiées par les paramètres d'E/S)- afin de satisfaire les exigences de l'utilisateur. Les résultats obtenus sont encourageants et méritent d'être poursuivis.

Mots-clés: service web, web sémantique, composition des services web, appariement sémantique, algorithme génétique.

Abstract

Web services are applications accessible on the Internet each performing a specific task. To provide a solution to a complex task, one can combine web services to form another one; this is called web service composition. Hence the need for a dynamic and automatic composition arises. In this work we propose an approach for the dynamic composition of semantic web services based on genetic algorithm. This later takes into account both functional properties of web services -specified by their I/O parameters- and non-functional properties -specified by their QoS parameters- in order to fulfill the user requirements. The obtained results are encouraging and merit to be continued.

Keywords: web service, semantic web, service composition, semantic matching, and genetic algorithm.

ملخص:

تعرف خدمات الواب كبرنامج مستقل ينفذ على الواب كما توصف خدمات الواب عن طريق (WSDL) التي تخزن في ملفات (UDDI) لتسهيل عملية البحث فيما بعد.

و تجدر الإشارة أن استقلالية خدمات الواب تزيد من صعوبة التركيب الآلي لخدمات الواب.

لمقاربة هذه الإشكالية، نقترح في هاته الرسالة حلا يعتمد على الخوارزميات الجينية من أجل تحسين أو استمثال دالة موضوعية تمثل جميع متطلبات المستخدم.

تعتبر النتائج المحصل عليها جد مشجعة و تفتح المجال من أجل مقاربات أخرى.

الكلمات المفتاحية : خدمات الواب، الواب الدلالي، تركيب الخدمات، الخوارزمية الجينية .

Table des matières

Chapitre I : Les services Web

Chapitre II : Les services Web sémantiques

III. Chapitre: Conception et implémentation de prototype

Liste des Figures

Liste des tableaux

Introduction Générale

Introduction générale

Plusieurs innovations technologiques ont eu lieu ces dernières années pour répondre aux besoins de partage des ressources et de communication entre les entreprises. Ceci a conduit à remettre en cause la rechercher de meilleures façons de développer des solutions informatiques [**Chouchani 2010**].

Apparus dès la fin des années 1990, à l'aube du 21ème siècle, les services Web ont provoqué une forte évolution dans le monde de l'informatique distribuée, et un bouleversement majeur dans la façon de concevoir des architectures. Un des intérêts du service Web est de faciliter l'interconnexion entre les différentes applications distantes, indépendamment des plateformes et des langages de programmation utilisés. Les services Web semblent être la solution de l'avenir pour implémenter les systèmes distribués, aujourd'hui, ces services sont distribués à large échelle sur Internet. Le développement d'Internet, la structuration des données via XML, et la recherche d'interopérabilité sont autant de facteurs qui ont favorisé l'essor des services Web. Les services Web constituent la technologie de base pour le développement d'architectures orientées services. Ces derniers sont des modèles qui définissent un système par un ensemble de services logiciels distribués, qui fonctionnent indépendamment les uns des autres afin de réaliser une fonctionnalité globale.

Les architectures orientées services, et en particulier les services Web, permettent l'accessibilité, la découverte et l'utilisation universelle de n'importe quelle application logicielle sur le Web en utilisant des normes ouvertes. Ils constituent un paradigme permettant d'organiser et d'utiliser des services Web distribués. Ainsi, les logiciels codés dans divers langages de programmation et sur diverses plateformes d'exécution peuvent employer des services Web, pour échanger des données à travers le Web. Bien que, les architectures orientées service soient récemment proposées, elles connaissent un engouement et focalisent l'attention de bon nombre de professionnels dans le domaine des technologies de l'information et de la communication, des chercheurs et des industriels [**Boukhadra 2011**].

Contexte du travail :

Les services Web fournissent une nouvelle manière de développer des applications conformes aux besoins de l'Internet en vue de rendre le Web plus dynamique. Ils semblent être la solution la plus adaptée pour assurer l'interopérabilité, qui permet de transmettre les données entre les différentes applications d'une organisation. Cette technologie permet de réaliser le traitement de ces données, et gérer les liaisons entre les différentes applications.

La particularité d'une plateforme de services Web réside dans le fait qu'elle utilise la technologie Internet comme infrastructure pour la communication entre les services Web, et ceci en mettant en place un cadre de travail basé sur un ensemble de standards. La technologie services Web est la technologie clé permettant l'intégration des applications via le Web. Les services Web peuvent être définis comme des programmes modulaires, généralement indépendants et auto descriptifs, qui peuvent être découverts et invoqués via l'Internet ou l'Intranet.

La popularité des services Web et leur utilisation dans ces contextes variés s'expliquent par le fait que les services Web peuvent opérer dans des environnements distribués particulièrement hétérogènes, aussi bien, du point de vue des plateformes logicielles déployées que des modèles de programmation utilisés. Les services Web sont généralement fournis par des organisations différentes et indépendamment de tout contexte d'exécution.

Dans la communauté des services Web, des efforts importants sont maintenant consacrés à ce problème, qui consiste a proposé une architecture orienté service qui regroupe la découverte, la composition et la sélection de services Web.

Problématique

Le processus de composition s'est avéré essentiellement un processus qui prenait beaucoup de temps. En effet, une composition n'est pas simplement un regroupement quelconque de services web, mais un ensemble dont les tâches sont ordonnées en fonction des relations reliant

ses services web. Ces derniers sont généralement fournis par des organisations déférentes et indépendamment de tout contexte d'exécution. Puisque, chaque organisation possède ses propres règles de travail, ils doivent être traités comme des unités strictement autonomes. A cause de cette hétérogénéité et de cette autonomie inhérente aux services web sémantiques, leur composition devient plus complexe.

En outre, une composition manuelle des services web génère beaucoup d'erreurs et comporte plusieurs tâches fastidieuses et répétitives. De ce fait le besoin d'une composition dynamique et automatique se fait sentir.

Organisation du mémoire

Le manuscrit est structuré comme suit :

Le premier chapitre est consacré aux services web et les technologies adjacentes. Dans le deuxième chapitre nous présentons les objectifs visés par les services web sémantiques ainsi que les différents langages qui sont utilisés pour expliciter la sémantique dans les descriptions des services web. Le troisième chapitre est lié à notre contribution, nous présentons l'approche de composition proposée tout en détaillant notre mécanisme de matching sémantique puis nous détaillerons l'algorithme génétique (codage, fitness,opérateurs génétiques,…..), nous présentons l'outil expérimental développé ainsi que les résultats obtenus.

Pour finir, une conclusion générale reprendra notre contribution dans ce mémoire, et en analysera ses limites. Nous proposerons enfin quelques perspectives de recherche de ce travail.

Chapitre I

Les services Web

I. Chapitre 1 : Les services Web

1. Introduction :

L'une des tendances historiques qui a conduit à l'apparition des services Web est l'utilisation de l'architecture par composants comme approche d'intégration des applications [PFI96]. Les composants sont des entités logicielles indépendantes fondées sur une interface et une sémantique bien définie.

Après l'avènement du B2C (Business To Consumer), où les entreprises mettent en ligne leurs services pour leurs consommateurs à travers des applications Web, celles-ci souhaitent augmenter leur productivité à l'aide du paradigme B2B (Business To Business). Le B2B repose sur l'échange de produits, d'informations et de services entre entreprises. Ceci implique l'utilisation de services et la collaboration avec des systèmes proposés par d'autres concepteurs et par conséquent une maîtrise de l'hétérogénéité. L'interopérabilité est ainsi devenue une nécessité pour l'entreprise dans le monde du B2B. C'est justement ce que les services Web apportent par rapport aux solutions dites monolithiques. D'une certaine façon, le modèle des services Web est une évolution du modèle des composants distribués rendu nécessaire par l'utilisation intensive de l'Internet.

Cependant la solution des services Web repose sur l'ubiquité de l'infrastructure d'Internet alors que les autres architectures reposent chacune sur sa propre infrastructure.

L'interopérabilité est donc une caractéristique intrinsèque aux services Web. Définir les services Web nécessite alors d'introduire à la fois leurs architectures et leurs infrastructures. Dans ce qui suit, nous proposons une définition des services Web puis nous exposons l'architecture ainsi que l'infrastructure nécessaire.

2. Définition :

Dans la littérature, il existe de nombreuses définitions des Web services. Cette prolixité montre que la notion de service Web a besoin d'être éclaircie, et motive des travaux de recherches.

En 2001, Nagy et al. écrivent dans [CNW01] : "Un service Web est une application accessible à partir du Web. Il utilise les protocoles Internet pour communiquer et utilise un langage standard pour décrire son interface."

Un service web est un système logiciel conçu pour supporter les interactions entre application à travers le réseau. Les services web offrent un moyen standard d'interopérabilité entre différentes applications qui s'exécutent sur une variété de machine/plateforme. Ils sont caractérisés par leur grande interopérabilité et extensibilité, ainsi qu'une description interprétable/compréhensible automatiquement par la machine grâce au standard XML. Ils peuvent être combinés d'une façon faiblement couplée afin de réaliser des opérations complexes. Les programmes offrant des services simples, peuvent interagir ensemble afin de mettre en place des services sophistiqués avec des valeurs ajoutées."

Les services web sont des compléments aux programmes et applications existantes, développées dans différents langages de programmation, et servent de pont pour que ces programmes communiquent entre eux. Ainsi, les services web permettent d'interfacer des systèmes d'information hétérogènes et ont pour atouts :

- Un faible couplage avec les technologies employées en interne.
- Une grande flexibilité de mise à jour des systèmes employés de part et d'autre.
- L'emploi de protocoles réseau simples, répandus et bénéficiant d'implémentations dans toutes les technologies majeures.

Les services offerts par l'infrastructure des services web concernent essentiellement deux aspects fondamentaux :

- Un service de communication qui permet l'échange de données entre les services web.
- Un ensemble de services techniques destinés à automatiser le processus de localisation et d'invocation des composants

3. Architecture des services web :

3.1 Besoin en architecture :

Les architectures en informatique proposent des schémas directifs et une vision sur le fonctionnement et la dynamique des systèmes. Pour promouvoir l'interopérabilité et l'extensibilité du paradigme des services Web, une architecture de référence est

nécessaire afin de préserver les objectifs initiaux visés par les services Web lors des évolutions technologiques successives.

Une architecture orientée services SOA : (Service Oriented Architecture) : L'architecture des services Web comme l'architecture du Web sont des instances d'architectures orientées services (SOA) **[**BOO03**]**. Elle propose une perspective globale sur le développement, la gestion et le fonctionnement des services Web [BAR03]. L'architecture SOA est un modèle (abstrait) qui définit un système par un ensemble d'agents logiciels distribués qui fonctionnent de concert afin de réaliser une fonctionnalité globale préalablement établie [HEA01].

Elle consiste à diviser le logiciel répondant à un problème, en un ensemble d'entités proposant des services. Chacune de ces entités peut utiliser les services proposés par d'autres entités. On obtient ainsi un réseau de services interagissant entre eux. Cette architecture s'appuie sur une architecture à composants (implémentation « réelle » des services **[**HUM04**]**)et suit l'évolution logique des architectures logicielles [END et al 04] (Figure 1) :

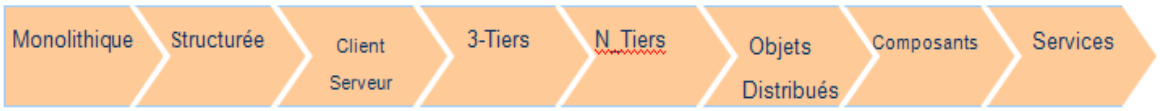

Figure 1 - Evolution des architectures logicielles.

Le choix d'une architecture SOA entre dans la perspective de transformer le Web en une énorme plate-forme de composants faiblement couplés et automatiquement intégrables. L'architecture SOA vise trois objectifs importants [SW4] : (i) identification des composants fonctionnels, (ii) définition des relations entre ces composants et (iii) établissement d'un ensemble de contraintes sur chaque composant de manière à garantir les propriétés globales de l'architecture.

3.2 Modèle de fonctionnement :

Les interactions entre les services Web impliquent trois participants (acteurs) **[**KEL03**]**: le fournisseur de services (prestataire), l'annuaire de services et le client (utilisateur du service) (voir Figure2).

- Le fournisseur de services correspond au propriétaire du service (i.e. entité responsable du services Web). D'un point de vue technique, il est constitué par la plate-forme d'accueil du service.
- Le client correspond au demandeur du service. D'un point de vue technique,

  il est constitué par l'application de recherche et d'invocation d'un service. L'application client peut être elle même un service Web.
- L'annuaire des services représente l'entité logicielle qui joue le rôle de l'intermédiaire entre les clients et les fournisseurs de services, il correspond à un registre de descriptions de services offrant des facilités de publication de services pour les fournisseurs ainsi que des facilités de recherche pour les clients et le moyen de localiser leurs besoin en terme de services.

La dynamique de l'architecture se décompose ainsi : Le fournisseur de services définit la description de son service et la publie dans un annuaire de service (dans des registres) en vue d'être localisé par des clients; Le client utilise les facilités de recherche disponibles au niveau de l'annuaire pour retrouver et sélectionner un service donné. Il examine ensuite la description du service sélectionné (extrait sa description du registre) pour récupérer les informations nécessaires lui permettant de se connecter au fournisseur du service et d'interagir avec l'implémentation du service considéré (entreprend une interaction) [BHI05].

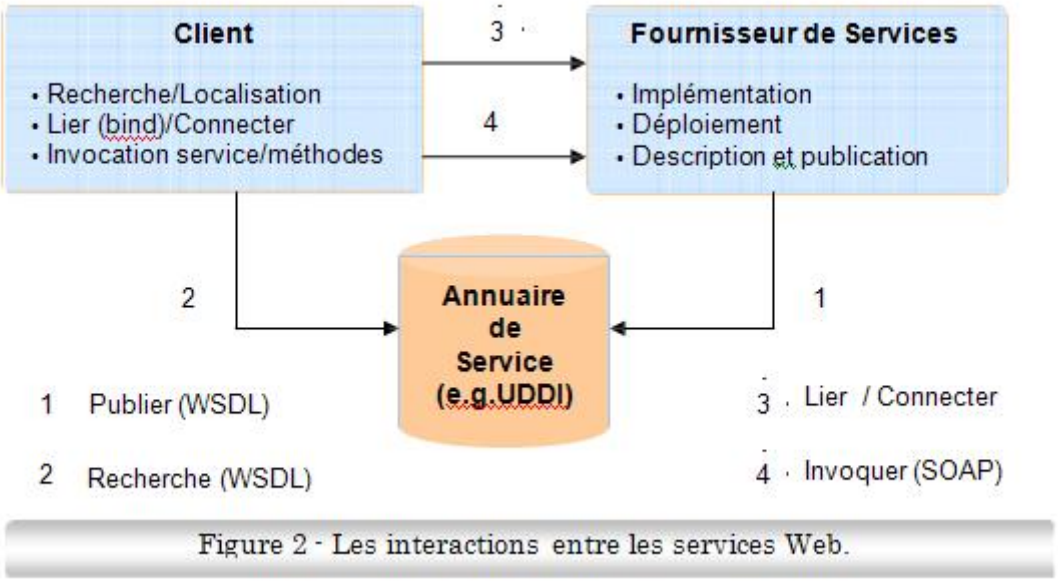

Figure 2 - Les interactions entre les services Web.

Les interactions entre les services Web impliquent trois intervenants: le fournisseur de services, l'annuaire de services et le client.

Le modèle de fonctionnement de l'architecture des services Web se base sur un cadre technologique qui constitue l'infrastructure de l'architecture par composants des services Web. Cette infrastructure offre les services nécessaires pour la réalisation des différentes étapes du cycle vie d'un service Web.

4. L'infrastructure des services Web : les standards mis en jeu dans l'architecture

L'originalité de l'infrastructure des services Web consiste à mettre en place ces services en se basant exclusivement sur les protocoles les plus répandus d'Internet.

Pour garantir l'interopérabilité des trois opérations précédentes (publication, recherche et lien), l'infrastructure services Web s'est concrétisé autour d'un ensemble de spécifications considérées comme des standards pour chaque type d'interactions:

Un protocole abstrait de description et de structuration des messages, SOAP[1] [SOA00], une spécification XML qui permet la publication et la localisation des services dans les annuaires, UDDI[2] [UDD02] et un format de description des services Web publiées dans les annuaires, WSDL[3] [WSD01].

4.1. Rôle de XML dans l'infrastructure services Web :

La technologie XML (eXtensible Markup Language), standardisée par le W3C (World Wide Web Consortium) en 1998 est aujourd'hui largement reconnue, acceptée et utilisée par de nombreuses entreprises comme format universel d'échange de données. On retrouve dans XML une généralisation des idées contenues dans HTML et SGML (Standard Generalized Markup Language). Reposant sur un système de balises au sein d'un fichier texte, XML peut être employé pour exprimer n'importe quel type d'information. On le retrouve par exemple aussi bien comme élément de sauvegarde de documents au sein de fichiers ou de bases de données ou encore comme format d'échange de données. [DAN03]

Dans HTML, on n'utilise les balises que pour décrire l'aspect graphique que doit revêtir la page dans le navigateur Web. Dans XML, les balises permettent d'associer toutes sortes d'informations au fil du texte.

XML est aujourd'hui un standard qui permet de décrire des documents structurés transportables sur les protocoles communs d'Internet. Il constitue la technologie de base des architectures services web, il est un facteur important pour contourner les barrières techniques. En effet, il apporte à l'architecture l'extensibilité et la neutralité vis à vis des plates-formes et des langages de développement. De plus, grâce à la structuration, XML permet la distinction entre les données des applications et les données des protocoles permettant ainsi une correspondance facile entre les différents protocoles.

L'interopérabilité entre les systèmes hétérogènes demande des mécanismes puissants de correspondance et de gestion des types de données des messages entre les fournisseurs et les clients. Pour les services Web, on utilise systématiquement XML avec les Namespaces et la spécification XML Schema, tous deux indispensables pour exprimer les structures des données habituellement complexes figurant dans les messages échangés.

4.2. La communication : SOAP

Le formalisme XML étant basé sur du texte, il est tout à fait adapté pour être véhiculé par le protocole de communication HTTP. Pour assurer la

communication entre les participants d'un service Web, le couple HTTP/XML est donc naturellement utilisé. Les messages exprimés à l'aide de XML respectent un format standard définis par le W3C que l'on appelle le protocole SOAP (Simple Object Access Protocol).

Ainsi, dans le cadre des services Web, une application communique avec une autre application par l'intermédiaire du protocole SOAP/HTTP. Cela signifie qu'une application envoie un message SOAP (donc un message exprimé en XML) vers une autre application, que celle-ci traite la demande effectuée par ce message et renvoie un message de réponse à l'application appelante, basant sur le mécanisme RPC (Remote Procedure Call: appel de procedures localisées sur des machines distantes) qui simplifie la coopération entre applications en fournissant un mécanisme pour transmettre un appel de procédure vers une application distante offertes sur le Web. [DAN03]

Le protocole de communication HTTP

Le choix de HTTP est un élément très important car il illustre parfaitement la capacité de SOAP à s'adapter aux échanges sur Internet réutilisant une technologie largement déployée et acceptée. Le protocole HTTP est un protocole simple capable de s'accommoder à la fois de la qualité de service et des temps de latence très variables de l'Internet; ce que n'est pas le cas par exemple des protocoles d'environnements répartis tels que DCOM ou CORBA.

De plus, en s'appuyant sur HTTP, on ne se heurte pas aux problèmes de pare-feu (firewall) ou de configuration de réseau IP qui rendent parfois difficile le déploiement d'applications à objets répartis au-delà du périmètre du réseau d'entreprise. [CHA02]

Les messages SOAP

La communication par message constitue un point crucial dans toute architecture SOA. SOAP est au cœur de l'échange des messages entre applications sur le Web. Du fait qu'il est basé sur XML, il permet l'échange de données structurées indépendamment des langages de programmation ou des systèmes d'exploitation. Le protocole SOAP est une spécification XML qui définit un protocole léger d'échange de données structurées entre

services Web dans un environnement totalement distribué et hétérogène. Il est indépendant du contenu du message, à proprement parler, il laisse la responsabilité de l'interprétation aux couches de communication supérieures [GHM00]. Il se contente d'offrir la possibilité de structurer des messages destinés à des objectifs particuliers allant d'un simple échange de données jusqu'à l'appel de procédures à distance.

N'imposant aucun modèle de programmation spécifique, SOAP peut être employé dans tous les styles de communication : synchrone (s'appuyer sur HTTP) ou asynchrone (s'appuyer sur SMTP[1]), point à point ou multipoint, intranet ou Internet.

Modèle d'échange de message en SOAP

Les messages SOAP sont des transmissions fondamentalement à sens unique (unidirectionnel) d'un expéditeur à un récepteur (figure3). Lorsqu'une transmission d'un message commence (e.g. invocation d'un service web), un message SOAP (ou document XML) est généré. Ce message est envoyé à partir d'une entité appelée le SOAP sender, localisé dans un SOAP nœud, celui-ci permet d'appeler une ou plusieurs fonctions du service. Les paramètres des fonctions invoquées se situent à l'intérieur du message. Le message est transité par zéro ou plusieurs noeuds intermédiaires (SOAP intermediates) et le processus fini lorsque le message arrive au SOAP receiver. Le message est alors traité par le service Web avec les paramètres correspondants. Un message SOAP, de même structure, est retourné.
Mais les paramètres transportés représentent les résultats de la fonction invoquée.
Le chemin suivi par un message SOAP est nommé message path. [RIC04].

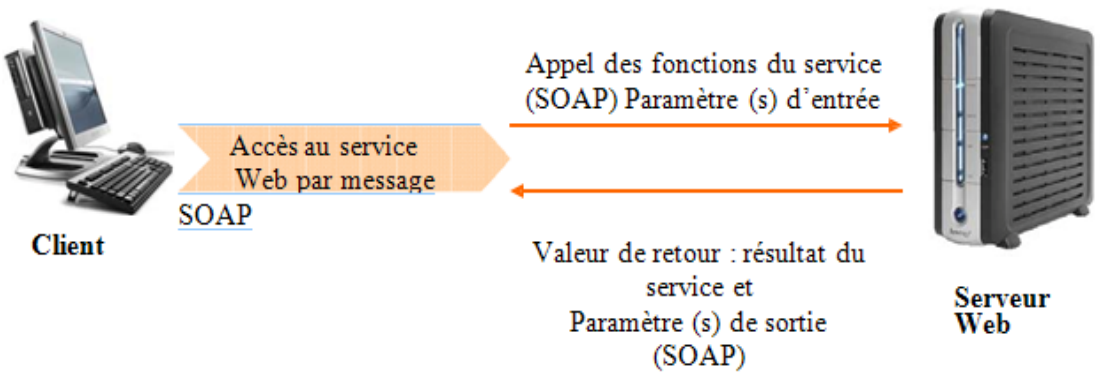

Figure 3 - Schéma de fonctionnement du système unidirectionnel SOAP

Enveloppe SOAP: Un message SOAP est un document XML composé d'un élément envelope qui est appelé "envelope SOAP", contenant un en-tête (header) facultatif et un corps (body) du message obligatoire. (Figure 4).

L'enveloppe définit l'espace de noms (namespace : URI permettant de connaître la provenance de chaque balise) précisant la version supportée de SOAP. Cet espace de noms est standard et permet de différentier les éléments du schéma. e.g:

"http://schemas.xmlsoap.org/soap/envelope/ ".

La seconde partie inclut l'espace de noms concernant les conventions de codage. Cette partie est facultative. Le document devra alors suivre le schéma défini dans cet espace de noms. Ce namespace peut être : " http://schemas.xmlsoap.org/soap/encoding/ ". La définition type d'une enveloppe est donnée par l'Extrait de code 1 :

```
<soap : Envelope  xmlns :soap =http://schema.xmlsoap.org/soap/envelope/
xmlns : enc= « http : //schemas .xmlsoap.org/soap/encoding/ »>
<soap :Header> … </soap :Header>
<soap :Body> …   </soap : Body >
                                        </soap>
```

Extrait de code 1 définition type d'envelope SOAP

SOAP Header : l'en-tête de message SOAP, est le premier fils de l'élément envelope. Même s'il peut être vide, il doit impérativement être écrit. Cet élément apporte des données supplémentaires à SOAP. Ces données peuvent être des informations d'authentification, de gestion de transactions, de paiement, etc.

SOAP Body : l'élément Body contient l'information destinée au receveur. Il faut que cet élément encadre une balise contenant le nom de la méthode invoquée (pour une requête), ou le nom de la méthode suivie de Response (pour la réponse). Cette balise doit aussi contenir l'espace de noms correspondant au nom de service.

Un message SOAP peut aussi contenir un ou plusieurs attachements (document, images, etc.) à transmettre avec le message. Ces données ne sont pas représentables en XML.

De ce fait, SOAP utilise un mécanisme d'inclusion appelé MIME (Multipurpose Internet Mail Extensions), cette méthode est répandue pour transmettre des documents autres que du texte dans des courriers électroniques.

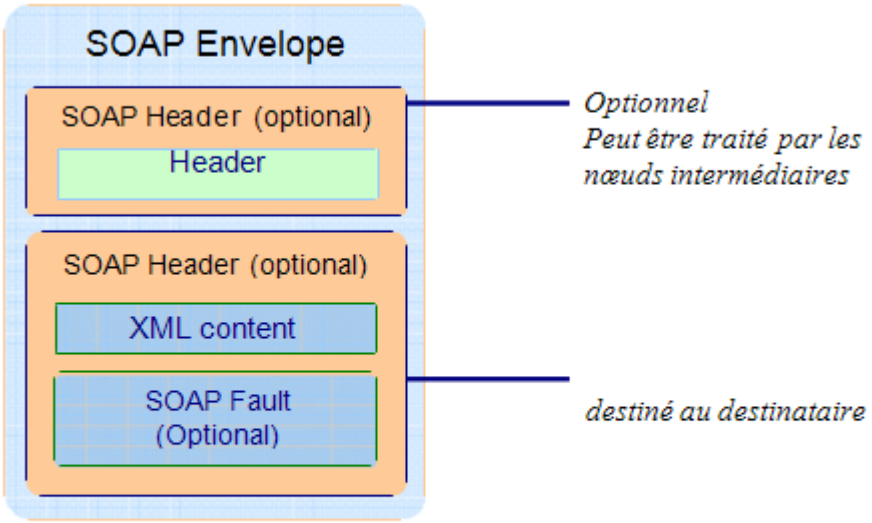

Figure 4 - La structure des messages SOAP

Pour résumer, un message SOAP en cours de transit est composé de deux parties indépendantes :

- une structure XML qui constitue le message SOAP;
- un en-tête du protocole de transport où le message SOAP est encapsulé en vue d'être livré à l'application destination;

Nous pouvons cependant ajouter que le protocole SOAP n'intègre pas la sémantique du message. Pour remédier à cela, le langage RDF technologies du Web sémantique, permet de supporter les échanges de messages.

4.3. Description de service : WSDL

Le protocole SOAP met à la disposition des services Web un moyen standard de structuration et d'échange de messages XML. Il ne fournit aucune indication sur la structure que le message doit respecter vis à vis du service Web sollicité. C'est toujours dans le but de rendre les services Web faiblement couplés et autonomes, que la spécification WSDL a vu le jour. Contrairement aux architectures monolithiques où la description des composants ainsi que les moyens de les invoquer dépendent fortement de l'infrastructure utilisée, la spécification WSDL offre une grammaire qui décrit l'interface des composants services Web de telle façon qu'ils se suffisent à eux-mêmes.

WSDL est un langage de description des capacités de services Web basé sur XML. Un

document WSDL décrit essentiellement le nom de la méthode utilisé, son nombre de paramètres, et leur type, ce qu'un service Web offre, où il réside et comment on peut l'invoquer. Nous pouvons dire que les services Web sont auto descriptifs. [BHI05]

La spécification du service est composée de deux parties : une définition abstraite des services en terme de messages échangés entre les différents types de port et la définition des mécanismes de liaison entre les définitions abstraites et un ensemble de techniques de déploiement (généralement des protocoles Internet).

Un document WSDL est devisé en sept sections distinctes (Figure 5) :

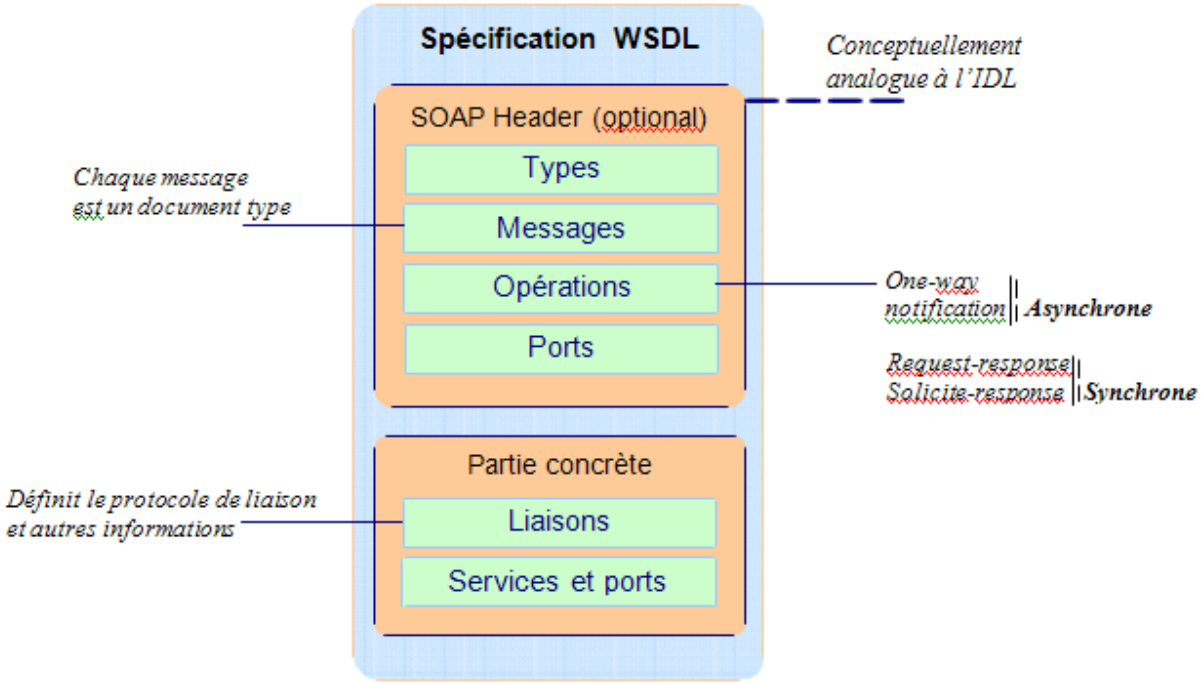

Figure 5 - Structure d'une interface WSDL

types : Fournissent des définitions de types de donnée afin de décrire les messages échangés.

message : Par l'intermédiaire de cette section, on définit le format des messages échangés, Un message correspond aux données qui seront véhiculées selon les méthodes invoquées. Chaque méthode du service possède deux éléments message, le premier correspond à la requête et le second correspond à la réponse. Chacun est constitué d'une ou plusieurs parties (sous-éléments part) décrivant un élément du message. La description contient le nom de l'élément en paramètre d'entrée ou de sortie selon le message et son type.

type de port (portType) : Ensemble d'opérations abstraites. Chaque opération se réfère à un message entrant et à des messages sortants. Chaque opération est identifiée dans le document WSDL par l'élément operation déclarant le nom de la méthode et les données passées en paramètres. Ces opérations sont regroupées dans un élément portType. Le document WSDL est constitué d'autant d'élément portType que le service contient de groupe de méthodes.

opération : Décrit les opérations invoquées à l'aide des messages reçu, émit par le service et éventuellement des messages d'erreur.

port : Ce qui spécifie une adresse pour un rattachement, déterminant ainsi un seul nœud branché, ou point terminal [1] (endpoint).

rattachement (binding) : Ce qui spécifie les aspects concrets de protocole de communication et le format des données pour les opérations et messages définis par un type de port particulier.

service : Représente une collection de points d'entrée (endpoint) relatifs, il sert à regrouper un ensemble de ports.

La spécification WSDL joue un rôle important dans l'interopérabilité des composants services Web. Moyennant un schéma uniforme obéissant à une sémantique bien définie, elle permet aux composants de définir ce qui est nécessaire à leur invocation. 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 au composant d'interagir même si l'application a été modifiée 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.

4.4. Service recherche et publication : UDDI (Universal Description, Discovery and Integration)

Les deux standards exposés précédemment définissent ensemble l'aspect le plus

basique de 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 [BCR02].

UDDI est un annuaire mondial d'entreprises s'appuyant sur le réseau Internet. Il permet d'automatiser les communications entre prestataires, clients, etc. Dans ce but, celui-ci propose plusieurs entrées : nom, carte d'identité des sociétés, description des produits et services. Et, si ces derniers sont des applicatifs invocables à distance, il fournit les références des connexions permettant de communiquer avec eux.

UDDI est une spécification qui définit les mécanismes qui permettent aux entreprises de publier leurs services et de découvrir et interagir avec d'autres services via le Web. UDDI se base sur SOAP et suppose que les requêtes et les réponses sont des objets UDDI envoyés comme des messages SOAP [BHI05].

L'enregistrement des services Web dans un annuaire UDDI s'effectue auprès d'un opérateur en accédant au site Web de ce dernier à partir d'un navigateur ou d'un outil intégré à un environnement de développement. Des recherches précises peuvent s'effectuer dans l'annuaire par catégories d'entreprise en utilisant des standards taxinomiques d'identification d'entreprise et par mots clés.

La spécification UDDI constitue une norme pour les annuaires des services Web. 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. Elle offre aussi une API aux applications clientes, pour consulter et extraire des données concernant un service et/ou son fournisseur (Interrogation de service).

Les registres UDDI sont des services Web qui offrent deux fonctionnalités de base : la publication des différents types d'information sur les services et leurs fournisseurs selon un schéma de description, et la consultation du contenu des registres.

La publication d'un service chez un opérateur, donne lieu automatiquement à un processus de propagation des informations aux différents registres UDDI. L'accès à l'ensemble d'informations des registres peut se faire de n'importe quel opérateur UDDI. La recherche se fait grâce à un moteur de recherche intégré au site de l'opérateur UDDI choisi. Ce moteur de recherche permettra d'affiner la recherche selon plusieurs critères : Nom de l'entreprise, la localisation de l'entreprise, identifiant de l'entreprise, le nom du service Web …etc.

Chaque registre UDDI stocke trois sortes de données : des données concernant les fournisseurs de services telles que le nom de l'entreprise, ses coordonnées et des descriptions de l'entreprise consultable par l'utilisateur appelées pages blanches, des données concernant l'activité ou le service métier des fournisseurs, la description des services Web déployés par les entreprises appelées pages jaunes et les données techniques de chaque service publié qui constituent les pages vertes. La spécification UDDI offre un schéma qui structure d'une manière uniforme les différents types concernant les trois niveaux de description.

5. L'intérêt d'un Service Web :

Les services Web fournissent un lien entre applications. Ainsi, des applications utilisant des technologies différentes peuvent envoyer et recevoir des données au travers de protocoles compréhensibles par tout le monde.

Les services Web sont normalisés car ils utilisent les standards XML et HTTP pour transférer des données et ils sont compatibles avec de nombreux autres environnements de développement. Ils sont donc indépendants des plates-formes. C'est dans ce contexte qu'un intérêt très particulier a été attribué à la conception des services Web puisqu'ils permettent aux entreprises d'offrir des applications accessibles à distance par d'autres entreprises. Cela s'explique par le fait que les services Web n'imposent pas de modèles de programmation spécifiques. En d'autres termes, les services Web ne sont pas concernés par la façon dont les messages sont produits ou consommés par des programmes. Cela permet aux vendeurs d'outils de développement d'offrir différentes méthodes et interfaces de programmation au-dessus de n'importe quel langage de programmation, sans être contraints par des standards comme c'est le cas de la plate-forme *CORBA* qui définit des ponts spécifiques entre le langage de définition IDl et différents langages de programmation. Ainsi, les fournisseurs d'outils de développement peuvent facilement différencier leurs produits avec ceux de leurs concurrents en offrant différents niveaux de sophistication.

Les services Web représentent donc la façon la plus efficace de partager des méthodes et des fonctionnalités. De plus, ils réduisent le temps de réalisation en permettant de tirer directement parti de services existants.

6. Conclusion

De plus en plus, avec l'essor d'Internet, le développement tend vers les technologies du Web. Les services Web sont des composants logiciels représentant une fonction applicative, ils représentent un mécanisme de communication entre applications distantes à travers le Web. Les services Web permettent le changement de la nature du Web, du Web du document utilisé par les organisations pour la publication des informations, au Web orienté service, qui permet aux serveurs d'applications la communication entre eux. Les services Web peuvent être utilisés par les humains ou programmes afin d'accomplir une tâche donnée.

Les services Web sont suffisamment développés pour que les développeurs les utilisent maintenant dans tous les domaines de l'informatique, afin de récolter les divers bénéfices de la technologie. La diversité de ces infrastructures et des organismes qui les déploient entraîne des hétérogénéités.

Les services Web sémantiques qui fera l'intérêt du prochain chapitre s'intéressent à l'automatisation de l'utilisation des services Web, en ajoutant des annotations à ces services afin d'améliorer l'expression sémantique de leurs fonctionnalités.

Chapitre II

Les services Web sémantiques

II. Chapitre 2 : Les services Web sémantiques

1. Introduction :

L'absence de la sémantique des technologies de base des services Web. Ce problème est posé avec la description en WSDL d'un service Web, ne permet pas d'automatiser les diverses tâches liées aux services Web, par exemple, la publication, l'invocation, la découverte, la composition, . . . etc.

Pour résoudre ce problème, il faut enrichir les descriptions des services Web par d'autres informations complémentaires, réutilisables, partageables et compréhensibles par les machines. Ces informations sont des données et métadonnées, qui permettent d'interpréter les descriptions des services Web. En d'autres termes, c'est la sémantique des descriptions des services Web.

Les recherches faites sur le Web sémantique qui exploite dans le domaine des services Web, pour donner naissance à la nouvelle technologie importante dans le domaine de technologie de l'Internet, qui s'appelle les services Web sémantiques.

Le Web sémantique constitue le point de départ pour le développement futur de services Web. Les services Web sémantiques sont une évolution nécessaire des services Web. Ils se basent totalement sur les langages du Web sémantique, en particulier sur les ontologies.

2. Présentation et Objectifs des services Web sémantiques :

Partage, Echange, Réutilisation ; Ces trois mots définissent à eux seuls le Web. Le Web sémantique constitue une prolongation, et une sorte de révolution de fond du Web actuel qui permet une définition non ambiguë de l'information, pour favoriser une meilleure coopération entre humain et machine. Il permet de s'ouvrir à de nouvelles possibilités d'automatisation d'une grande quantité d'information sur le Web.

Proclamé technologie du futur, en 2001, par son créateur Tim Berners-Lee, le Web sémantique propose une nouvelle plateforme permettant une gestion plus intelligente du

contenu, à travers sa capacité de manipuler les ressources sur la base de leurs sémantiques. En réalité, l'intégration de la sémantique au Web n'est pas une nouvelle idée mais au contraire, elle est née avec le Web [Falquet 2001].

Dans l'architecture en couche des technologies du Web sémantique, présentée précédemment, on a présenté les technologies Web de base. Puis, on a parlé des services Web comme un moyen qui fournit une plateforme interopérable, pour l'intégration des applications en utilisant des technologies Web (Voir la Figure 6).

Le Web sémantique constitue le point de départ pour le développement de services Web intelligents.

En effet, non seulement l'humain pourra partager, échanger et réutiliser la connaissance et l'information qui est disponible sur le Web mais en plus, il pourra le faire plus vite et avec l'aide des machines. HTML a été le langage du Web jusqu'à présent. Il permettait de présenter l'information aux humains. Désormais, il est nécessaire de présenter cette information de façon à ce qu'un programme puisse s'en servir. Le Web sémantique a promit de permettre aux machines de tirer parti du contenu statique du Web, en utilisant les annotations. En effet, la sémantique et la structure des données requirent une représentation de la sémantique compréhensible et échangeable par les machines [Bru¼n 2007].

Le terme services Web sémantique est réservé pour l'automatisation des tâches d'utilisation des services Web, tels que : la publication, la découverte, la composition . . . etc. Ils se trouvent à la convergence de deux domaines de recherche importants concernant les technologies de l'Internet ; le Web sémantique et les services Web (Voir la Figure 6) [Cardoso 2007].

Cette tâche de convergence est accomplie en rendant les services Web auto-exploitables par machines, et de réaliser l'interopérabilité entre les applications via le Web en vue de rendre le Web plus dynamique.

De la même façon, que le Web sémantique a promit de considérer comme un vaste espace d'échange de ressources entre humains et machines, permettant une meilleure

exploitation de masses de données disponibles sur le Web. L'objectif est non pas de permettre aux machines de se comporter comme des êtres humains, mais de développer des langages pour représenter les informations d'une manière traitable, représentable et intelligible par les machines, afin, d'améliorer les rapports des utilisateurs avec le Web.

Il semble donc nécessaire de tendre vers des services intelligibles pour des machines, c'est le concept de service Web sémantique [Kadima 2003].

Les avantages potentiels des services Web sémantiques ont mené à l'établissement d'un domaine de recherche important, dans le milieu industriel et académique. Plusieurs, initiatives sont apparues pour faire ce qu'on appelle l'annotation sémantique des services Web, ce qui a produit une variété de descriptions des services Web et leurs aspects relatifs, ce qui en retour a abouti à de divers genres de support pour la découverte et la composition (Voir la Figure 6).

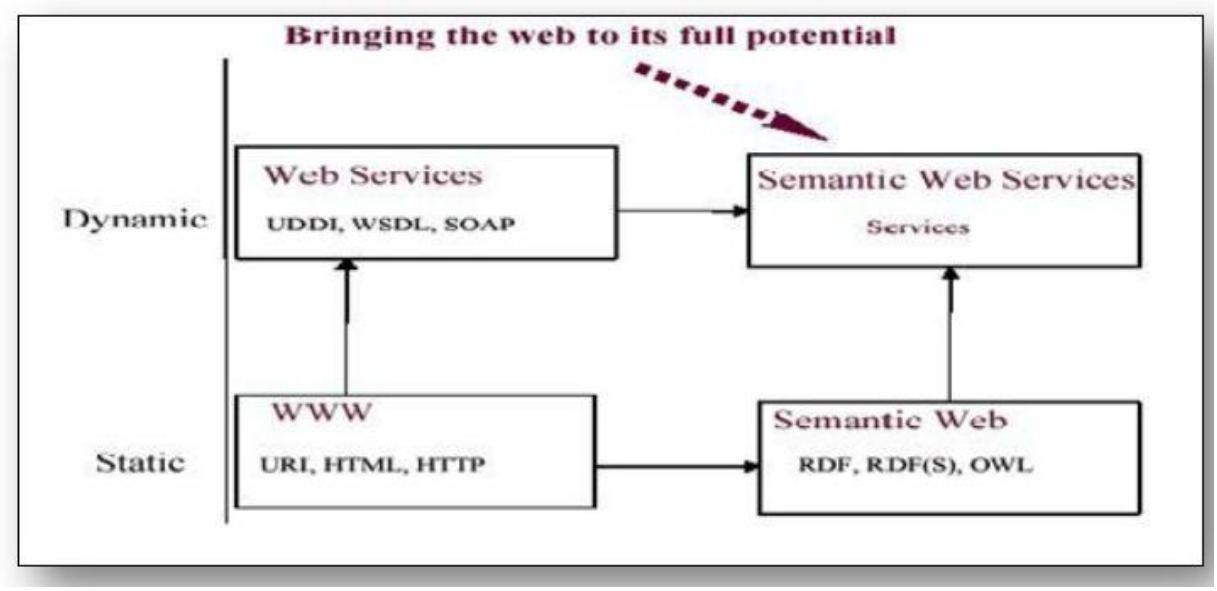

Figure.6 – L'évolution du Web.

Le concept fondamental du Web sémantique et des services Web sémantique est l'ontologie, qui produit une signification bien définie des informations contenu dans le Web. Une ontologie représente donc un schéma conceptuel, qui tente de désigner une description rigoureuse et exhaustive d'un domaine. Habituellement, une ontologie est une structure de données hiérarchique qui comprend toutes les entités du domaine que l'on tente de décrire,

ainsi que, les relations sémantiques qui existent entre ces différentes entités. Mais attention, une ontologie se doit d'être plus qu'une simple taxonomie [Bru¼n 2007].

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

3. Approches proposées pour les services Web sémantiques :

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

3.1 WSDL-S :

WSDL-S est un langage de description sémantique des services Web. Une description WSDL-S de service Web est une description WSDL augmentée d'annotation sémantique. Les annotations sémantiques peuvent être des références à des concepts définis dans des ontologies externes. WSDL-S ne prescrit aucun langage particulier d'ontologies, et il est défini pour être lié au langage de représentation sémantique.

WSDL-S est issu du projet METEOR-S de l'université de Georgia. WSDL-S définit un modèle sémantique pour capturer les termes et les concepts utilisés pour décrire et représenter la connaissance, cette sémantique est ajoutée en deux étapes [Cerami 2002] :

- ✓ La première étape consiste à faire référence, dans la partie définition de WSDL, à une ontologie dédiée au service à publier ;

- ✓ La deuxième étape consiste à annoter les opérations de la définition WSDL de sémantique, en ajoutant deux nouvelles balises ; la balise **« Action »** qui permet de

représenter l'action de l'opération et la balise **« Contrainte »** qui représente les prés et post conditions d'une opération.

Quatre rôles du modèle sémantique sont distingués :

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

3.2 OWL-S :

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

OWL-S succède aux travaux antérieurs de DAML-S qui étaient basés sur DAML+OIL. OWL-S décrit un service à l'aide des trois classes suivantes (Voir la Figure 7) [Chappell 2002] [Chatterjee 2003] :

_ **ServiceProfile :** Chaque instance de **« Service »** produit zéro ou plusieurs profils de services.

Un service Profile exprime **« Que fait un Service »**, aux fins des avertissements et sert comme un Template pour les requêtes de services, permettant ainsi la découverte et leurs arrangements.

OWL-S fournie cette classe pour décrire un service Web. Cette classe **« ServiceProfile »** spécifie trois informations :

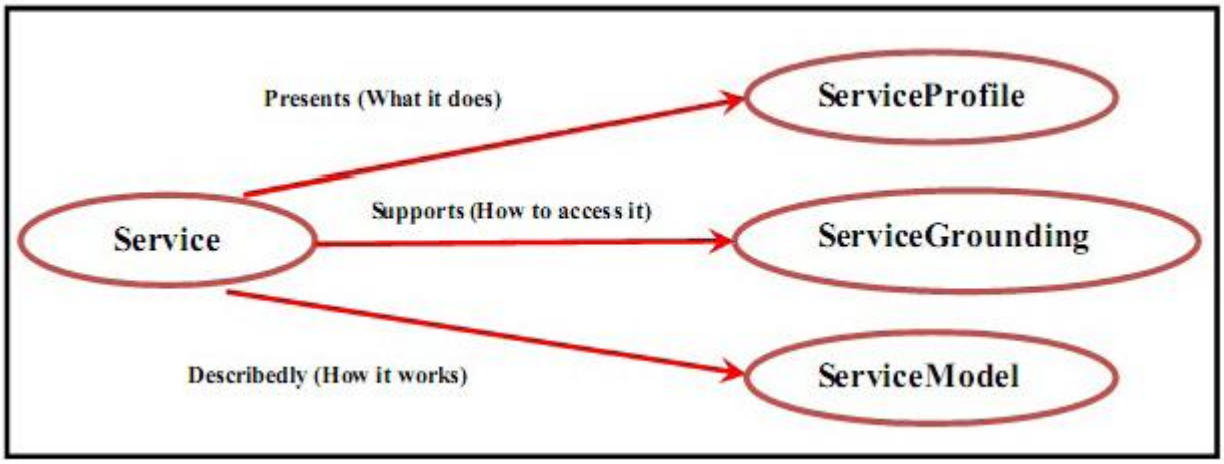

Figure 7 – Structure générale de l'ontologie OWL-S.

- **Le fournisseur du service :** Cette information précise les données nécessaires pour identifier et contacter le fournisseur du service Web.
- **La description fonctionnelle :** Elle spécifie ce que l'on peut attendre du service en termes d'entrées attendues et de résultats produits en sortie. Les transformations d'informations sont représentées par des **« Inputs »** et des **« Outputs »**. Le changement d'état du monde réel causé par l'exécution du service est représenté par **« preconditions »** et **« effects »** qui sont les préconditions et les postconditions de son exécution. Les **« Inputs »** et les **« Outputs »** font références à des classes d'OWL décrivant les types des instances à envoyer au service et aux réponses respectives attendues. Le format des préconditions et leurs conséquences n'est pas fixe, mais les auteurs de OWL-S ont permis aux divers formats d'être intégrés dans une ontologie OWL.
- **Propriétés additionnelles :** Plusieurs propriétés sont utilisées pour qualifier le service Web. La première est la catégorie du service Web. La seconde est la qualité de service Web **« QoS »**. Enfin, le service Web peut fournir une liste de paramètres de façon libre.

_ **ServiceModel :** définit le fonctionnement du service Web. Les services Web peuvent être modélisés avec OWL-S en tant que processus. La classe ainsi définie est **« Process »**, qui est une sous classe de **« ServiceModel »**. Pour décrire un processus, on spécifie ses entrées et

sorties. Les transitions d'un état à un autre sont décrites par les préconditions et les effets de chaque processus. Un model peut être décrit par le lien **« descriptedBy »**, un model de service qui expose comment un service fonctionne. Le modèle de service voit les interactions du service comme des processus. Un processus n'est pas nécessairement un programme à exécuter, mais plutôt des spécifications, qui permettent à un client d'agir avec un service.

OWL-S différencie les processus en processus atomiques, processus simples, et processus composés. Les processus atomiques sont des opérations simples. Ils représentent des services, qui sont directement appelés. Ils ont une unique appellation, et ils sont directement liés avec le **« Service-Grounding »**. Tandis que, les processus simples sont vus comme des processus atomiques composé ne possédant pas de liaison avec le **« ServiceGrounding »**. Même, si un service simple est considéré comme atomique, il ne peut pas être appelé directement. En effet, il a besoin d'un service atomique pour fournir un résultat. Dans ces conditions, un service simple a toujours besoin d'un service atomique en deuxième plan pour fonctionner.

Des processus composés sont accumulés des processus atomiques, ou simples par des constructions standard deWorkflow telles que la séquence, la décomposition **« Split »** ou la composition **« Join »** pour déterminer le flux de contrôle, plus des informations additionnels sur le flux de données.

_ **ServiceGrounding :** Définit les détails techniques permettant d'accéder au service Web publié, tels les protocoles, les URIs, les messages envoyés, . . . etc. Pour cela, il fournit les détails pour connecter la spécification abstraite et la spécification concrète. Il est effectivement nécessaire d'avoir une combinaison entre le grounding et WSDL. Le fonctionnement du grounding est assuré si et seulement si, les deux entités (WSDL et Grounding) sont présentes et correspondent. Normalement, les concepts de grounding sont compatibles aux concepts de **« binding »** du WSDL. Les deux classes **« ServiceProfile »** et **« ServiceModel »** d'une description OWL-S s'attachent à abstraire la représentation d'un service Web. Par contre, la classe **« ServiceGrounding »** est la forme concrète d'une représentation abstraite.

3.3 IRS-II :

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

- **Ontologie du domaine (Domain Model) :** Permet de décrire le domaine d'une application ;
- **Ontologie de tâche à résoudre (Task Models) :** Fournit une description générique de la tâche à résoudre, spécifie les types d'entrées et sortie, le but à atteindre et les préconditions à satisfaire ;
- **Ontologie des méthodes de résolution d'un problème (Problem Solving Methods):**

Sépare la description de ce qu'un service fait des paramètres et des contraintes d'une mise en œuvre particulière ;

- **Liens (bridges) :** Permettent la correspondance entre les différents modèles d'une application.

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

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

Le composant éditeur (IRS-II Publisher) a deux fonctions. Premièrement, il permet de lier les services Web à leurs descriptions sémantiques respectives dans le serveur. Chaque PSM est associé à exactement un service Web. Un service Web peut être associé à plusieurs PSM, puisque un service Web peut avoir plus qu'une fonction. Deuxièmement, il génère automatiquement un programme qui enveloppe le code LISP ou Java du service Web, afin de

l'invoquer, comme c'est le cas d'un service Web dans sa description WSDL. Un client IRS-II invoque un service Web en envoyant une requête de la tâche à traiter. Sur la base de cette tâche le serveur sélectionne le PSM approprié, et invoque le service Web avec lequel est lié ce PSM.

3.4 WSFM :

WSFM comprend quatre éléments principaux [Gardien 2002] [Dallons 2004] :

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

L'implémentation de WSMF est partagée en deux projets : le projet SWWS; et le projet WSMO.

L'objectif de SWWS est de définir une structure de description et de découverte de services Web, ainsi qu'une plateforme de médiation pour services selon une architecture conceptuelle.

Le projet WSMO permettra de raffiner WSMF, en plus, du développement d'ontologie formelle de service et un langage pour les services Web sémantiques [Casati 2000].

3.5 WSMO :

Le WSMO est un projet de l'union européenne qui constitue un cadre compréhensible pour SESA et défini un model conceptuel avec un langage de spécification, comme, il fourni une implémentation avec plusieurs outils. L'implémentation WSMX fournie un environnement de développement et d'exécution pour SESA à base de WSMO. L'approche WSMO définie les ontologies, les services web, les buts et les médiateurs comme ses éléments

de haut niveau avec un model conceptuel qui prend en charge ces derniers. Ce model conceptuel a pour but la structuration des annotations sémantique des services.

WSMO permet la description des services, en considérant les aspects suivants [Newcomere 2004] :

- Les propriétés non fonctionnelles incluent des propriétés telles que la performance, la fiabilité, la sécurité, la robustesse et la scalabilité du service Web ;
- La fonctionnalité du service est décrite en termes de préconditions, postconditions, hypothèses et effets ;
- Une interface de description d'un service Web est constituée d'informations telles que les erreurs gérées par le service, une description d'orchestration si le service fait appel à d'autres services, une description de la conversation du service appelée échange de message, et des stratégies de compensations utilisées dans le cas ou certaines opérations exécutées par le service doivent être annulées ;
- Le grounding spécifie les informations concrètes pour l'accès au service Web. Il est clair que de nombreux aspects définis dans cette ontologie sont également couverts par OWL-S. Par exemple, les préconditions, postconditions, hypothèses et effets dans WSMO. D'autre part, les deux ontologies se basent sur WSDL pour la liaison. Un autre point commun est la description de la conversation d'un service dans les interfaces WSMO qui est également décrite dans le process Model de OWL-S.

  Ceci dit qu'il existe quelques différences entre les deux approches.

Contrairement à OWL-S qui n'exige aucun style architectural, l'approche proposée par WSMO est basée sur l'architecture WSMF. Cette architecture définit notamment la notion de médiateurs. Les médiateurs sont des composants logiciels utilisés par les services Web pour répondre à des problèmes d'interopérabilité tels que l'incompatibilité des types. D'autre part, dans WSMO, certaines propriétés telles que les hypothèses et effets sont exprimées dans le langage formel F-Logic qui permet le raisonnement automatique sur les valeurs de ces propriétés. Dans OWL-S, des propriétés telles que les préconditions et effets doivent

également être exprimées sous forme d'expressions logiques. Ceci dit, à l'heure actuelle, ces expressions sont exprimées sous formes de structures XML [Burstein 2004].

Le Web sémantique est un projet qui a été créer pour la modélisation des données hétérogènes, par contre la modélisation des programmes disponibles sur le Web, c'est-à-dire les services Web constitue un nouveau projet relativement récent s'appelle les services Web sémantiques, ce dernier s'intéresse principalement d'ajouter de la sémantique aux services Web. Des services Web sémantiques seraient intuitivement des programmes dont les effets sur leur environnement seraient connus, et dont les données manipulées possèderaient aussi une sémantique. Ce projet peut être décomposé en trois parties [Bru¼n 2008] :

- Développer des ontologies pour décrire les données manipulées par les services Web ;
- Définir la sémantique des services Web ;
- Développer une ontologie de services Web.

Le premier point rejoint les préoccupations du Web sémantique. Le second point, l'idée que des machines puissent inférer automatiquement la sémantique des programmes n'est pas neuve. Le dernier point, définir une ontologie de Web services, a été développée avec OWL-S comme nous le verrons auparavant.

Les deux premiers points constituent des briques indispensables pour les services Web sémantiques. Le dernier point ne s'apparente pas exactement au second et s'il constitue le cœur du projet des services Web sémantiques, il est une conséquence du second point. En effet, disposer d'une ontologie de services Web n'implique pas forcément que toutes les facettes des services Web ont été examinées, cela pose le problème de la granularité de la description des services Web. Apparemment, une ontologie de services Web servira à préciser les relations des services Web entre eux [Bru¼n 2008].

4. Conclusion :

Un service Web sémantique est un service Web décrit, en utilisant des annotations sémantiques dans un langage du Web sémantique bien défini, qui permettent au service Web d'avoir une interface compréhensible par les humains et les machines. Ces services Web

sémantiques s'appuient en général sur les langages du Web sémantique, pour décrire leurs fonctionnalités et les données qu'ils échangent.

Les motivations pour développer, ou tendre vers les services Web sémantiques sont évidemment de faciliter les phases automatiques de découverte, sélection et composition de services Web. En effet, si leur sémantique est connue, alors chercher et composer des services Web pourra être fait automatiquement en donnant la sémantique cible. De plus, pour les mêmes raisons que pour le Web sémantique, il faut organiser les services Web entre eux. Une première solution pour les trier est que des machines puissent connaître leur sémantique. En effet, si la sémantique était connue, il suffirait d'indiquer sous forme, par exemple, de requête. Les objectifs que devrait remplir le service Web afin de pouvoir, par exemple, le composer [Grimm 2007].

Les avantages de l'utilisation du Web sémantique pour la description des services Web sont nombreux. Ce qui nous intéresse est l'automatisation, autant que possible, des divers aspects relatifs aux services Web, en particulier la composition des services Web sémantiques à base d'algorithmes génétiques, qui fera l'intérêt du prochain chapitre.

Chapitre III

Conception et implémentation de prototype

III. Chapitre 3: Conception et implémentation de prototype

1. Introduction

L'objectif de la composition des services web est de créer de nouvelles fonctionnalités en combinant des fonctionnalités offertes par d'autres services existants, composés ou non en vue d'apporter une valeur ajoutée en fonction d'une requête de l'utilisateur. Contrairement aux processus métier dans un environnement statique, les services web composés s'exécutent dans un environnement versatile où le nombre de services disponibles évolue très rapidement. Afin d'automatiser ce processus, il est primordial d'expliciter la sémantique dans les descriptions des services web soit par l'ajout d'annotations sémantiques ou en utilisant des ontologies de haut niveau tels qu'OWL-S ou WSMO, évitant ainsi les problèmes d'hétérogénéité sémantique qui peuvent survenir.

Notre contribution consiste à proposer une approche pour la composition dynamique des services web sémantiques basée sur un algorithme génétique, ainsi nous modélisons ce problème comme un processus d'optimisation, le but est de trouver une solution (une combinaison de services web) qui répond au mieux à la requête de l'utilisateur.

Pour évaluer le degré de similarité entre une requête donnée et un service composite nous proposons un mécanisme de matching (appariement) sémantique basé sur les concepts ontologiques annotant les paramètres d'entrée/sortie des services web participant à la composition ainsi que les concepts ontologiques annotant les paramètres d'E/S de la requête.

Ce chapitre est organisé comme suit : nous commençons d'abord par présenter notre approche de composition basée sur la technique des AG's, ensuite nous présentons notre modèle sémantique qui décrit le mécanisme de matching sémantique.

Dans la dernière section nous présentons notre AG en détail (codage, opérateurs génétiques, fonction fitness,…..) ainsi que les résultats obtenus.

2. Architecture de composition proposée

Notre approche de composition se base sur un algorithme génétique mono-objectif qui prend en compte l**'aspect fonctionnel :** où les descriptions sémantiques des services web sont

exploitées pour calculer le degré de correspondance sémantique entre la requête de l'utilisateur et une solution candidate (service composite).

Notre intérêt pour les algorithmes génétiques est du à deux raisons :

- Les algorithmes génétiques ont prouvé leur efficacité dans divers domaines pour résoudre des problèmes d'optimisation NP hard (comme le problème de voyageur de commerce).
- Ils n'imposent aucune hypothèse sur la résolution d'un problème donné, ainsi leur principe consiste à définir une fonction objective (à optimiser) pour trouver une solution de bonne qualité.

L'AG reçoit en entrée les besoins de l'utilisateur exprimés via une interface graphique, ces besoins incluent une requête exprimée en termes de concepts d'E/S et il retourne comme résultat une solution quasi-optimale qui représente une composition des services web (voir Figure 8).

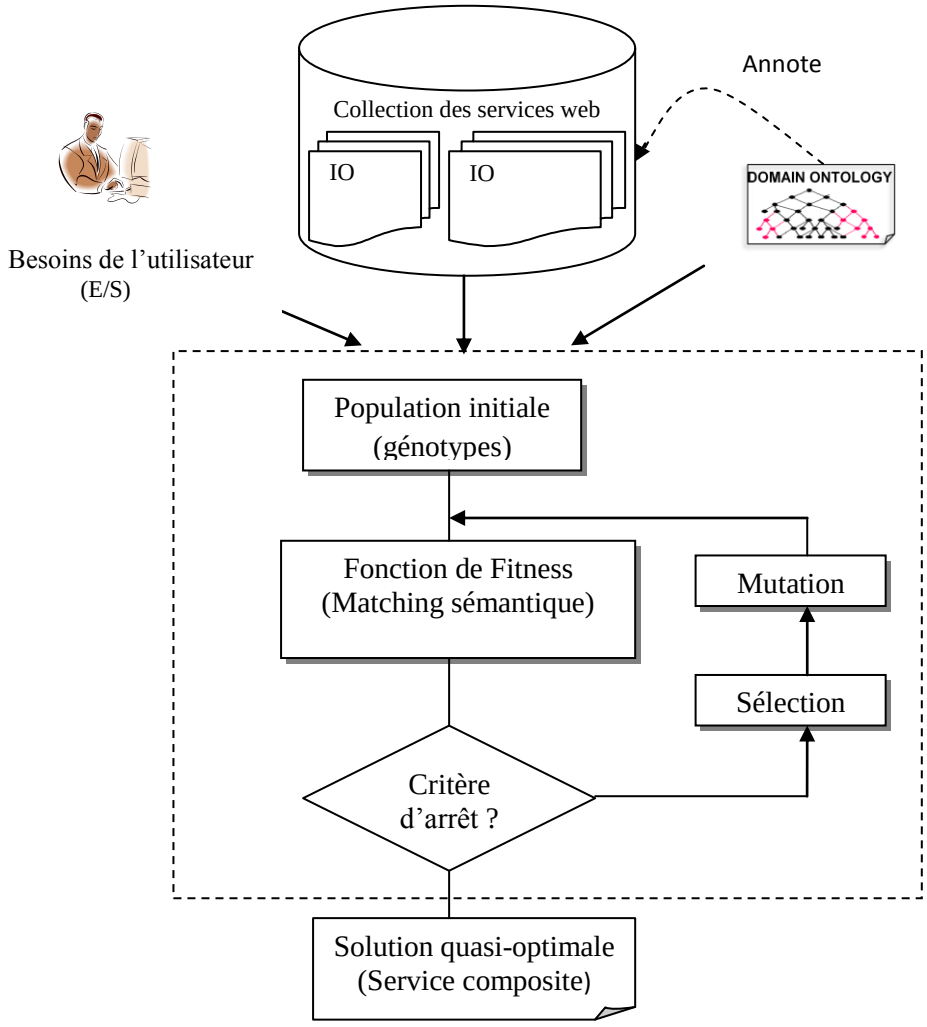

Figure 8 : Architecture de composition proposée

3. Le modèle sémantique (Modèle fonctionnel)

Représentation conceptuelle

Avant de parler de la similarité sémantique d'un service composite il est nécessaire de donner quelques définitions formelles d'un service web, un service composite ainsi que la requête de l'utilisateur.

Définition 1 (service web)

Un service web est caractérisé par des propriétés fonctionnelles.

Formellement :$S = \{ID, CE, CS\}$

Où *ID* est l'identifier du service S, *CE* et *CS* sont les concepts d'entrée et de sortie respectivement.

Définition 2 (service composite)

Un service web composite (SC) est définit par un ensemble de services web. Formellement $SC = \{S_1,\ S_2, \dots, S_n\}$ ou *n* est le nombre des services web.

Définition 3 (requête)

Les besoins de l'utilisateur sont définis comme suit:

$$\mathrm{R} = \{\ \mathrm{CE}, \mathrm{CS}\ \}$$

Où *CE et CS* représentent respectivement les concepts d'entrée et de sortie de la requête R.

Calcul de degré de similarité sémantique d'un service composite

Introduction

Le degré de similarité sémantique entre une requête donnée et un service composite est déterminé en fonction de la sémantique des éléments des descriptions à apparier. Selon la littérature on a principalement trois approches de matching [Chabeb, 2011]:

- ***IO-matching*** *:* dit aussi "IO-matching de profil de service". Ce type d'appariement est déterminé à partir des données sémantiques des paramètres de service : les entrées : inputs (I) et les sorties : outputs (O). Ce type d'appariement est adopté dans [Paolucci et al., 2002], [Fan et al., 2005] et [Paolucci et al., 2004].

- ***PE-matching :*** Ce type d'appariement est déterminé à partir d'appariements sur des pré-conditions (P) et des effets (E) des services et des requêtes. Ce type d'appariement est adopté dans PCEM [Schumacher et al., 2008] avec des pré-conditions et des effets spécifiées en Prolog.
- ***IOPE-matching :***Ce type d'appariement est déterminé à partir d'appariements sur les données sémantiques des inputs (I), des outputs (O), des pré-conditions (P) et des effets (E) des services et des requêtes. Cet appariement est adopté dans [Jaeger et al., 2005], [keller et al., 2005] [Küster et al., 2008]et [Stollberg et al., 2007] .

Dans notre approche on a adopté un *IO-matching* qui est basé sur le raisonnement par *subsomption*. Ce mécanisme estime le degré de *similarité* entre la requête de l'utilisateur et le service composite.

Afin de réaliser le processus de l'appariement, nous utilisons les quatres types de matching *Exact, Plugin, Subsume et Fail* introduits dans [Paolucci et al., 2002], sachant que dans la découverte des services web, l'appariement est effectué sur les mêmes catégories de paramètres, c'est à dire sur les paires d'entrées, ou les paires de sorties. A l'opposé, pour la composition de web services, le problème est adressé par l'étude de l'appariement sur des paires distinctes de paramètres d'entrée et de sortie des différents services web.

- ***Exact* :** deux paramètres sont dits similaires si leurs concepts C1 et C2 sont équivalents : $\boldsymbol{C1 \equiv C2}$
- ***Plugin* :** deux paramètres sont dits similaires si le concept C1 du premier paramètre est subsumé par le concept C2 du deuxième paramètre(C1 est plus spécifique que C2): $\boldsymbol{C1 \subseteq C2}$.Considérons le fragment d'ontologie Figure 5.2 et deux services respectivement nommés **NiveauScolaire_ManuelBiologie** et **ManuelSecondaire_Prix**. Le premier service prend en entrée un niveau **secondaire** et fournit une liste de manuels de **biologie**. Le deuxième service prend en entrée tous les types de **manuels** scolaires et en fournit le **prix**. Le premier service peut entrer en interaction avec le second dans une relation plugin car le concept manuel est plus général que le concept biologie.
- ***Subsume* :** le deux paramètres sont dits similaires si le concept C1 du premier paramètre est plus général que le concept C2 du deuxième paramètre (C1 subsume C2) :$\boldsymbol{C1 \supseteq C2}$. Considérons le même fragment d'ontologie Figure 5.2et deux

services définis comme suit. Le premier service nommé **NiveauSecondaire_ManuelTechnologie** prend en entrée le niveau scolaire **secondaire** et fournit une liste de manuels de **technologie** (biologie, informatique, etc.). Le deuxième service nommé **ManuelInformatique_Prix** prend en entrée des manuels d'**informatique** et en fournit le **prix.** Le premier service peut entrer en interaction avec le second dans une relation subsume car le concept informatique est plus spécifique que le concept technologie.

- ***Fail* :** signifie qu'il n'y a aucune relation de subsomption entre les concepts. Soient les deux services suivants décrits sur la base de l'ontologie Figure 5.2. Le premier service **NiveauSecondaire_ManuelSecondaire** fournit tout type de **manuels** secondaires à partir du niveau fourni. Le deuxième service **ManuelUniversitaire_Tarif** fournit les **tarifs** de manuels **universitaires**. Dans cette situation, aucune interaction entre ces deux services n'est possible.

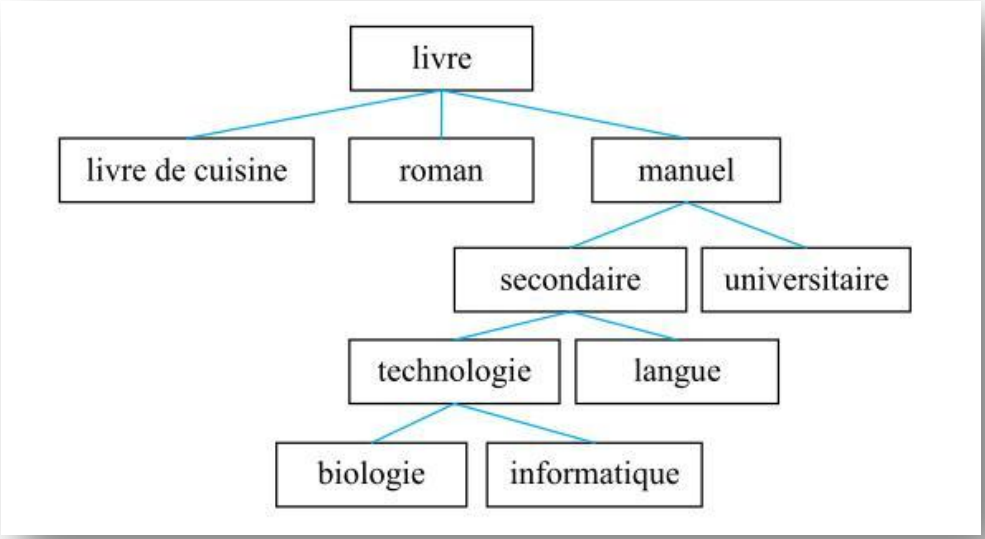

Figure 9 : Fragment d'ontologie relative aux livres

On définit 4 étapes pour obtenir la similarité sémantique entre une requête et une composition (SC) :

Etape 1 : calcul de la similarité entre le paramètre d'entrée (concept d'entrée) de la requête et le paramètre d'entrée (concept d'entrée) de SC selon la fonction ME_{RIN} (définit dans la section suivante).

Etape 2 : calcul de la similarité entre le paramètre de sortie (concept de sortie) de la requête et le paramètre de sortie (concept de sortie) de SC selon la fonction ME_{ROUT} (définit dans la section suivante).

Etape 3 : calcul de la similarité entre les concepts d'E/S des services qui composent la solution selon la fonction ME_S (définit dans la section suivante).

Etape 4 : le matching sémantique global est calculé à partir des scores obtenus lors des étapes 1,2 et 3 (il est définit dans la section suivante).

Les fonctions précédentes utilisent une fonction nommé ***SIM*** qui permet de calculer la similarité entre deux concepts C1 et C2, son pseudo code est donné par la suite :

$$\boldsymbol{SIM}(\boldsymbol{C1}, \boldsymbol{C2}) = \begin{cases} \boldsymbol{Exact} & \boldsymbol{if}\ \ \boldsymbol{C1} \equiv \boldsymbol{C2} \\ \boldsymbol{Plugin} & \boldsymbol{if}\ \ \boldsymbol{C1} \subseteq \boldsymbol{C2} \\ \boldsymbol{Subsume} & \boldsymbol{if}\ \ \boldsymbol{C1} \supseteq \boldsymbol{C2} \\ \boldsymbol{Fail} & \boldsymbol{Autrement} \end{cases} \quad \textbf{(1)}$$

Procédure de calcul de la similarité sémantique

Supposant qu'on a un service composite $SC = \{S_1,\ S_2, \dots, S_n\}$ et une requête $\mathrm{R} = \{\mathrm{CE}, \mathrm{CS}\}$, le calcul de la similarité sémantique passe par les étapes suivantes :

Etape 1 :

On calcule la similarité entre les concepts d'entrée de la requête et les concepts d'entrée de SC par la fonction ME_{RIN} /$ME_{RIN} = SIM(R.CE, S_1.CE)$.

Etape 2 :

On calcule la similarité entre les concepts de sortie de la requête et les concepts de sortie de SC par la fonction$\mathrm{ME_{ROUT}}$/ $\mathrm{ME_{ROUT}} = SIM(\mathrm{R.CS}, \mathrm{S_N.CS})$.

Etape 3 :

On calcule la similarité entre les concepts d'E/S des services composants par la fonction ME_S/ $ME_S = SIM(S_i.CS, S_{i+1}.CE)$.

Les fonctions $EM_S, EM_{RIN} et EM_{OUT}$ retournent des valeurs dans [0,1] en fonction de type de matching (exact, plugin, subsume, fail) entre les concepts. Dans le Tableau 1 nous récapitulons les fonctions de matching sémantique avec leurs scores.

Type de matching	**Exact**	**Plugin**	**Subsume**	**Fail**
$ME_S(S_i.CS, S_{i+1}.CE)$	1	2/3	1/3	0
Description Logique	*S_i.CS et S_{i+1}.CE sont équivalent*	*S_i.CS est subsumé par S_{i+1}.CE*	*S_i.CS subsume S_{i+1}.CE*	Autrement
$ME_{RIN}(R.CE, S_1.CE)$	1	2/3	1/3	0
Description Logique	*R.CE et S_1.CE sont équivalent*	*R.CE est subsumé par S_1.CE*	*R.CE subsume S_1.CE*	Autrement
$ME_{ROUT}(R.CS, S_N.CS)$	1	2/3	1/3	0
Description Logique	*R.CS et S_N.CS sont équivalent*	*R.CS est subsumé par S_N.CS*	*R.CS subsume S_N.CS*	Autrement

Tableau 1 : Les fonctions de matching sémantique

Etape 4 :

Le matching global entre la requête et la composition SC se fait en calculant la moyenne des scores des 3 premières fonctions (Equation (2)).

$$\boldsymbol{MG(SC) = ((ME_{RIN}(R.CE, S_1.CE) + \sum_{i=1}^{N} ME_S(S_i.CS, S_{i+1}.CE) + (ME_{ROUT}(R.CS, S_N.CS))/((N-1)+2)} \quad \textbf{(2)}$$

Où N représente le nombre de services composants.

Formalisation du problème

Notre objectif est de trouver une composition $SC = \{S_1, S_2, \dots, S_n\}$ tel que :

$MG(SC)$ est maximisée.

4. L'Algorithme Génétique

4.1 Les paramètres de l'AG

La solution quasi-optimale (représenté par un chromosome) est déterminé par l'évolution d'une population initiale à travers un nombre déterminé de générations en appliquant des opérateurs génétiques jusqu'à ce qu'un critère d'arrêt soit satisfait .Dans ce qui suit nous détaillons tous ces paramètres.

Le codage

Le chromosome (service composite) est définit par une liste de tableaux .Le nombre de tableaux est égal au nombre de services web composants et chaque tableau (gène) représente un service web qui est décrit par son identificateur, son concept d'entrée, son concept de sortie. La Figure 10 montre le codage d'un SC avec 3 services.

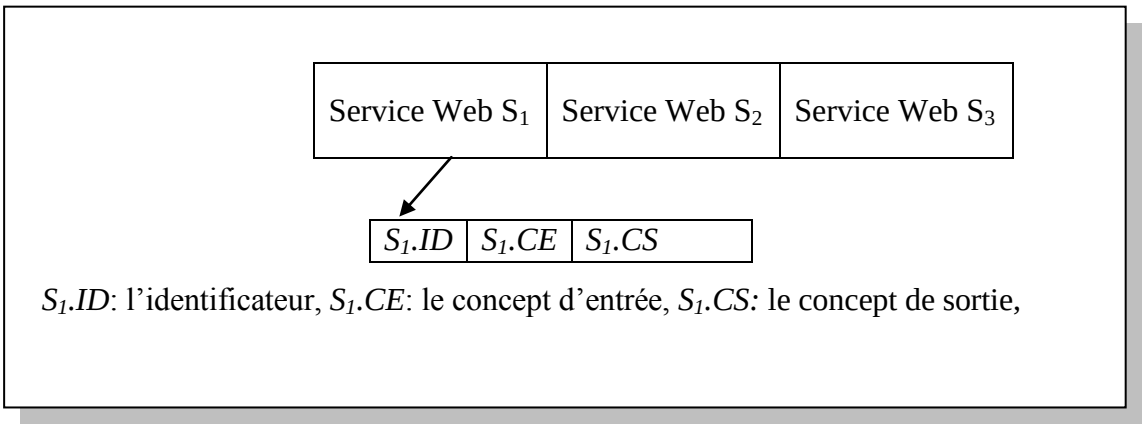

Figure 10: Codage d'un service composite

Population initiale

Elle est constituée d'un ensemble de compositions (caractérisées par ses chromosomes) ou les services web composants sont sélectionnés aléatoirement.

Fonction Fitness (fonction objective)

La fonction fitness joue un rôle prépondérant dans le déroulement de l'AG, elle sert à maximiser la similarité sémantique. Elle est définit par l'équation(3).

$$\boldsymbol{F(SC) = MG(SC)} \qquad (\mathbf{3})$$

Les opérateurs génétiques

La mutation

Nous avons spécifié un opérateur de mutation spécialisée inspirée de celle décrite par [Weise et al., 2007].Cet opérateur soit il supprime un service web d'un service composite SC (mutate1) ou il remplace un service web dans SC par un autre service web tel que $MG(SC)$ est non nul en assurant ainsi une similarité minimale avec la requête de l'utilisateur (mutate2).Contrairement à [Weise et al., 2007], les services web supprimés et remplacés sont choisis aléatoirement. Nous utilisons une variable d'ajustement pour appliquer mutate1 ou mutate2 suivant un seuilσ.

$$\mathbf{mutate_1(SC)} \equiv \begin{cases} \mathbf{SC'} = \{\mathbf{s_1}, \mathbf{s_2}, \dots, \mathbf{s_{|SC|}}\} - \mathbf{s_{1\leq i\leq |SC|}} & \mathbf{if |SC| > 1} \\ \mathbf{SC} & \mathbf{Autrement} \end{cases} \quad (4)$$

$$\mathbf{mutate_2(SC)} \equiv \mathbf{SC'} + \mathbf{s_{1\leq i\leq |SC|}} \mid \mathbf{MG(SC'} + \mathbf{s_{1\leq i\leq |SC|}) > 0} \quad (5)$$

$$\boldsymbol{mutate\ (CS)} \equiv \begin{cases} \boldsymbol{mutate_1(CS)} & \boldsymbol{if\ \ random() > \sigma} \\ \boldsymbol{mutate_2(CS)} & \boldsymbol{Autrement} \end{cases} \quad (6)$$

La sélection

Cet opérateur permet d'identifier statistiquement les meilleurs individus (solutions candidates) de la population courante qui seront autorisés à se reproduire, en fonction de leur valeur de fitness. Dans notre AG on a appliqué une sélection par tournoi binaire.

Critère d'arrêt

L'AG s'exécute tant que le nombre de génération (nbr_gen) est inférieur à nbr_max ou jusqu'à la stagnation de la population, formellement :

Pseudo code de l'AG

L'algorithme de la Figure 11 décrit notre algorithme génétique.

Algorithme génétique

Début

Créer la population initiale

Evaluer chaque individu par la fonction fitness définit par l'équation (3)

Tantque (nbr_gen<nbr_max) **faire**

Sélectionner des individus pour la reproduction.

Appliquer la mutation (mutate 1 et muate 2) selon le seuil σ.

Evaluer de nouveaux les individus selon l'équation (3).

Fin Tant que

Fin

Figure 11 : Notre algorithme génétique

5. Présentation des outils technologiques utilisés

Le prototype a été développé sur un Intel Core2duo, avec une vitesse de 1.6 GHZ, doté d'une capacité mémoire de 1GB de RAM sous Windows XP en utilisant des outils Open source graphiques et développés en JAVA. Nous détaillons, dans ce qui suit, chacun des outils et langages utilisés pour la manipulation des données ainsi que l'implémentation de l'interface utilisateur.

- **Le langage JAVA**

Pour le langage de programmation notre choix s'est porté sur le langage JAVA, et cela parce que JAVA est un langage orienté objet simple ce qui réduit les risques d'incohérence; il est portable, il peut être utilisé sous Windows, sous Linux, sous Macintosh et sur d'autres

plateformes sans aucune modification , enfin il possède une riche bibliothèque de classes comprenant des fonctions diverses telles que les fonctions standards, le système de gestion de fichiers , les fonctions multimédia et beaucoup d'autres fonctionnalités;

- **Eclipse Indigo**

Pour le choix de l'environnement de développement, on a opté pour Eclipse car il possède de nombreux points forts qui sont à l'origine de son énorme succès dont les principaux sont :

- Une plateforme ouverte pour le développement d'applications et extensible grâce à un mécanisme de plugins;
- Support de plusieurs plates-formes d'exécution : Windows, Linux, Mac OS;
- Malgré son écriture en Java, Eclipse est très rapide à l'exécution grâce à l'utilisation de la bibliothèque SWT;

- **L'API JDOM**

JDOM est une API open source Java, vue comme un modèle de documents objets dont le but est de représenter et manipuler un document XML de manière intuitive pour un développeur Java sans requérir une connaissance pointue de XML. JDOM propose aussi une intégration de SAX, DOM, XSLT et XPath.

Un document XML est encapsulé dans un objet de type Document. Les éléments d'un document sont encapsulés dans des classes dédiées : Element, Attribute, Text, Processing Instruction, Namespace, Comment, DocType, EntityRef, CDATA. JDOM permet aussi de vérifier que les données contenues dans les éléments respectent la norme XML.

JDOM propose des réponses à certaines faiblesses de SAX et DOM. La simplicité d'utilisation de JDOM lui permet d'être une API dont l'utilisation est assez répandue.

- **L'API Pellet**

Cette API permet le raisonnement sur les ontologies formalisées avec OWL, elle offre des mécanismes d'inférences basés sur les logiques de description.

6. Présentation de la base des services web et l'ontologie OWL

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<base>
<service num="serv0">
<input>con2</input>
<output>con3</output>
</service>
<service  num="serv1">
<input>con4</input>
<output>con3</output>
</service>
<service num="serv2">
<input>con0</input>
<output>con4</output>
</service>
<service  num="serv3">
<input>con4</input>
<output>con5</output>
</service>
<service  num="serv4">
<input>con1</input>
<output>con2</output>
</service>
<service num="serv5">
<input>con3</input>
<output>con1</output>
</service>
<service num="serv6">
<input>con2</input>
<output>con3</output>
```

Figure 12 : Le modèle XML des services web

- Les balises <*input*> …… </*input*>et <*output*> …… </*output*> contiennent respectivement les concepts d'entrée et de sortie du service identifié par l'identificateur contenu dans l'attribut *num.*

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#" xmlns:rdfs="http:
  <owl:Ontology rdf:about="" />
<owl:Class rdf:ID="con0"/>

<owl:Class rdf:ID="con1">
    <rdfs:subClassOf rdf:resource="#con0" />
  </owl:Class>
<owl:Class rdf:ID="con2">
    <rdfs:subClassOf rdf:resource="#con0" />
  </owl:Class>
<owl:Class rdf:ID="con5">
    <rdfs:subClassOf rdf:resource="#con1" />
  </owl:Class>
<owl:Class rdf:ID="con3">
    <rdfs:subClassOf rdf:resource="#con1" />
  </owl:Class>
<owl:Class rdf:ID="con3">
    <rdfs:subClassOf rdf:resource="#con2" />
  </owl:Class>
<owl:Class rdf:ID="con4">
    <rdfs:subClassOf rdf:resource="#con2" />
  </owl:Class>
<owl:Class rdf:ID="con5">
    <rdfs:subClassOf rdf:resource="#con1" />
  </owl:Class>
<owl:Class rdf:ID="con5">
    <rdfs:subClassOf rdf:resource="#con4" />
  </owl:Class>
 <owl:Thing rdf:ID="inst5">
    <rdf:type rdf:resource="#con0" />
  </owl:Thing>
```

Figure 13 : Le modèle XML de l'ontologie OWL

6.1 Description du prototype

Afin d'illustrer les fonctionnalités de notre prototype « Genetic Composer », nous avons effectué quelques prises d'écrans montrant les différentes étapes nécessaires à la réalisation d'une composition automatique des services web, qui sont :

Chargement de la base des services web et l'ontologie de domaine

La première étape consiste à charger la base des services web ainsi que l'ontologie de domaine (Figure 14).

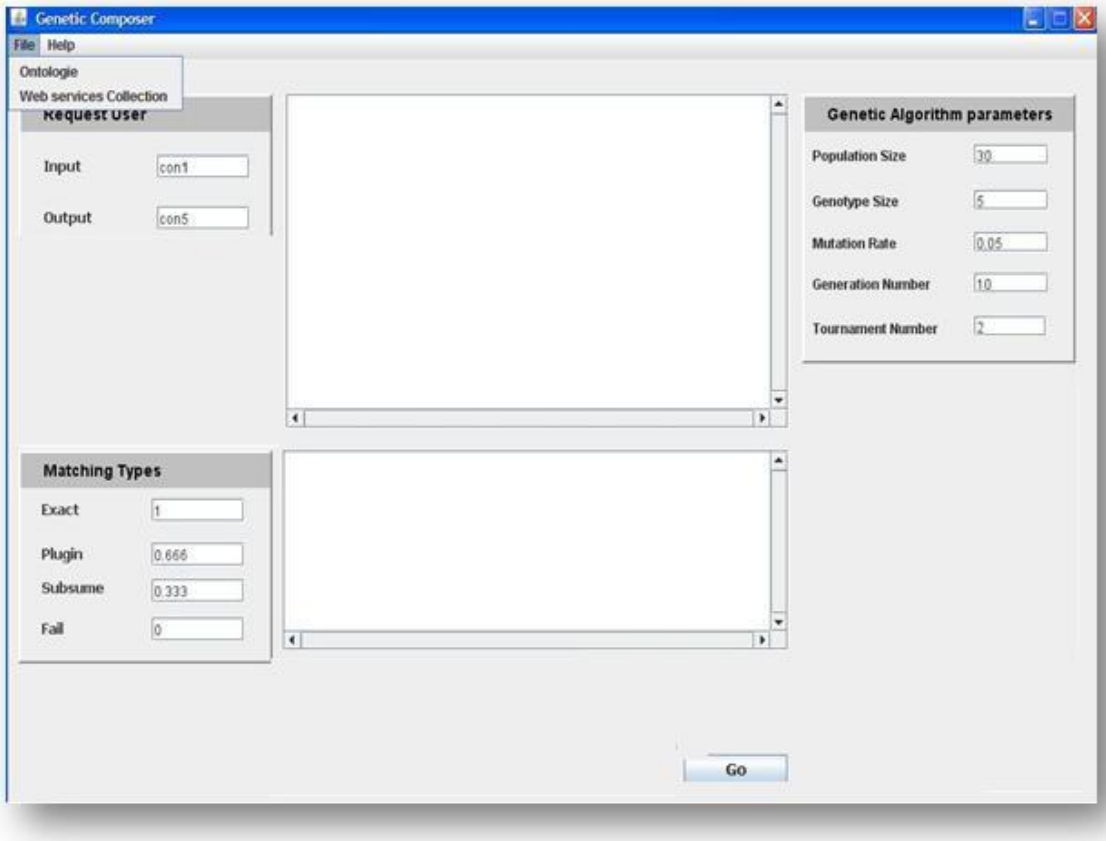

Figure 14 : Interface de prototype

Construction de la requête

Une fois que la base des services et l'ontologie OWL sont spécifiées et chargées, l'utilisateur pourra formuler sa requête. Il introduit les concepts d'E/S

6.2 Expérimentations

Dans cette section nous décrivons les expériences menées pour analyser les performances de notre approche.

Expérimentation 1 :

Dans cette 1ère expérimentation nous voulons enregistrer les meilleurs scores de la fonction fitness en fonction des itérations. Elle est conduite avec les paramètres suivants :

- La taille de la population : 50
- Le seuil de la mutation : 0.05
- La taille de chromosome (taille de la composition) : 5
- Le nombre maximal de générations : 50

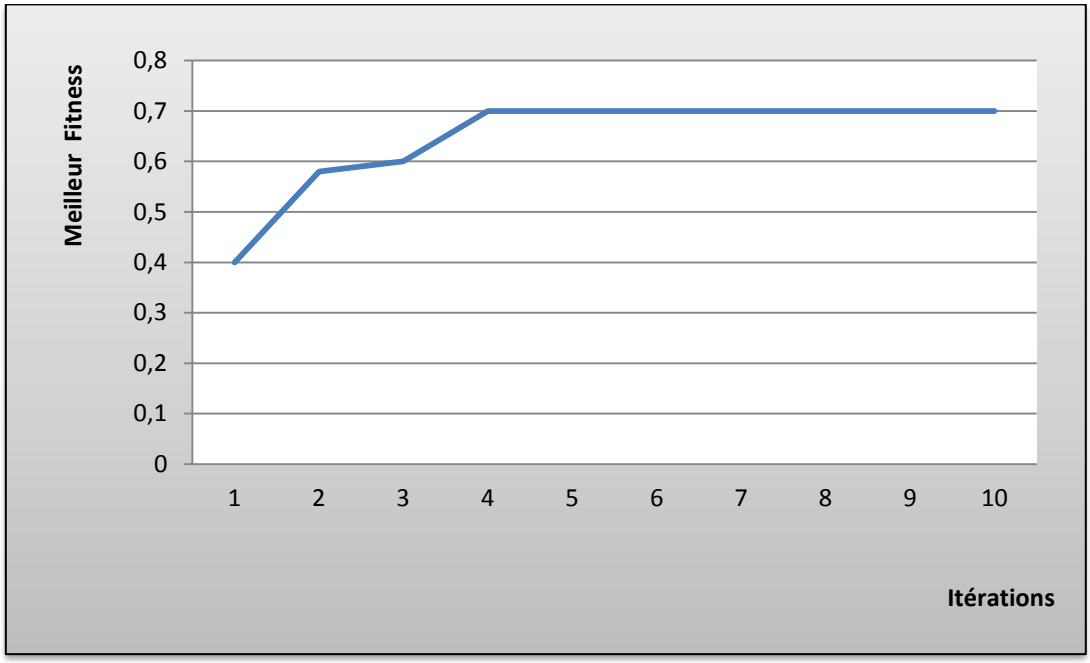

Figure 15 :L'évolution de la fonction fitness

La Figure 15 illustre les résultats de cette expérimentation. Nous remarquons une grande amélioration des performances entre l'itération 1 et l'itération 4. Ceci est du à l'opération mutate2. Après l'itération 5, nous constatons une stagnation des résultats, ceci est expliqué par le fait que l'amélioration de l'aspect fonctionnel (sémantique) des composants du chromosome n'est plus possible, de ce fait les meilleurs chromosomes des anciennes itérations persistent.

Expérimentation 2:

Dans cette deuxième expérimentation nous voulons étudier l'impact de la taille de chromosome sur le temps d'exécution de l'AG. Les autres paramètres de l'AG sont les mêmes que l'expérience 1.

La Figure 16 montre les résultats de cette expérimentation. On remarque que si on augmente la taille de la composition alors le temps d'exécution sera trop élevé, cela peut être expliqué par la faible scalabilité de l'AG.

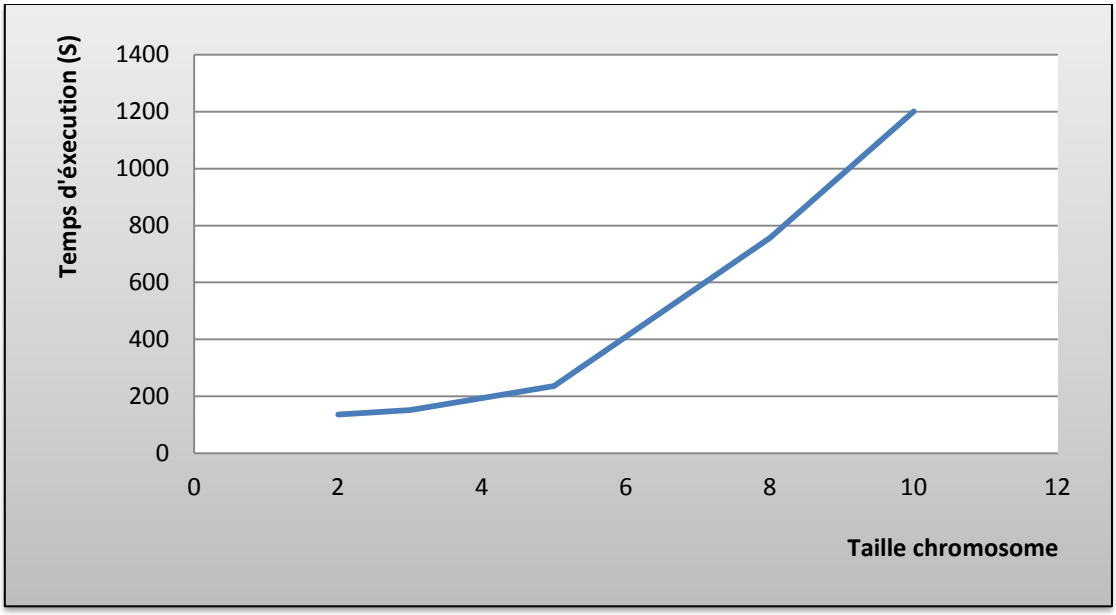

Figure 16 :L'impact de la taille de chromosome sur le temps d'éxécution

7. Conclusion

Au cours de cette dernière étape de notre travail, nous avons réalisé un prototype de système de composition de services Web et nous avons présenté les outils utilisés pour son implémentation. L'AG proposé peut atteindre des solutions proches de l'optimum (une fitness au voisinage de 0.7 pour 3 sous fonctions objectives).Il possède aussi un temps d'exécution acceptable pour les compositions ayant une petite taille (≤ 6) mais pour les compositions de grande taille, l'AG présente un temps de réponse très élevé.

Conclusion Générale

et

Perspectives

Conclusion Générale et Perspectives

Dans ce travail de master, nous avons proposé une approche de composition basée sur un algorithme génétique (les méthodes basés sur les méta-heuristiques) en utilisant les descriptions sémantiques (concepts d'E/S) des services web .En effet, nous avons proposé un mécanisme de matching sémantique basé sur les I/O. Pour explorer l'espace de recherche, l'AG proposé adopte deux fonctions de mutation, la première « mutate 1 » permet de retirer aléatoirement un service web, la deuxième fonction « mutate 2 » permet de remplacer un service web par un autre qui possède une compatibilité sémantique non nulle avec les voisinages.

Les résultats obtenus sont très encourageants et montrent l'efficacité (en terme d'optimalité) de notre approche.

La réalisation de notre approche de composition nous a permis de dégager plusieurs perspectives de travail :

- Considérer que les services web ainsi que la requête sont caractérisés par un ensemble de concepts d'E/S.
- Enrichir les descriptions sémantiques par d'autres aspects fonctionnels telles que les pré-conditions et post-conditions et rendre ainsi plus intelligent le schéma de matching.
- Prendre en compte la pluralité des ontologies de domaine durant le processus de matching, ainsi nous pourrons effectuer un matching entre deux concepts appartenant à des ontologies différentes.
- Combiner l'algorithme génétique avec d'autres techniques d'optimisation comme les algorithmes de colonies de fourmis, les algorithmes d'optimisation par essaims particulaires dans le but d'accroitre son efficacité (temps et optimalité).
- Tester notre approche sur une large base de services web décrits par OWL-S et la comparer avec d'autres approches de composition.

Références bibliographiques

[BAR03] Douglas K. Barry. "Web Services and Service-Oriented Architectures : The Savvy Manager's Guide". Morgan Kaufmann, 2003.

[BCR02] T. Bellwood, L. Clément, and C. von Riegen. "Universal description, discovery and Integration. Technical report, OASIS UDDI Specification Technical Committee", mar 2002. http ://www.oasis-open.org/cover/uddi.html.

[BHI05] Sami Bhiri; Thèse Doctorat "Approche Transactionnelle pour Assurer des Compositions Fiables de Services Web " université Henri Poincaré- Nancy 1. Octobre 2005.

[BOO03] D. Booth, H. Haas, F. McCabe, and E. Newcomer. "Web services architecture ", aout2003. http ://www.w3.org/TR/2003/WD-ws-arch-20030808/.

[Boukhadra 2011] Adel Boukhadra, La composition dynamique des services Web sémantiques a base d'alignement des ontologies owl-s, 2011.

[Bru¼n 2007] Jos de Bru¼n, John Domingue, Dieter Fensel, Holger Lausen, Axel Polleres, Dumitru Roman, et Michael Stollberg, *Enabling Semantic Web Services : The Web Service Modeling Ontology*, édition Springer, 2007.

[Bru¼n 2008] Jos de Bru¼n, Dieter Fensel, Mick Kerrigan, Uwe Keller, Holger Lausen, et James Scicluna, *Modeling Semantic Web Services*, *The Web Service Modeling Language*, édition Springer-Verlag Berlin Heidelberg, 2008.

[Burstein 2004] M. Burstein, J. Hobbs, Ora Lassila, D. Martin, D. McDermott, S. McIlraith, S.Narayanan, M. Paolucci B. Parsia, T. Payne, E. Sirin, N. Srinivasan, and K. Sycara, *OWL-S : Semantic markup for Web services*, pages 63-100, 2004.

[Cardoso 2007] Jorge Cardoso, *Semantic Web Services : Theory, Tools, and Applications*, University of Madeira, Portugal, Information Science reference, édition Dunod, 2007.

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

[Cerami 2002] Ethan Cerami, *Web Services Essentials*, édition O'Reilly, Février 2002.

[CHA02] Extrait de l'ouvrage "Services Web avec SOAP, WSDL, UDDI, ebXML... " de Jean- Marie Chauvet, © Eyrolles, 2002.

[Chappell 2002] David Chappell, et Tyler JEWELL, *Java Web Service*, édition O'Reilly, Mars 2002.

[Chatterjee 2003] Sandeep Chatterjee, et James Webber, *Developing Enterprise Web Services*, édition Prentice Hall PTR, le 14 November 2003.

[CNW01] F. Curbera, A. Nagy, et S. Weerawarana. "Web-services : Why and how". DansWorkshop on Object-OrientedWeb Services (in OOPSLA), Aout 2001.

[Daconta 2003] Michael Daconta, Leo Obrst, et Kevin Smith, *Developing the Semantic Web : a Guide to the Future of XML, Web Services, and Knowledge Management*, edition Wiley, 2003.

[Dallons 2004] Gautier Dallons, *DAML-S : interactions, critique et évaluation*, Institut d'informatique des FUNDP, Namur, Belgique, 2004.

[DAN03] Jérôme Daniel. Extrait de livre "SERVICES WEB Concepts, techniques et outils" Edition vuibert Informatique 2003.

[END et al 04] Mark Endrei, Jenny Ang, Ali Arsanjani, Sook Chua, Philippe Comte, Pål Krogdahl, Min Luo, Tony Newling - 2004 – "Pattern: Service Oriented Architecture and Web Services", IBM INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION , ISBN 073845317X

[Falquet 2001] G. Falquet, et C.L. Mottaz-Jiang, *Navigation hypertexte dans une ontologie multi-pointsde vue*, Nîmes TIC, 2001.

[Fan et al., 2005] J.Fan,B.Ren and L.R.Xiong .An Approach to web service discovery based on the semantics.In FSKD(2),pages 1103-1106,2005.

[Gardien 2002] Georges Gardien, *XML des bases de données aux Services Web*, édition Dunod, 2002.

[Gesnu 2003] Xavier Gesnu, *Développer des Services Web XML et des composants serveurs*, édition Dunod, 2003.

[GHM00] M. Gudgin, M. Hadley, N. Mendelsohn, J. Moreau, and H. Nielsen. "Simple object access protocol (soap) 1.1". Technical report, World Wide Web Consortium, may 2000. http://www.w3.org/TR/SOAP/.

Chouchani, Utilisation d'un algorithme génétique pour la composition de service web, UNIVERSITÉ DU QUÉBEC À MONTRÉAL, mai 2010.
http://www.research.ibm.com/people/b/bth/OOWS2001.htm

[Grimm 2007] Stephan Grimm, et Rudi Studer, *Semantic Web Services, Concepts, Technologies, and Applications*, edition Springer-Verlag Berlin Heidelberg, 2007.

[HEA01] K. Heather. "Web services conceptual architecture (wsca 1.0) ", may 2001.http ://www-306.ibm.com/software/solutions/webservices/pdf/WSCA.pdf.

[HUM04] Humberto, Cervantes - 2004 - "Vers un modèle à composants orienté services pour supporter la disponibilité dynamique", Université Joseph Fourier - Grenoble 1.

[Jaeger et al., 2005]M.C.Jaeger,G.Rojec-Goldmann,G.Muhl,C.Liebetruth,and K.Geihs.Ranked Matching for Service Descriptions using OWL-S.pages 91-102,2005.In Paul Muller, Reinhard Gotzhein,and Jens B,editors,Kommunikation in verteilten Systemen (KiVS 2005),Kaiserslautern,Gemany,February 2005.Springer.

[Kadima 2003] Hubert Kadima, et Valérie Monfort, *Les Web services techniques, démarches et outils*, édition Dunod, 2003.

[KEL03] P. Kellert and F. Toumani. "Les web services sémantiques". In Web sémantique, Action spéci_que 32 CNRS/STIC, October 2003.

[Keller et al., 2005]U.Keller,R.Lara,H.Lausen,A.Pollers,and D.Fensel.Automatic location of services.In Proc.of the 2nd European Semantic Web conference (ESWC),pages 1-16,Heraklion,Crete,2005.LNCS 3532,Springer.

[Küster et al., 2008] U.Küster and B.König-Ries.Evaluating semantic web service matchmaking effectiveness based on graded relevance. 7th International Semantic Web Conference (ISWC08),Karlsruhe,Germany,October 2008.

[Lopes 2005] Denivaldo cicero pavão Lopes, *Etude et applications de l'approche MDA pour des platesformes de Services Web*, Thèse de Doctorat, UFR Sciences et Techniques, Université de Nantes, le 11 Juillet 2005.

[Newcomere 2004] Eric Newcomere, *Understanding Web Services Xml WSDL SOAP And UDDI*, édition O'Reilly, 2004.

[Paolucci et al., 2002]Massimo Paolucci, Takahiro Kawamura, Terry R. Payne, and Katia Sycara.Semantic matching of web services capabilities.2002.

[Paolucci et al., 2004] N.Srinivasan,M.Paolucci,and K.Svcara.Adding OWL-S to UDDI,implementation and throughput.In First International Workshop on semantic web services and webprocess composition (SWSWPC 2004),pages 6-9,2004.

[PFI96] Cuno Pfister and Clemens Szyperski. "Why objects are not enough ". In Proceedings, International Component Users Conference, Munich, Germany, 1996. SIGS.

[RIC04] Ricardo DE LA ROSA-ROSERO. ProjetMaster Mathématiques Informatique "Découverte et Sélection de Services Web pour une application Mélusine".Septembre 2004.

[Schumacher et al., 2008] M.Schumacher,H.Helin,and H.Schuldt.Semantic Web Service Coordination.Chapter 4,CASCOM: Intelligent service Coordination in the Semantic Web,Birkhauser Basel,2008.

[SOA00] SOAP, " Simple Object Access Protocol (SOAP) 1.1", rapport, may 2000, World Wide Web Consortium, http ://www.w3.org/TR/SOAP/.

[Stollberg et al., 2007]M.Stollberg,U.Keller,H.Lausen,and S.Heymans.Two-Phase Web Service Discovery Based on Rich Functional Descriptions.In ESWC'07:Proc.of the 4th European conference on the Semantic Web,pages 99-113,Berlin,Heidelberg,2007.Springer-Verlag.

[UDD02] UDDI, "Universal Description, Discovery and Integration", rapport, mar 2002, OASIS UDDI Specification Technical Committee, http ://www.oasisopen.org/cover/uddi.html.

[Weise et al., 2007] Thomas Weise, Steffen Bleul, and Kurt Geihs.Web Service Composition Systems for the web Service Challenge – A Detailed Review. University of Kassel, Wilhelmsh¨oher Allee 73, 34121 Kassel, Germany, November 19, 2007.

[WSD01] WSDL, "Web Services Description Language (WSDL) 1.1", rapport, mar 2001, World Wide Web Consortium, http://www.w3.org/TR/wsdl.

Sites web

[SW4] http://www.w3.org/2002/ws/