Ministère de l'Enseignement Supérieur
et de la Recherche Scientifique
Université Abou Bekr Belkaid
Faculté de Science
Département Informatique

MÉMOIRE DE FIN D'ÉTUDE

pour obtenir le grade de

Master en Informatique

Spécialité : **MID**

présenté et soutenu publiquement

par

Melle CHEMIDI Zoulikha

le 2 Juillet 2012

Titre:

Sélection de sérvices Web à base de colonies de fourmis

Jury

Président du jury.	Mr.BENAZZOUZ,	MAA UABB Tlemcen
Examinateur.	Mr.MERZOUG,	MAA UABB Tlemcen
Examinateur.	Mr.MOUAFEK,	MAA UABB Tlemcen
Encadreur.	Mr.HADJILA ,	MAA UABB Tlemcen

Ministère de l'Enseignement Supérieur
et de la Recherche Scientifique
Université Abou Bekr Belkaid
Faculté de Science
Département Informatique

MÉMOIRE DE FIN D'ÉTUDE

pour obtenir le grade de

Master en Informatique

Spécialité : **MID**

présenté et soutenu publiquement

par

Melle CHEMIDI Zoulikha

le 2 Juillet 2012

Titre:

Sélection de sérvices Web à base de colonies de fourmis

Jury

Président du jury.	Mr.BENAZZOUZ,	MAA UABB Tlemcen
Examinateur.	Mr.MERZOUG,	MAA UABB Tlemcen
Examinateur.	Mr.MOUAFEK,	MAA UABB Tlemcen
Encadreur.	Mr.HADJILA ,	MAA UABB Tlemcen

Remerciements

Louange à dieu, qui m'a donné la force, le courage et l'espoir nécessaire pour accomplir ce travail et surmonter les difficultés survenues.

Je tien d'abord à exprimer mon profond remercîment) « Mr HADJILA Fethallah » de m'avoir proposé le sujet et de l'attention qu'il a porté sur mon travail, de la confiance qu'il ma accordé dé le début de ce projet, ses conseils son aide, sa présence, et son suivi durant mon projet ; grâce a son intérêt et sa compétence scientifique il a largement contribué à la diversité des travaux réalisé et de l'amélioration de manuscrit

Je désir vivement remercier Mr. M.BENAZZOUZ,Mr M.MERZOUG,et Mr B.MOUFOK d'avoir accepté d'examiner le travail avec beaucoup d'attention.

Je désire remercier le site du zero et mes amis Ghouti et Nassma qui m'ont aider a réaliser mon mémoire avec LATEX.

Au terme de ce travail, je tiens remercier tous ceux qui ont contribué d'une manière ou de l'autre à l'aboutissement de ce travail.

Dedicace

A mes parents ABDERRAHIM ET ASSIA

A mes frères WALID et AYMEN

Vous vous êtes dépensée pour moi sans compter.

En reconnaissance de tous les sacrifices consentis par tous et chacun pour me permettre d'atteindre cette étape de ma vie.

Avec toute ma tendresse.

A mes grands parents, mes oncles, tantes, cousins et cousines.

Vous avez de prés ou de loin contribué à ma formation.

A tous mes amis que je les portent dans mon cœur :

Nawel,Amel,Téma,Hanane,Zineb,Soumia,Naziha,Ghouti.

Et enfin a toute ma promotion de MID.

Table des matières

Table des figures

Liste des tableaux

Glossaire

SW :service web.

SOA : Service Oriented Architecture.

SOAP : Simple Object Access Protocol.

QOS : Qualité de service.

WSDL :Web Services Description Language.

UDDI :Universal Description, Discovery and Integration.

OASIS :Organization for the Advancement of Structured Information Standards.

XML :eXtensible Markup Language.

W3C :World Wild Web Consortium.

B2B :Business to Business.

B2C :Business to Consumer.

SMTP :Simple Mail Transfer Protocol.

FTP :File Transfer Protocol.

RPC :Remote procedure call, un protocole d'appel de procédures à distance.

firewall :par feu.

TCP/IP :Transmission Control Protocol / Internet Protocol.

HTTP :Hyper Text Transfert Protocol.

HTML :Hyper Text Markup Language.

REST :Representational State Transfer.

ACO :Algorithme de Colonies de Fourmis.

Introduction

1 contexte

Les organisations sont de plus en plus nombreuses à se tourner vers des architectures à base des services Web pour le développement et l'intégration d'applications ou de systèmes d'information. L'importance des standards dans ce contexte a sans doute accentué le phénomène. Le service web traite les données et relie à l'internet et à l'externe d'une application logiciel. comme il devient important d'être en mesure d'un système distribué référentiel d'information dynamique. Il a maintenant évolué pour englober diverses ressources d'information dans le monde entier accessibles.

2 Problématique

En quelques années, internet a touché un public de plus en plus nombreux et satisfait des besoins de plus en plus variés.Cependant, Pour certains types d'application, il est nécessaire de combiner un ensemble des services Web en services plus complexes (services Web agrégés ou composites) afin de répondre à des exigences plus complexes, Pour cela on cherche à sélectionner un groupe de services web(S.W) qui répondent à une requête complexe prédéfinie sachant que chaque S.W est caractérisé par cinq critères de QoS qui sont définit comme suit :

- La Latence :qui est le temps pour transmettre et recevoir les résultats,
- La Fiabilité :définit le nombre total d'appels,

- La Disponibilité : qui est le temps où l'opération accessible pendant le temps de mesure total d'opération.
- Le Coût :Définit par un montant pour exécuter l'opération pour cela le prix d'un utilisateur est prêt à engager pour obtenir un niveau de service
- La Réputation : la réputation d'un service défint en fonction de trois facteurs : notes prises par les utilisateurs, le respect de la qualité de service, et sa vérité, à savoir les changements de conformité de qualité de service au fil de temps.

3 Contribution

On propose dans Ce travail une approche basée sur les colonies de fourmis « ACO ». L'ACO forme une classe des métaheuristiques récemment proposée pour des problèmes d'optimisation difficile (problèmes NP-difficiles et les problèmes dynamiques).

4 Plan du mémoire

Le mémoire est organisé selon le plan du travail suivant :

Chapitre 1 : « Les Services Web »
Ce chapitre introduit de façon générale le domaine de service web dont il y a la représentation de ces technologies, comme on site ces caractéristiques ainsi les avantages et les inconvénients.

Chapitre 2 : « Méta-heuristique et la Sélection des Services Web »
Ce chapitre repose sur le problème de Méta-heuristique et de la sélection des services web et aussi ces différentes méthodes, ce chapitre explique le fonctionnement de l'approche et les différentes méthodes de sélection

Chapitre 3 : « L'Algorithme de colonies de fourmis»
Ce chapitre expose les concepts fondamentaux des algorithmes d'optimisation dont on consacre l'algorithme de colonie de fourmis. Après une définition de son

fonctionnement, et une terminologie pour définir le vocabulaire employé par notre travail.

Chapitre 4 : « Conception et implémentation »

Ce chapitre illustre notre propre cas d'application dont le cadre de sélection de service web en utilisant l'approche de colonie de fourmis qui est extrêmement important pour obtenir de bons résultats.

Chapitre 1

Les Services Web

1 Introduction

De nombreuses organisations tournent actuellement vers des architectures à base des services Web pour le développement et l'intégration d'applications ou des systèmes d'information. Les services web constituent une technologie idéale pour l'interopérabilité et l'intégration des applications sur internet car ils sont indépendants des, langages, des systèmes d'exploitation, des environnements sous jacents. Dans ce chapitre, on va présenter les technologies des services web, les langages de description des services web. Ce chapitre permet de faire l'état des lieux du domaine.

2 L'architecture SOA

Actuellement, l'architecture orientées services présentent des limitations par rapport à leurs adaptabilités dynamiques aux changements du logique métier. Ces limitations affectent les fournisseurs et les consommateurs de services.

2.1 Définition

L'Architecture Orienté Service (SOA)[3] est une approche structurale qui permet d'intégrer les fonctions qui incluent des applications d'entreprises comme des services interopérables. L'architecture orientée services est le terme utilisé pour dé-

signer un modèle d'architecture pour l'exécution d'applications logicielles réparties. Les deux modèles (CORBA et DCO) relèvent de l'architecture par composants logiciels répartis plutôt que de l'architecture orientée services, et le terme service est généralement absent de leurs terminologies. Ces concepts ont ouvert la voie aux services web. Les systèmes d'information qui découlent de ces deux technologies sont fortement couplés, c'est-à-dire que les interactions entre deux applications doivent être soigneusement décrites et qu'une modification dans un composant conduit généralement à une défaillance du système.

- Un des avantages des SOA réside dans le fait que les applications réparties n'ont plus besoin d'un système de middleware réparti commun pour communiquer, mais seulement des protocoles et des technologies de communication interopérables sur Internet.
- une architecture orientée services peut donc être représentée par une interconnexion de multiples points d'accès.
- Le contrat de service du modèle des SOA s'inspire directement du modèle des contrats professionnels de service. Il s'agit d'un document qui développe l'ensemble des points permettant de décrire et donc de définir la relation de service entre les parties contractantes ,et dans le cas des services web, ce contrat est un document WSDL.

2.2 Fonctionnement de SOA

L'idée de SOA est de considérer chaque application non pas comme une entité isolée, mais comme un ensemble des services interdépendants les uns des autres. Ce système permet une bonne modularité puisque le fait de remplacer un service par un autre ne gene pas au fonctionnement des autres services. SOA dispose aussi de bonnes possibilités d'évolution. Il n'est pas nécessaire de faire évoluer tous les programmes séparément puisque la modification des services se répercute sur toutes les applications qui l'utilisent. Bien qu'il n'existe pas de spécification officielle d'une SOA, on retrouve toujours les éléments suivants : En premier lieu, « les services ».

Un service, est une fonction contenue dans un composant que l'on peut interroger à l'aide de requêtes contenant des paramètres et fournissant des réponses. Idéalement chaque service doit être indépendant des autres. Ensuite, « La description des service » qui consiste à décrire les paramètres d'entrée du service et le type des données en réponse. Le principal format de description de services est WSDL (Web Services Description Language), normalisé par le W3C. WSDL et a été proposé en 2001 au W3C pour standardisation, La version 1.1 n'est pas approuvée par le W3C mais la version 2.0 a été approuvée le 27 juin 2007 et est désormais une recommandation officielle du W3C. WSDL sert à détailler le protocole de communication, le format des messages requis pour communiquer avec ce service, les méthodes que le client peut invoquer et la localisation du service. Puis « La publication » (advertising) et « la découverte » (discovery) des services. La publication consiste à publier dans un registre les services disponibles aux utilisateurs tandis que la découverte quand à elle consiste a rechercher un service parmi ceux qui ont été publiés. Le principal standard utilisé est UDDI (Universal Description Discovery and Integration), normalisé par l'OASIS. (Organization for the Advancement of Structured Information Standards) consortium fondé en 1993 et qui compte plus de 3500 membres dans plus de 100 pays parmi lesquels Boeing, Adobe ou le département de la défense des États Unis . Finalement, « L'invocation » fait la connexion et l'interaction du client avec le service. Le principal protocole utilisé pour l'invocation de services est SOAP (anciennement Simple Object Access Protocol). SOAP a été initialement défini par Microsoft et IBM, mais est devenu par la suite une recommandation du W3C. Il permet la transmission des messages entre objets distants, ce qui veut dire qu'il autorise un objet à invoquer des méthodes d'objets

physiquement situés sur un autre serveur. SOAP n'est plus un acronyme depuis la version 1.2. Il existe une mise en œuvre de la SOA qui repose entièrement sur Internet est appelée la WOA (Web Oriented Architecture). Les services sont donc tous présentés sur le web ce qui privilégie les services web. L'avantage de cette conception est que le support des messages d'invocation de service est universel et ne nécessite

pas de traitement. Cependant, cette solution est actuellement assez dépréciée dans les situations où les performances sont une priorité.

3 Définition

Les Services Web[3] sont, en quelque sorte, le prolongement de la programmation objet. Ainsi, un Web Service est donc une sorte d'objet avec une seule fonctionnalité permettant, avec d'autres Web Services, la composition d'une application plus large pouvant avoir plusieurs fonctionnalités. Il est aussi une « unité logique applicative » accessible en utilisant les protocoles standard d'Internet. Les services web émergent au début des années 2000 dans le monde
des systèmes d'information dans le contexte de la mise en œuvre d'Architectures Orientées Services (SOA : Service Oriented Architecture),dont l'objectif est de rendre plus modulaire et moins propriétaire le développement des logiciels des applications informatiques, en implémentant les fonctions applicatives élémentaires sous forme de modules.

Les services web sont des applications qui relient des programmes, des objets, des bases de données ou des processus d'affaires à l'aide de XML et de protocoles internet standard.

4 Cycle de vie d'utilisation

- L'architecture de référence des services Web se base sur les trois concepts suivants :
- le fournisseur de service (service provider) : définit le service publie sa description dans l'annuaire réalise les opérations
- l'annuaire (discovery agency) : reçoit et enregistre les descriptions de services publiées par les fournisseurs et répond aux recherches de services lancées par les clients

- le client (service requestor) : obtient la description du service grâce à l'annuaire utilise le service Le rôle de chacun des éléments précédents :
- Le fournisseur de service créé le service Web, puis publie son interface ainsi que les informations d'accès au service, dans un annuaire de services Web.
- L'annuaire de service rend disponible l'interface du service ainsi que ses informations d'accès, pour n'importe quel demandeur potentiel de service.
- Le consommateur de service accède à l'annuaire de service pour effectuer une recherche afin de trouver les services désirés. Ensuite, il se lie au fournisseur pour invoquer le service.

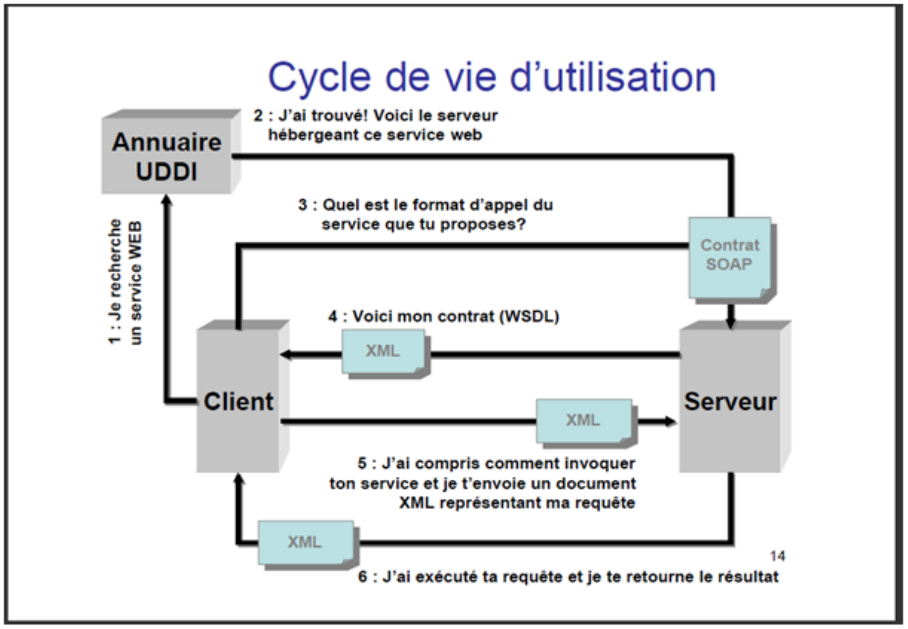

FIGURE 1.1 – Cycle de vie d'utilisation des services web

5 Le fonctionnement d'un service web

5.1 Services Web non publics

- Services web non publiés dans un annuaire UDDI
- Services web dont le point d'accès est connu des utilisateurs du service

- Généralement des SW intranet, des SW de type B2B (Business to Business)

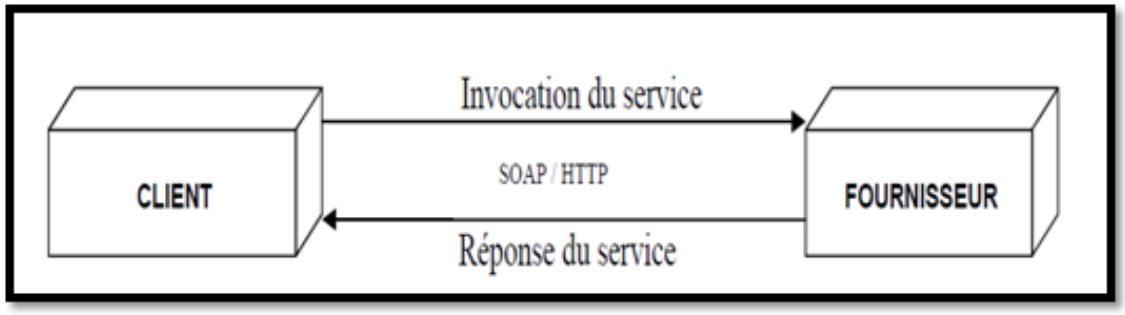

FIGURE 1.2 – Fonctionnement d'un service web à accés non public

5.2 Services Web publics

- Services web publiés dans un annuaire UDDI
- Généralement des SW de type B2C (Business to Consumer), ex : agence de voyage, ...etc.
- Le SW et l'annuaire UDDI qui le publie peuvent ne pas résider sur la même machine

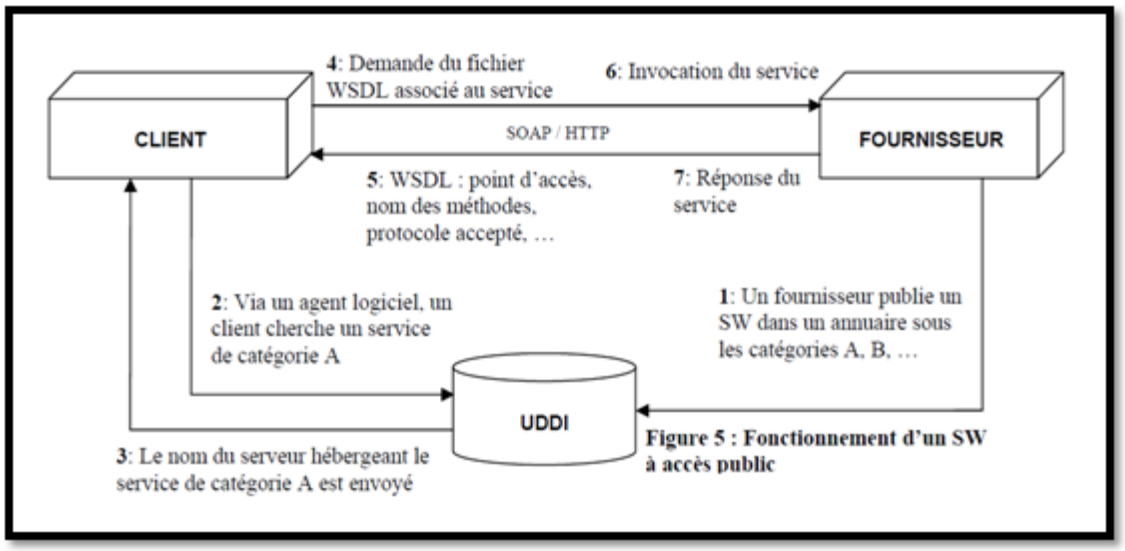

FIGURE 1.3 – Fonctionnement d'un service web à accés public

6 Les caractéristiques d'un service web :

– Un service Web possède les caractéristiques suivantes :

- il est accessible via le réseau ;
- il dispose d'une interface publique (ensemble d'opérations) décrite en XML ;
- ses descriptions (fonctionnalités, comment l'invoquer et où le trouver ?) sont stockées dans un annuaire ;
- il communique en utilisant des messages XML, ces messages sont transportés par des protocoles Internet (généralement HTTP, mais rien n'empêche d'utiliser d'autres protocoles de transfert tels : SMTP, FTP, BEEP...) ;
- l'intégration d'application en implémentant des services Web produit des systèmes faiblement couplés, le demandeur du service ne connaît pas forcément le fournisseur.

– Ce dernier peut disparaître sans perturber l'application cliente qui trouvera un autre fournisseur en cherchant dans l'annuaire.

7 Intérêt des services 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 compréhensives par tout le monde.

– Ils sont indépendamment de

- la plate-forme (UNIX, Windows, ...)
- l'implémentation (VB, C#, Java, ...)
- l'architecture sous-jacente (.NET, J2EE,Axis...)

– Les Web services sont basés sur les protocoles et les langages du Web, en particulier HTTP et XML.

– Le cadre des Web services contient en lui-même toutes les informations nécessaire à l'utilisation des applications, sous la forme de trois fonctions : trouver, décrire et exécuter.

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 des services existants.

8 Les technologies des Services Web

Internet est devenu le moyen de communication,centaines de millions d'utilisateurs qui échangent des informations numériques à travers des serveurs. Ce succès d'Internet s'est construit autour des technologies suivantes :TCP/IP, HTTP,HTML,WSDL, REST, XML-RPC, SOAP et UDDI.

- TCP/IP (Transmission Control Protocol / Internet Protocol) a apporté la connectivité mondiale.
- HTML (Hyper Text Markup Language) est devenu le langage universel de présentation des informations aux utilisateurs.
- REST (Representational State Transfer) est une architecture des services Web. Élaborée en l'an 2000 par Roy Fiedling, l'un des créateurs du protocole HTTP, du serveur Apache HTTPd et d'autres travaux fondamentaux, REST est une manière de construire une application pour les systèmes distribués comme le World Wide Web.
- XML (eXtensible Markup Language) : XML est un métalangage de représentation de données, c'est un ensemble de règles de formatage pour composer des messages valides. XML est un langage de balisage extensible qui a été mis au point par le XMLWorking Group sous l'égide du World Wide Web Consortium (W3C) en 1996 ,il est née de la tentative de mettre SGML sur le Web. La première spécification de XML est apparue en février 1998 et se concentre sur les données,contrairement à HTML (par exemple) qui focalise sur la présentation. XML permet donc de transformer Internet d'un univers d'information et de présentation des sites Web statiques à un univers Web programmable et dynamique, centré sur les données. Comme Il est indépendant

des plates-formes informatiques. Il est lisible par l'humain mais est destiné a être lu par la machine. Il est flexible en ce sens qu'on peut définir d'autres langages à partir d'XML.

- XML-RPC : XML-RPC est un protocole simple utilisant XML pour effectuer des messages RPC. Les requêtes sont écrites en XML et envoyées via HTTP POST. Les requêtes sont intégrées dans le corps de la réponse HTTP. XML-RPC est indépendant de la plate-forme, ce qui lui permet de communiquer avec divers applications. Par exemple, un client Java peut parler de XML-RPC à un PerlServer.
- XML Schema : XML Schema publié comme recommandation par le W3C en mai 2001 est un langage de description de format de document XML permettant de définir la structure d'un document XML. La connaissance de la structure d'un document XML permet notamment de vérifier la validité de ce document.

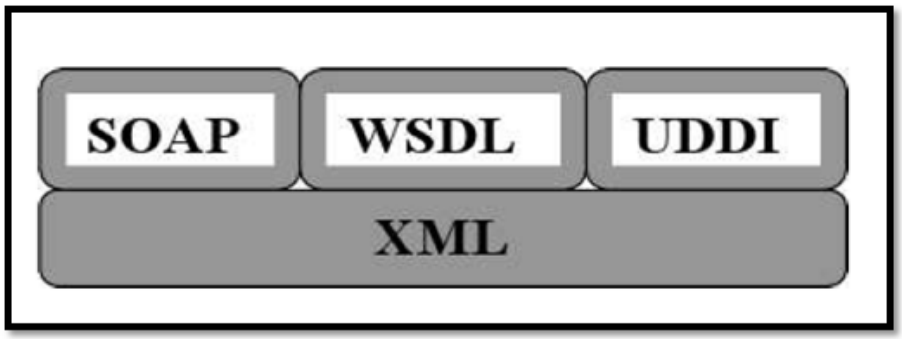

FIGURE 1.4 – Architecture des composants des services web

– Ces technologies qui permettent un accès rapide à l'information ont contribuée à la révolution Internet. Cependant, bien qu'un grand nombre d'informations existent sur Internet, il est souvent difficile de les localiser. De plus, les utilisateurs et les entreprises souhaitent des sites Web qui fournissent des services tels que les commandes des clients. HTML n'étant pas conçu pour développer de tels services, un autre langage était nécessaire :XML est née.

- Fondamentalement, SOAP, WSDL et UDDI sont des technologies issues de

l'intérêt parmi des membres de la communauté Internet à développer un mécanisme pour échanger des documents XML sur le Web entre systèmes d'information.

8.1 SOAP (Simple Object Access Protocol) :

Les applications communiquent grâce aux RPC(Remote Procedure Calls) entre les objets. Mais HTTP n'a pas été conçu pour ca. Or RPC ne résoud pas
les problèmes de sécurité et de compatibilité car certains firewall et proxy bloque ce genre de trafic. L'utilisation de la communication via HTTP reste incontournable car il est supporté par tous les navigateurs et serveurs. Aussi SOAP a été créer pour palier ce manque et permettre la communication entre application qui tournent sur différents OS, différentes technologies, et différents langages
de programmation.

8.1.1 Définition

- SOAP[18] est un protocole de la famille XML servant à l'échange d'informations dans un environnement distribué et décentralisé. Il est considéré comme la technologie la plus importante des services web.
- Le standard SOAP a été proposé au W3C par Microsoft, IBM, Lotus, DevelopMentor et Userland, sa première version a été accepté en 2000 définit trois éléments composant un message : l'enveloppe, l'entête du message et le corps du message.
  - L'enveloppe définit le cadre pour décrire ce qui est dans le message et comment le traiter.
  - Les règles d'encodages sont placées dans l'en-tête et servent à exprimer et définir le mécanisme de représentation des données. Le corps du message permet de transmettre les requêtes et les réponses entre les systèmes.

8.1.2 La structure d'un message soap :

SOAP définit un format pour l'envoi des messages. Les messages SOAP sont structuré en un document XML et comporte deux éléments obligatoires : Une enveloppe et un corps (une entête facultative).

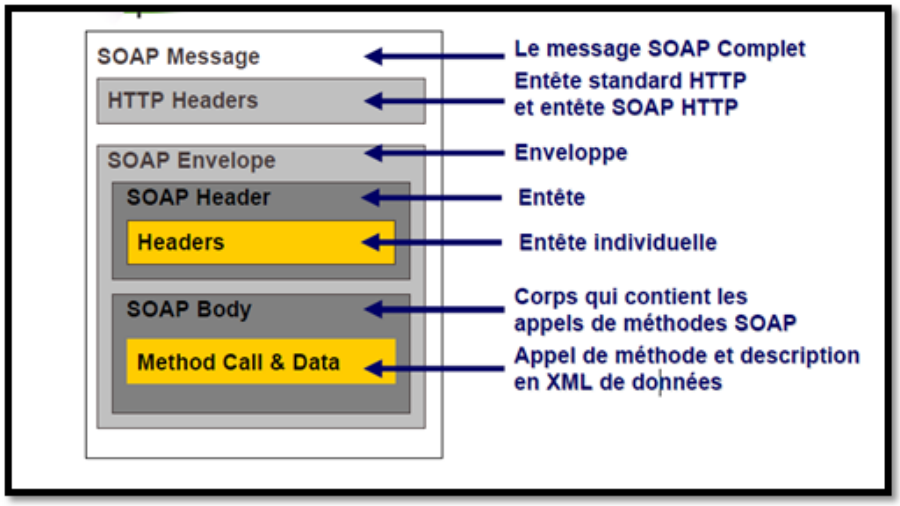

FIGURE 1.5 – La structure d'un message soap

- Le schéma ci-dessus montre la structure d'un message SOAP :
  - Une enveloppe qui définit le contenu du message
  - Une en-tête optionnel qui contient les informations d'en-tête (autorisations et transactions par exemple)
  - Un corps contenant les informations sur l'appel et la réponse
  - Des attachements optionnels.

8.1.3 Exemples d'un message SOAP :

Exemple 1 :

- Soit un dialogue RPC encodé par SOAP qui contient un message de requête et un message de réponse. Considérons la méthode d'un service simple qui double la valeur d'un entier donné.
- Signature de la méthode :

```
Int doubleAnInteger (int numberToDouble) ;
```

Requête

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<SOAP-ENV:Envelope
 SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
 xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
 xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance"
 xmlns:xsd="http://www.w3.org/1999/XMLSchema">
      <SOAP-ENV:Body>
             <ns1:doubleAnInteger
              xmlns:ns1="urn:MySoapServices">
                    <param1 xsi:type="xsd:int">123</param1>
             </ns1:doubleAnInteger>
      </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

FIGURE 1.6 – Requête SOAP

- Dans la requête ci-dessus, on retrouve bien les deux éléments obligatoires caractéristiques d'un message SOAP. Le tag de l'enveloppe dans le message de requête contient également des définitions de namespaces. On trouve ensuite le tag SOAP Body qui encapsule le tag de méthode qui porte le nom de la méthode elle même (ou le même nom suivi de "réponse" dans le cas du message de réponse.

Exemple 2 :

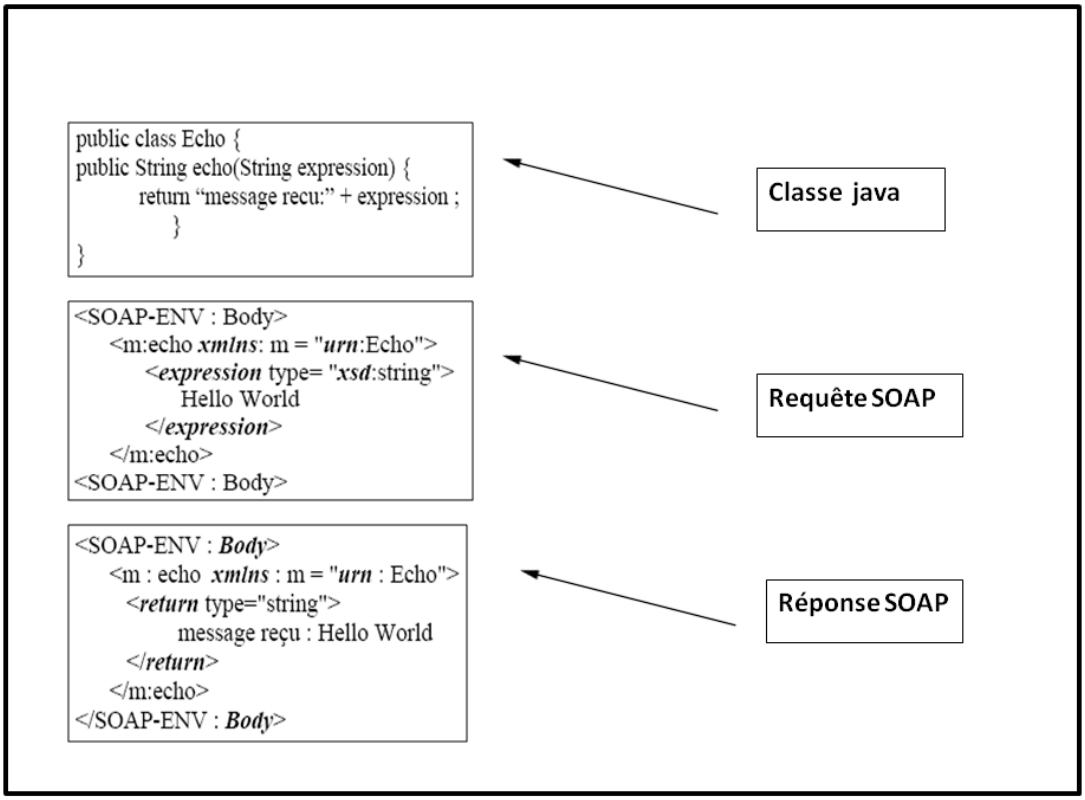

FIGURE 1.7 – Exemple SOAP

8.1.4 Les caractéristiques de SOAP :

- Plus de 80 implantations de la spécification SOAP ont été développé jusqu'à présent. Cela démontre déjà la simplicité, la popularité et la justesse de l'approche qu'il offre pour transporter des données sur le Web.
- SOAP permet une normalisation des échanges de données. Les données sont encodées en XML et échangées par des appels de procédures à distance (RPC) en utilisant HTTP/SMTP/POP comme protocole de communication.
- SOAP est un Message unidirectionnel, il fonctionne de manière synchrone et asynchrone ,et indépendant de la plate-forme et du langage ,et il n'est pas perturbé par les pare-feu.
- SOAP est un protocole de type requête/réponse fonctionnant sur le protocole

de communication HTTP.

- SOAP est un protocole de l'architecture SOA (Service Oriented Architecture) qui assure la messagerie. 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.
- Une des mesures les plus importantes prises par les concepteurs du protocole SOAP est que la spécification du protocole ne donne aucune indication sur le mécanisme de transport du message.
- SOAP fait une séparation entre le message (i.e le document XML) et le moyen de transport utilisé.

8.1.5 Principe de fonctionnement de SOAP :

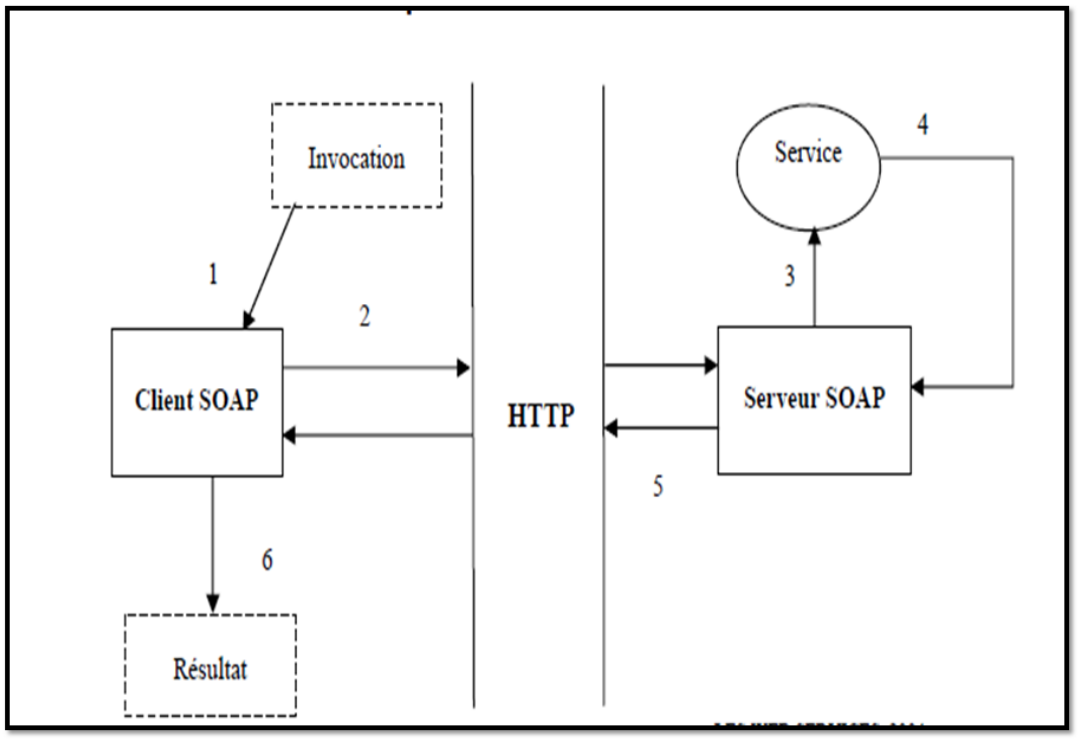

FIGURE 1.8 – Principe de fonctionnement SOAP

- Explication :

1. Un demandeur de service invoque un service par l'entremise d'un agent logiciel,

généralement un navigateur web.

2. Le message est rassemblé par le client SOAP qui le met sous format SOAP avant de l'acheminer via HTTP au serveur SOAP
3. La requête SOAP arrive au niveau du serveur qui effectue une analyse syntaxique sur le document XML afin de vérifier sa cohérence avant de donner la main au service invoqué.
4. À ce niveau, si l'analyse syntaxique du serveur échoue, la demande est rejetée et un message d'erreur est envoyé au client.
5. Si l'analyse syntaxique réussit, le service invoqué traite le message et renvoie le résultat au serveur SOAP qui le met sous format SOAP avant de le retransmettre vers le client SOAP toujours via HTTP.
6. Le client SOAP reçoit la réponse SOAP et accomplit son traitement applicatif.

8.2 WSDL (Web Services Description Language) :

8.2.1 définition :

WSDL[18]C'est un langage permettant de décrire les Services Web et les données attendues par ces Services Web. L'utilité de WSDL est donc de décrire et publier le format et les protocoles d'un Service Web de manière homogène par l'utilisation de XML.Cela permettra au requérant et à l'émetteur d'un service de comprendre les données qui seront échangées. WSDL a été développé conjointement par IBM, Microsoft et Ariba et a été présenté pour l'analyse au W3C(http ://www.w3.org/2002/ws/) qui l'accepté comme une notice. Malgré que WSDL soit neutre du point de vue du protocole, la majorité des implantations WSDL se font à partir de SOAP puisque les promoteurs de SOAP sont aussi les promoteurs de WSDL.

8.2.2 Les six éléments de base d'un document WSDL :

Un document WSDL est formé des six éléments suivants :

- l'élément types : définition des types de données utilisés lors de l'échange ;
- l'élément message : représentation abstraite du contenu d'un message (données à transmettre, valeur de retour) ;
- l'élément portType : définition d'un ensemble d'opérations offerts par un service web ;
- l'élément binding : définition

  d'un protocole de transport et le format des messages ;
- l'élément service : ensemble de ports (lien url) associés à une opération donnée ;
- l'élément port : adresse d'une liaison définissant le point d'accès associé à un service.

– WSDL permet de décrire les Types des données, des messages ,des opérations, et des Binding Ces descriptions doivent être exhaustives et précises, Par exemple pour google nécessite 200 lignes WSDL pour 3 opérations.

8.2.3 Squelette d'un document WSDL :

Voici le squelette d'un document WSDL

- Remarque : Les symboles ? et * présents dans la grammaire ci-dessus ont la même signification que dans les expressions régulières : ? signifie apparaît au plus une fois, * signifie apparaît 0 ou plusieurs fois.

– Ce document constitue le squelette d'un document WSDL. La plupart des éléments de ce document autorisent des extensions pour des besoins spécifiques. Alors il ne sera pas rare de voir dans un vrai document WSDL de nouveaux éléments ou attributs qui n'auraient pas été mentionnés sur ce présent document.

```
1. <wsdl:definitions name="nmtoken"? targetNamespace="uri"?>
2. <import namespace="uri" location="uri"/>*
3. <wsdl:types> ?
<wsdl:documentation .... />?
<xsd:schema .... />*
</wsdl:types>
4. <wsdl:message name="nmtoken"> *
<wsdl:documentation .... />?

<part name="nmtoken" element="qname"? type="qname"?/> *
</wsdl:message>
5. <wsdl:portType name="nmtoken">*
<wsdl:documentation .... />?
<wsdl:operation name="nmtoken">*
<wsdl:documentation .... /> ?
<wsdl:input name="nmtoken"? message="qname" ...>?
<wsdl:documentation .... /> ?
</wsdl:input>
<wsdl:output name="nmtoken"? message="qname" ...>?
<wsdl:documentation .... /> ?
</wsdl:output>
<wsdl:fault name="nmtoken" message="qname"> *
<wsdl:documentation .... /> ?
</wsdl:fault>
</wsdl:operation>
</wsdl:portType>
6. <wsdl:binding name="nmtoken" type="qname">*
<wsdl:documentation .... />?
<wsdl:operation name="nmtoken">*
<wsdl:documentation .... /> ?
<wsdl:input> ?
<wsdl:documentation .... /> ?
</wsdl:input>
<wsdl:output> ?
<wsdl:documentation .... /> ?
</wsdl:output>
<wsdl:fault name="nmtoken"> *
<wsdl:documentation .... /> ?
</wsdl:fault>
</wsdl:operation>
</wsdl:binding>
7. <wsdl:service name="nmtoken"> *
<wsdl:documentation .... />?
<wsdl:port name="nmtoken" binding="qname"> *
<wsdl:documentation .... /> ?
</wsdl:port>
</wsdl:service>
</wsdl:definitions>
```

FIGURE 1.9 – Squelette d'un document WSDL

8.2.4 Description de la structure d'un document WSDL :

La structure d'un document WSDL se compose de deux parties importantes qui sont la partie abstraite et la partie contrainte définies comme suit :

- La partie abstraite : décrit les messages envoyés et les messages reçus, et les associés d'exploitation d'un modèle d'échange de messages avec un ou plusieurs messages.
- La partie contrainte : précise les détails du format de transport et de fil pour une ou plusieurs interfaces , un port (un point de terminaison) associe une adresse réseau avec une liaison et un service qui regroupe les points de terminaison qui implémentent une interface commune .

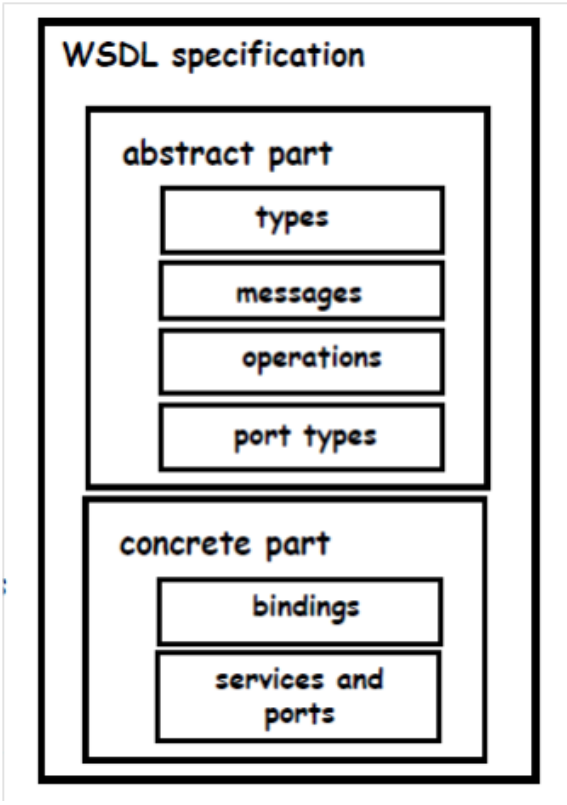

FIGURE 1.10 – la structure d'un document WSDL

- **a.L'élément définition :** L'élément definitions est la racine d'un document WSDL. Cet élément peut optionnellement avoir les attributs name et target-Namespace qui permettent respectivement de lui associer un nom et un nom d'espace. Ces deux attributs permettent à un document WSDL d'être importé

par d'autres documents de définitions (schéma ou document WSDL) au cas où ceux-ci l'utilisent en tant qu'extension à leur propre définition. Cette façon permet la réutilisation de définition de service, et le point numéro 2 du squelette d'un document WSDL montre sa mise en oeuvre. D'autres noms d'espace peuvent être utilisés s'ils sont requis dans la définition du service. Les éléments documentation sont optionnels et peuvent être utilisés sous n'importe quels éléments d'un document WSDL afin de leur associer une explication en langage naturel.

- **b. L'élément types :** L'élément types, grâce à l'importation d'un schéma ou à la définition de structures de données, permet de spécifier le type (l'encodage) des données qui sont utilisées lors de l'échange des messages.
- **c. L'élément message :** L'élément message sert à donner une représentation abstraite du contenu d'un message qui peut être une invocation ou une réponse. L'élément message peut se répéter plusieurs fois et possède un attribut name qui lui sert de référence parmi les autres éléments message du même document WSDL. L'élément message comprend une ou plusieurs sous-éléments part appelés également parties logiques. L'élément part peut être vue comme des paramètres d'entrée et de sortie d'une fonction. L'élément part a trois attributs qui sont name, type et element. Seul l'attribut name est obligatoire et permet d'associer un nom unique à un élément part parmi les autres éléments part du même élément message. L'attribut type permet de définir le type de donnée de l'élément part. L'attribut element est une alternative syntaxique pour spécifier un type complexe pour l'élément part.
- **d. L'élément portType :** L'élément portType permet de définir un ensemble d'opérations (ou fonctions). Il est optionnel et peut se répéter plusieurs fois dans un document WSDL. Son attribut name lui sert d'identifiant unique parmi les autres éléments portType d'un même document WSDL. Le sous-élément de portType est l'élément operation qui a aussi un attribut name qui lui sert d'identifiant non nécessairement unique (surcharge de fonction par

exemple). Les sous-éléments de l'élément operation sont input (un message d'entrée), output (un message de sortie) ou fault (un message d'erreur). La présence ou non de ces trois éléments dépend du type de transmission utilisé dans l'élément binding4. Le sous-élément fault est optionnellement utilisé pour indiquer un rapport d'erreur dans les cas de transmission à deux sens. Les trois sous-éléments de l'élément operation ont un attribut optionnel name et un attribut obligatoire message. L'attribut name sert d'identifiant unique, et l'attribut message permet aux éléments operation de se référer aux éléments message.

- **e. L'élément binding :** L'élément binding définit un protocole de transport et le format des données pour les opérations et les messages pour chaque élément portType. Il est optionnel et peut être présent plusieurs fois. Plusieurs éléments binding peuvent être définis pour un même élément portType. Un élément binding comprend deux attributs obligatoires name et type. L'attribut name lui sert de référence unique parmi d'autres éléments binding du même document WSDL. L'attribut type sert de référence à l'élément portType auquel il est lié. Un élément binding peut contenir plusieurs sous-éléments operation. Ce sous-élément a un attribut name qui permet de faire correspondre à un élément operation défini dans portType. Puisque la valeur d'un attribut name d'un élément operation dans portType peut ne pas être unique, dans ce cas on spécifiera les éléments input, output et fault correspondants à l'élément operation que l'on veut référencer. À noter la présence sous le sous-élément body l'attribut obligatoire use avec la valeur literal. L'autre valeur possible de use est encoded. La valeur literal de use sous un élément indique que lors de la génération du message SOAP, les données textuelles de cet élément ne renfermeront pas l'information d'encodage spécifiée soit dans types ou via l'importation d'un schema XML. La valeur encoded de use indique que les informations d'encodage spécifiées seront incorporées à l'élément dans le message SOAP généré.

- **f. L'élément service et l'élément port :** L'élément service permet de grouper ensemble une multitude de ports. Il est optionnel et peut se répéter plusieurs fois. L'élément service possède un attribut name qui lui sert de référence unique parmi d'autres éléments service du même document WSDL. Il possède un sous-élément port qui indique l'adresse d'un point terminal pour un élément binding. L'élément port a deux attributs obligatoires qui sont name et binding. L'attribut name lui sert de référence unique parmi d'autres éléments port du même document. L'attribut binding permet d'associer un élément port à un élément binding.

8.3 UDDI (Universal Description, Discovery and Integration) :

8.3.1 définition :

UDDI[11] C'est un format de schéma XML permettant de décrire un service Web en précisant les
méthodes disponibles, les formats des messages d'entrée et de sortie et comment y accéder. Il définit un espace de travail extensible pour décrire des interfaces de Services Web. Il a été premièrement développé par Microsoft et IBM et soumit au W3C devant une commission de 25 compagnies (fait exceptionnel). Il est au cœur de l'espace de travail d'un service Web, desservant une façon de représenter les types de données envoyés dans les messages, les opérations à effectuer sur les messages, et l'envoi des messages sur les réseaux. Comme les autres technologies XML, WSDL est tellement extensible et possède un si grand nombre d'options qu'assurer une compatibilité et une interopérabilité entre les différentes implémentations serait difficile. Si l'expéditeur et le destinataire d'un message peuvent partager et comprendre le même fichier WSDL de la même façon, alors l'interopérabilité sera assuré. UDDI définit les mécanismes permettant de répertorier des Web Services. Ce standard régit donc l'information relative à la publication, la découverte et
l'utilisation d'un Web Service. En d'autres mots, UDDI détermine comment nous

devons organiser l'information concernant une entreprise et les Web Services qu'elle offre à la communauté afin de permettre à cette communauté d'y avoir accès. En fait, UDDI définit un registre des Web Services sous un format XML. Ce registre peut être public, privé ou partagé. Le modèle de données UDDI est défini sous forme de schéma W3C XML Schéma. Tout comme WSDL, UDDI est une création du trio IBM, Microsoft et Ariba. Au départ ils ont développé la spécification puis ont rassemblé quelques 300 entreprises sous le chapeau de l'organisation UDDI.org afin de continuer le développement et de légitimer leurs efforts. UDDI a été déposé à OASIS (Organization for the Advancement of Structured Information Standards) en juillet 2002 afin de permettre à cet organisme de standardisation de parrainer la spécification et d'en assurer son développement technique de façon indépendante. OASIS a officialisé son implication en créant le OASIS UDDI Specification Technical Committee en août 2002.

8.3.2 Les registres UDDI :

Les registres UDDI permettent :

- la recherche et la publication des différents types d'information sur les services et leurs fournisseurs selon un schéma de description,
- la consultation du contenu des registres.

Les registres UDDI sont gérés par les opérateurs UDDI (Microsoft, HP, IBM, etc.). Ces registres sont en réseaux donc ils se partagent les informations publiées. Ainsi une publication dans un registre se propage dans tous les autres. Cette propagation se fait par synchronisation des registres. Les informations présentes dans le registre sont réparties en trois catégories :

1. Les pages blanches : Ces pages comprennent la liste des organisations, leurs informations de contacts et les services qu'elles fournissent.
2. Les pages jaunes : Ces pages classifient les compagnies et les services web selon des taxonomies qui peuvent être standardisées ou définies par l'utilisateur.et décrive d'une manière non technique les services proposés par

les différentes organisations.

3. Les pages vertes : Ces pages décrivent comment un service web peut être invoqué.

Elles fournissent des références vers le document WSDL du service qui est stocké normalement à l'extérieur du registre. Sur la figure il y a deux exemples de registres

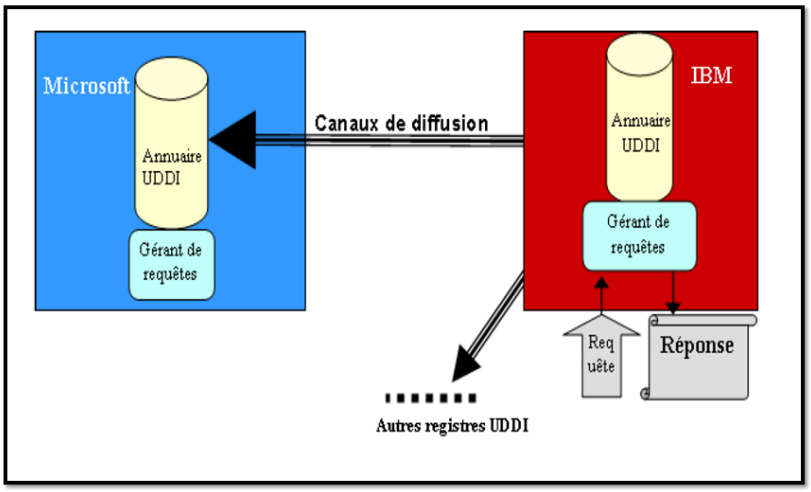

FIGURE 1.11 – Architecture des registres UDDI

UDDI. Il s'agit des registres de Microsoft et de IBM. La synchronisation entre les contenus des deux registres est illustrée par les flèches à double traits et portant la mention canaux de diffusion. Sur le registre IBM, nous avons illustré un échange entre le registre et un client par une flèche (requête) et une feuille de réponse.

8.3.3 Structure UDDI :

La phase de description d'un service comprend la description du fournisseur de service, la description du service lui-même ainsi que les informations techniques relatives au service. Pour ce faire, UDDI se base sur les éléments suivants : BusinessEntity, BusinessService, BindingTemplate, tModel, Publisher Assertions. Voici l'information véhiculée par chacun de ces éléments :

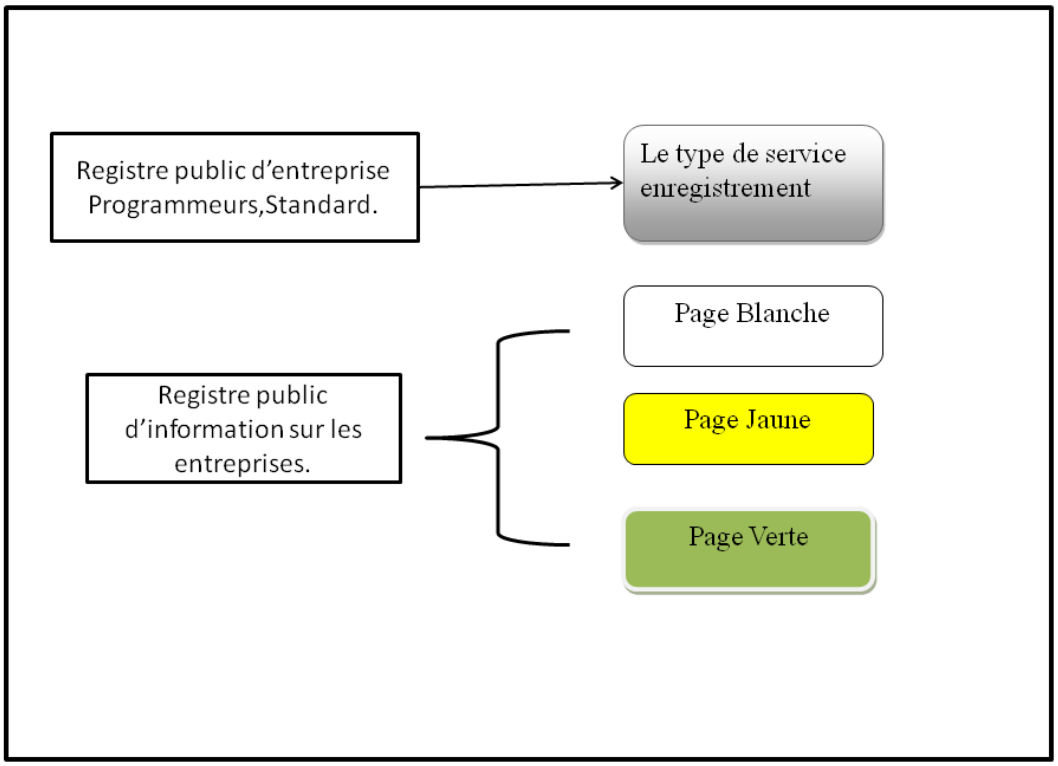

FIGURE 1.12 – Données du registre UDDI

- BusinessEntity : donne des informations sur l'activité ou l'organisation offrant le service ;
- BusinessService : donne des informations sur les services offerts par une organisation décrite dans BusinessEntity ;
- BindingTemplate : donne des informations techniques nécessaires à l'utilisation d'un service en particulier ;
- tModel (technical models) : donne une représentation du service en modèle technique en utilisant des concepts réutilisables tels que : le type du SW, le protocole utilisé par le SW, la catégorie du système, etc ;
- Publisher Assertions : décrit la relation de partenariat d'une BusinessEntity avec une autre dans le cadre de l'accomplissement d'un service.

8.3.4 Les Caractéristiques de UDDI :

- Neutre en terme de protocole –comme tout registre, il peut y avoir des pointeurs sur n'importe quoi (email, web page ...)
- Possibilité de faire des recherches par domaine d'activité, service, Web Service, binding
- Pas de QoS
- Nécessite un modérateur Risques d'entrées erronées, de doublons, de fraude
- Nœuds privés et publics autorisés

9 Les avantages et les inconvénients des Services Web :

9.1 Les avantages

- Peut appeler à distance à travers un réseau, indépendamment du système d'exploitation et du langage de
  programmation utilisé. Par exemple, un client écrit en PHP s'exécutant sur

Unix peut utiliser un service web développé en Java qui s'exécute sur Windows et inversement.

- permettent à des applications supportant différents processus d'une organisation ou de différentes organisations, de communiquer entre elles et/ou d'échanger des données facilement et à peu de frais.
- Les services Web peuvent fonctionner au travers de nombreux pare-feux sans nécessiter des changements sur les règles de filtrage.

9.2 Les inconvénients

- Les web services ne traitent que la syntaxe et pas la sémantique.
- Les web services possèdent de faible performance.
- Il n'est pas toujours évident de composer des services complexes avec d'autres services existants.

10 Conclusion

Les services Web constituent une technologie idéale pour l'intégration et l'interopérabilité des systèmes répartis. Basés sur le standard XML, ils sont caractérisés par leurs indépendances aux plates formes et aux systèmes d'exploitation, ce qui a impliqué leur adoption par les différentes organisations commerciales et industrielles offrant leurs services à travers le Web, et par conséquent l'augmentation du nombre de services offerts. De ce fait, les Web Services sont donc plus un phénomène ou un concept, voire un contexte, qu'une technologie. Il est évident que les Web Services reposent sur diverses technologies, mais ils représentent surtout une volonté commune des manufacturiers, des organismes de standards et des utilisateurs de développer des outils permettant une réelle interopérabilité.

Chapitre 2

Métaheuristique et la Sélection des Services Web

1 Introduction

Les organisations sont de plus en plus nombreuses à se tourner vers des architectures à base de services Web pour le développement et l'intégration d'applications ou de systèmes d'information. Il arrive très fréquemment que de nombreux services répondent à un même ensemble de besoins fonctionnels. Ces services se distinguent les uns des autres par leurs propriétés non fonctionnelles. On intéresse ici à les comparer selon des critères de qualités (réputation, fiabilité, etc.).On propose d'étudier la notion de communauté faisant référence à un ensemble de services de même fonctionnalité. Une communauté offre à ses utilisateurs (développeurs d'application, fournisseurs de services, applications clientes) des fonctions pour connaître des informations liées à la communauté, pour sélectionner, puis utiliser un ou plusieurs des services qui y sont enregistrés. Dans ce chapitre on va présenter une partie de l'état de l'art qui concerne l'emploi des méta-heuristiques dans l'optimisation mono-objectif et multi-objectif comme on présente la sélection des services web avec une explication de ça stratégie et la modélisation possible de ce probleme.

2 Exemple de motivation

Les Web Services fournissent une solution pour le transport de messages entre deux partenaires. Cependant, ils ne décrivent pas comment interpréter les messages transmis. Il appartient donc aux partenaires de s'entendre sur cette sémantique. Pour illustrer ce problème, prenons l'exemple d'un patient qui désirer effectuer une consultation. Pour cela il a besoin d'élaborer les quatre services suivants :

- Le service prescription médicale : ce service consiste à prescrire un traitement sur un document « l'ordonnance ». Cette ordonnance consigne la prescription médicale qui peut être certes, des médicaments mais également des examens radiologiques, biologiques, des traitements physiques et des actes de kinésithérapie, ainsi que des cures thermales ou des règles d'hygiène et de diététique. Pour cela se service a comme entrées les renseignements des patients (numéro, nom , prénom,...etc) et comme sorties les tests, recommandes, des analyses...etc.
- Le service de scanner : ce service permet la radiologie conventionnelle,y compris la mammographie, l'échographie, le scanner, certains examens de radiologie vasculaire.qui a comme entrés les renseignements des patients (numéro, nom, prénom...etc) et comme sortie l'image diagnostic.
- le service de paiement bancaire : ce service fournis par un établissement de paiement ou par une banque permettant de réaliser des opérations de paiement, il a comme entrées le numéro et la date de validation de la carte bancaire du patient, et comme sortie le reçu de paiement (montant, date et heure, numéro d'autorisation)
- le service de la décision finale : pour ce service il contient comme entrées le numéro du patient et l'image diagnostic et comme sortie le diagnostic.

♠ D'après l'exemple les services web permettent ainsi des échanges entre les applications, il est aussi possible d'utiliser un service web d'une application dans une autre application cela permet l'ouverture d'entreprises à d'autres entreprises c'est le

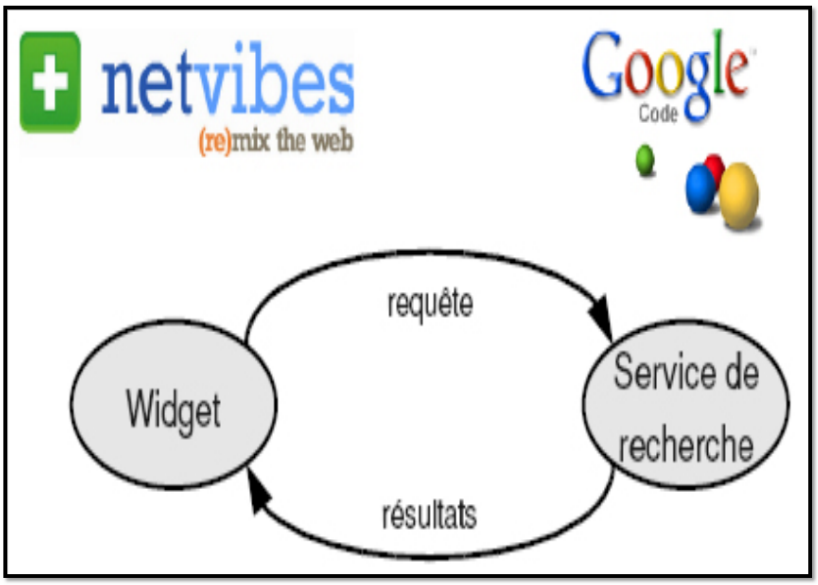

FIGURE 2.1 – L'utilisation du service web Google Code par Netvibes

cas par exemple du site Netvibes,qui utilise le moteur de recherche de Google ,via le service web Google code ,au lieu de créer son propre moteur de recherche.

2.1 Le scenario

Le scenario serait d'un patient X, a besoin de consulter un médecin a cause de sa maladie, ce patient a besoin de faire un scanner il doit prendre un rendez vous avec un paiement limité. Pour une consultation médicale, le patient voudrait tout d'abord connaitre le prix de scanner, il doit ensuite avoir le dévie de son scanner et aussi ses moyens de paiement, ce patient peut avoir des exigences particulières sur la qualité des opérations des services. Le patient peut définir son besoin par une requête de service permet également à l'utilisateur de spécifier les exigences relatives à la qualité des opérations de service cette requête spécifie le fonctionnement que l'utilisateur souhaite récupérer en termes d'opérations de services. Les utilisateurs peuvent avoir des préférences particulière pour certains paramètres de QoS.par exemple le patient X à besoin d'un scanner de corps avec une réputation élevé, c'est-à-dire il préfère

faire le scanner avec le matériel le plus réputé avec un cout moyen c'est-à-dire le reçu de paiement ne doit pas dépasser par exemple 300dollars. La latence et le coût prennent des valeurs scalaires dans R+ , la disponibilité et la fiabilité représentent des valeurs de probabilité (une valeur réelle entre 0 et 1), et la réputation varie sur l'intervalle [0,5].

3 Etat de l'art

3.1 Optimisation mono-objectif

Un problème d'optimisation en général est défini par un espace de recherche S et une fonction objectif « f ». Le but est de trouver la solution $s^* \in S$ de meilleure qualité $f(s^*)$. Suivant le problème posé, on cherche soit le minimum soit le maximum de la fonction f . Dans la suite, nous aborderons les problèmes d'optimisation essentiellement sous l'aspect minimisation, maximiser une fonction f étant équivalent à minimiser -f .
L'équation suivante résume la définition précédente.

$$S^* = min(f(s) \setminus s \in S)$$

De plus, un problème d'optimisation peut présenter des contraintes d'égalité et/ou d'inégalité sur les solutions candidates $s^* \in S$, être multi-objectif si plusieurs fonctions objectifs doivent être optimisées ou encore dynamique, si la topologie de « f » change au cours du temps. Il existe de nombreuses méthodes déterministes (ou exactes) qui permettent de résoudre certains types de problèmes d'optimisation en un temps fini. Cependant, ces méthodes nécessitent que la fonction objectif présente un certain nombre de caractéristiques, telles que la convexité, la continuité ou encore la dérivabilité. Parmi les méthodes les plus connues, on peut citer les méthodes de programmation linéaire [Schrijver, 1998], quadratique[Nocedal et al,1999] ou dynamique [Bertsekas,2000], la méthode de Newton[Nocedal et al, 1999], la méthode du

simplex[Nelder et al,1965] ou encore la méthode du gradient [Avriel, 2003].

3.1.1 Métaheuristiques pour l'optimisation mono-objectif difficile

Les métaheuristiques[2] sont une famille d'algorithmes stochastiques destinés à la résolution des problèmes d'optimisation. Leur particularité réside dans le fait que celles-ci sont adaptables à un grand nombre de problèmes sans changements majeurs dans leurs algorithmes, d'où le qualificatif méta. Leur capacité à optimiser un problème à partir d'un nombre minimal d'informations est contrebalancée par le fait qu'elles n'offrent aucune garantie quant à l'optimalité de la meilleure solution trouvée. Seule une approximation de l'optimum global est donnée. Cependant, du point de vue de la recherche opérationnelle, ce constat n'est pas forcément un désavantage, étant donné que l'on préférera toujours une approximation de l'optimum global trouvée rapidement qu'une valeur exacte trouvée dans un temps rédhibitoire. Les métaheuristiques sont des méthodes qui ont, en général, un comportement itératif, c'est-à-dire que le même schéma est reproduit un certain nombre de fois au cours de l'optimisation, et qui sont directes, dans le sens où elles ne font pas appel au calcul du gradient de la fonction. Ces méthodes tirent leur efficacité du fait qu'elles sont moins facilement piégeables dans des optima locaux, car elles acceptent, au cours du traitement, des dégradations de la fonction objectif et la recherche est souvent menée par une population de points et non un point unique. les métaheuristiques sont majoritairement conçues pour résoudre des problèmes à variables discrètes, mais font de plus en plus l'objet d'adaptations aux problèmes à variables continues. De plus, de par leur variété et leur capacité à résoudre des problèmes très divers, les métaheuristiques sont assez facilement sujettes à extensions. Parmi celles-ci, on peut citer :

- ⋆ Les métaheuristiques pour l'optimisation multi-objectif. Le but est ici non pas de trouver un optimum global mais de trouver un ensemble d'optimum qui forment une surface de compromis pour les différents objectifs du problème ;
- ⋆ Les métaheuristiques pour l'optimisation multimodale [Goldberg et al., 1987],

où l'on ne cherche plus l'optimum global, mais l'ensemble des meilleurs optimum locaux ;

⋆ Les métaheuristiques pour l'optimisation dynamique [Branke, 2001 ; Ourique et al., 2002 ; DiCaro et al., 1998], où il faut approcher l'optimum à chaque pas de temps, car la fonction objectif change de topologie au cours du temps ;

⋆ Les métaheuristiques hybrides [Talbi, 2002], qui combinent différentes métaheuristiques, afin d'en tirer les avantages respectifs ;

⋆ Les métaheuristiques parallèles [Alba, 2005], pour lesquelles on cherche à accélérer le calcul, en distribuant la charge de calcul sur plusieurs calculateurs. L'utilisateur est, certes, demandeur de méthodes rapides et efficaces, mais il est aussi demandeur de méthodes simples d'utilisation. Un enjeu majeur des métaheuristiques est donc de faciliter le choix des méthodes et de simplifier leur réglages. Le monde des métaheuristiques est un monde en constante évolution. De nombreuses méthodes sont proposées chaque année pour tenter d'améliorer la résolution des problèmes les plus complexes. Du fait de cette activité permanente, un grand nombre de classes de métaheuristiques existe actuellement. Les méthodes les plus courantes sont le recuit simulé, la recherche tabou, les algorithmes évolutionnaires, les algorithmes de colonies et fourmis ou encore l'optimisation par essaim particulaire.dont on cite quelque algorithmes.

- L'Algorithme recuit simulé :

  L'origine de la méthode du recuit simulé[2] provient de la métallurgie, où, pour atteindre les états de basse énergie d'un solide, on chauffe celui-ci jusqu'à des températures élevées, avant de le laisser refroidir lentement. Ce processus est appelé le recuit. Le recuit simulé a été développé simultanément par Kirkpatrick et al. [Kirpatrick et al., 1983] et Cerny [Cerny, 1985]. Le recuit simulé repose sur l'algorithme de Metropolis [Metropolis et al., 1953 ;Hastings, 1970] . Cette procédure permet de sortir des minima locaux avec une probabilité élevée si la température T est élevée et, quand l'algorithme atteint de très basses températures, de conserver les états les plus probables.

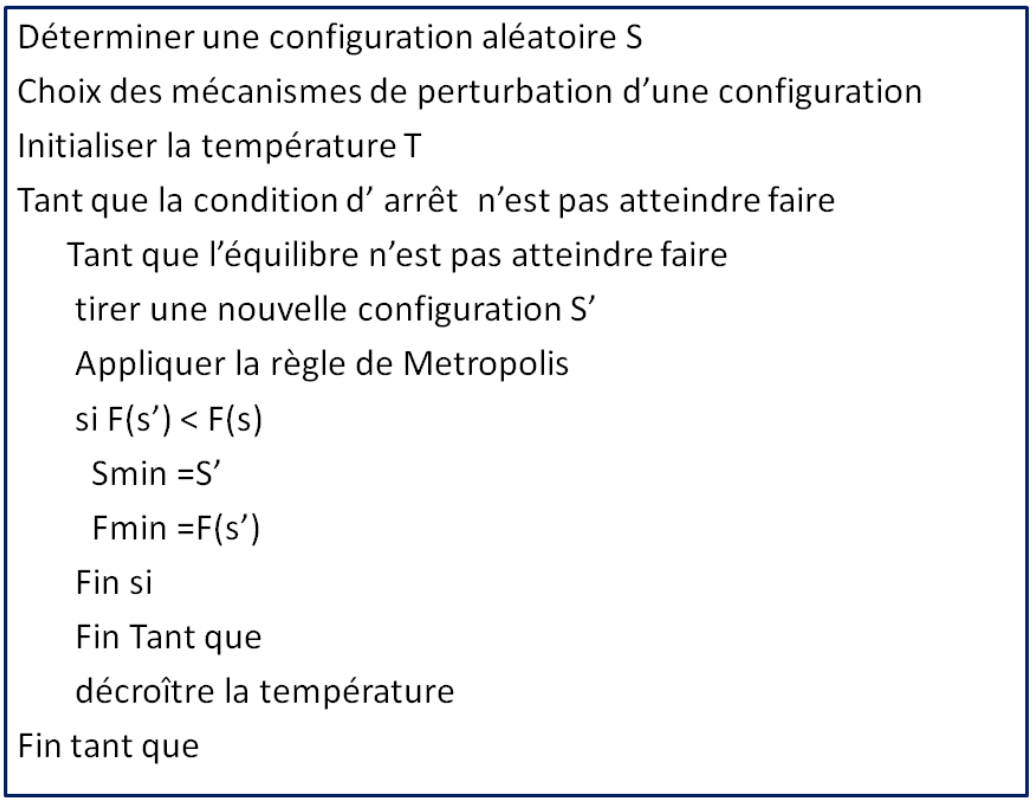

FIGURE 2.2 – Algorithme de recuit simulé

- La Recherche Tabou :

  L'algorithme de Recherche Tabou a été introduit par Glover en 1986 [Glover, 1986]. Le but de cette méthode est d'inculquer aux méthodes de recherche locale un surcroît d'intelligence. L'idée ici est d'ajouter au processus de recherche une mémoire qui permette de mener une recherche plus intelligente dans l'espace des solutions. Comme l'algorithme de recuit simulé, la méthode de recherche tabou fonctionne avec une seule configuration courante, qui est actualisée au cours des itérations successives. La nouveauté ici est que, pour éviter le risque de retour à une configuration déjà visitée, on tient à jour une liste de mouvements interdits, appelée « liste tabou ». Cette liste contient m mouvements $(t \mapsto s)$ qui sont les inverses des m derniers mouvements $(t \mapsto s)$ effectués. L'algorithme modélise ainsi une forme primaire de mémoire à court terme. Dans sa forme de base, l'algorithme de recherche tabou présente l'avantage de comporter moins de paramètres que l'algorithme de recuit simulé. Cependant, l'algorithme n'étant pas toujours performant, il est souvent approprié de lui ajouter des processus d'intensification et/ou de diversification, qui introduisent de nouveaux paramètres de contrôle [Glover et al., 1997].

3.2 Optimisation multi-objectif

Les métaheuristiques ont toutes été conçues à l'origine pour résoudre des problèmes mono-objectif. Cependant, devant l'omniprésence des problèmes multi-objectifs dans les cas réels, de nouvelles méthodes ont dû être créées pour résoudre de tels problèmes. Après avoir défini le problème d'optimisation multi-objectif, cette partie présentera de manière non exhaustive un état de l'art des métaheuristiques conçues pour résoudre les problèmes multi-objectifs.

⋆ Définition du problème : Un problème d'optimisation multi-objectif est un problème du type :

$$f(\overrightarrow{x}) = (f1(\overrightarrow{x}), f2(\overrightarrow{x}), ...fk(\overrightarrow{x}))$$

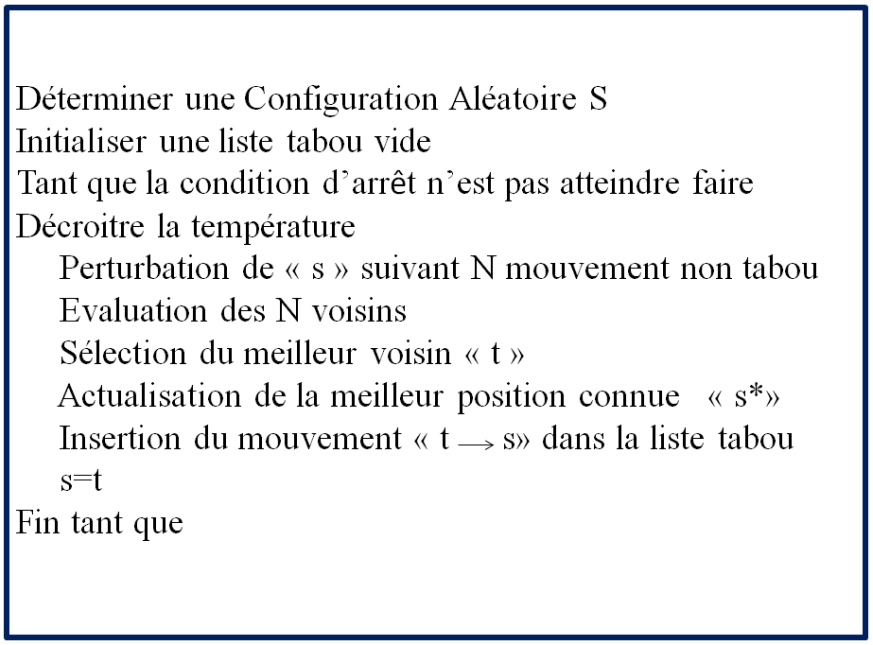

FIGURE 2.3 – Algorithme tabou

Minimiser sous les contraintes suivantes :

$$g_i(\overrightarrow{x}) < 0, i = 1, .., m.$$

$$h_i(\overrightarrow{x}) < 0, i = 1, .., p.$$

ou $\overrightarrow{x} = x1, ..xd$ est une position dans l'espace de recherche

$f_i : R^D \longmapsto R, i = 1, .., k$sont les fonctions objectifs du problème,

$g_i : R^D \longmapsto R, i = 1, .., m$ sont les contraintes d'inégalité du problème et

$h_i : R^D \longmapsto R, i = 1, .., p$ sont les contraintes d'égalité du problème. Dans le cas multiobjectif, le concept d'optimum est différent que dans le cas mono-objectif. En effet, on n'est plus ici à la recherche d'un unique optimum global, mais plutôt d'une surface de solutions qui offrent un bon compromis entre les différents objectifs.

3.2.1 Métaheuristiques pour l'optimisation multi-objectif

- Algorithme de recuit simulé :

  La méthode de recuit simulé en optimisation multi-objectif a d'abord été abordée sous l'angle agrégatif [Serani, 1992 ; Friesz et al., 1993]. Les deux méthodes les plus populaires sont la méthode MOSA (Multiple Objective Simulated Annealing) proposée par Ulungu et al. [Ulungu et al.,1999] et la méthode PASA (Pareto Archived Simulated Annealing) proposée dans [Engrand, 1997]. MOSA utilise les caractéristiques du recuit simulé pour rechercher le plus efficacement possible les solutions non-dominées. PASA utilise une fonction d'agrégation des fonctions objectifs, couplée avec un système d'archivage des solutions non-dominées.

- Algorithmes évolutionnaires :

  Les algorithmes évolutionnaires sont très largement utilisés pour résoudre des problèmes multi-objectifs. Une étude comparative des algorithmes évolutionnaires pour l'optimisation multi-objectif est disponible dans [Zitzler et al., 1999]. Parmi les méthodes utilisant l'approche Pareto, on distingue deux familles d'algorithmes : les non-élitistes et les élitistes. Parmi les méthodes non-élitistes, l'algorithme MOGA (Multiple Objective Genetic Algorithm) proposé par Fonseca et al. [Fonseca et al., 1995] est l'un des plus connus. Cet algorithme se base sur le classement des individus suivant le nombre d'individus dominés. NSGA (Non Dominated Sorting Genetic Algorithm) [Srivinas et al., 1995] propose de diviser la population en plusieurs groupes,fonctions du degré de domination de chaque individu.

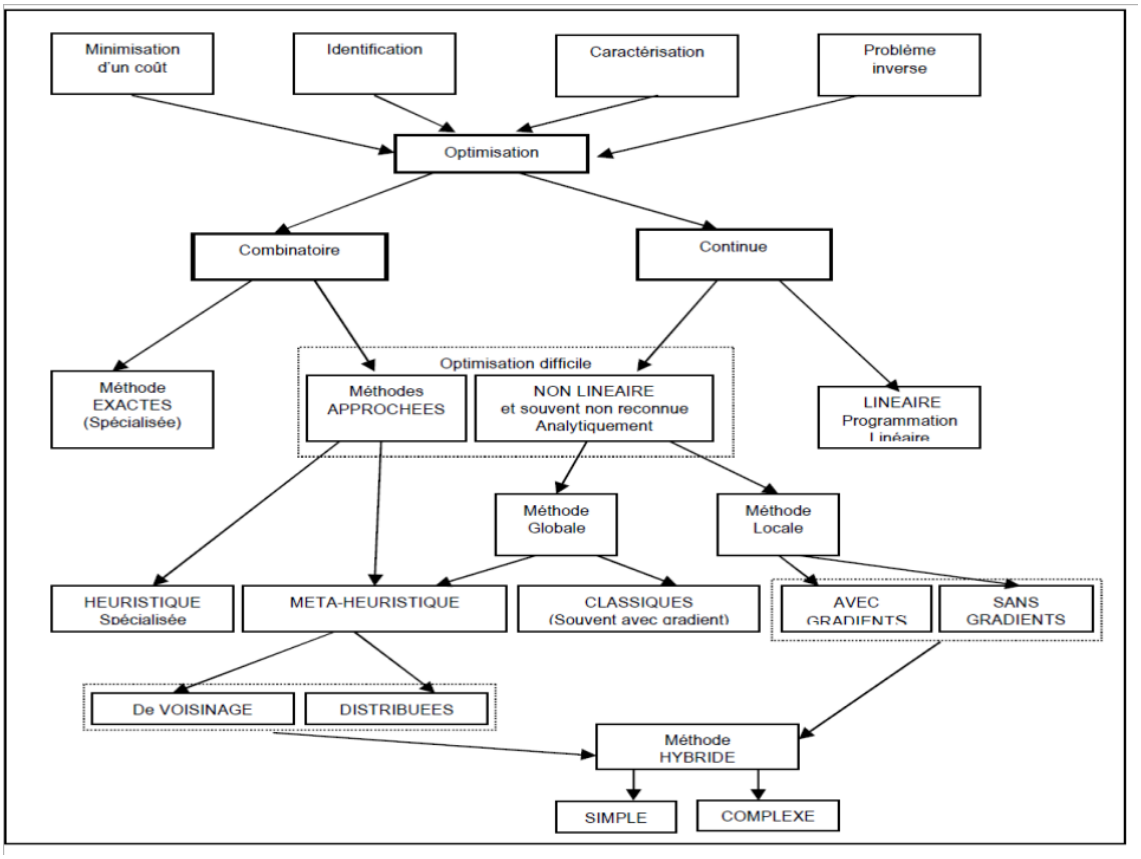

FIGURE 2.4 – Une hiérarchie des méthodes d'optimisation

4 La Sélection des services web basée sur la qualité de service

Le principal objectif d'une communauté de services web est d'offrir un cadre à la recherche et à la sélection dynamique de services web et ainsi de pallier les déficiences des annuaires UDDI . C'est aussi un moyen pour supporter la composition dynamique (d'un grand nombre) de services ou encore de permettre la substitution (à la conception ou à l'exécution) de services. Contrairement à un annuaire UDDI, une communauté est spécialisée dans un domaine spécifique. Les services d'une communauté diffèrent entre eux sur des propriétés non fonctionnelles comme par exemple, des critères de qualité (disponibilité, fiabilité, réputation,.etc.). Ces critères de qualité sont définis via un modèle de qualité spécifique à une communauté donnée. La sélection consiste à choisir, parmi les services web découverts, ceux qui répondent au mieux aux exigences de l'utilisateur sur la base des besoins fonctionnels et/ou non fonctionnels. Les besoins non fonctionnels des services web sont généralement exprimés à l'aide des critères de QoS. Dans une sélection de services web basée sur la QoS deux cas se présentent :

- ⋆ Si la réponse à la requête d'un client exige la sélection d'un seul service web non composite, alors la sélection est très simple. Le candidat qui présente la meilleure QoS sera désigné pour répondre à la demande du client.
- ⋆ Si la réponse à la requête d'un client exige la combinaison de plusieurs services existants, alors la sélection dans ce cas sera plus complexe du fait qu'il faut choisir la combinaison des services composants qui répond mieux aux besoins des clients. La modélisation de ce problème dépend de la nature de la stratégie de sélection : locale ou globale.

4.1 Les Stratégies de sélection de services web

- Dans la littérature, il existe deux stratégies de sélection de services web : une stratégie de sélection locale et une autre globale. Chaque stratégie est définie en

fonction de la nature des contraintes de QoS imposées. En fait, ces contraintes n'ont pas un caractère obligatoire, mais permettent de définir les limites de fonctionnement d'un service web composant ou composite.

⋆ La stratégie de sélection locale a pour objectif de choisir le meilleur service web pour chaque tâche individuelle à part entière en considérant des contraintes de QoS relatives à chaque tâche plutôt qu'en considérant des contraintes de QOS globales exprimées pour l'ensemble des tâches. Pour le cas d'une application temps réel, il s'agit de sélectionner pour chaque tâche un service web apte à l'exécuter en prenant en compte les contraintes temps réel locales et d'autres préférences imposées pour chaque tâche. Par exemple, la durée d'exécution d'une tâche devra avoir un temps de réponse qui ne dépasse pas quelques minutes, le service web exécutant une tâche doit être disponible à 100%, etc.

⋆ La stratégie de sélection globale a pour but de choisir la combinaison de services web qui garantit la meilleure qualité globale en tenant compte des contraintes de QOS et des préférences globales assignées pour l'ensemble des tâches.

4.2 Modélisation du problème de sélection de services web

La modélisation du problème de sélection dépend de la nature de la stratégie de sélection appliquée : locale ou globale.

⋆ Pour la stratégie locale, la modélisation du problème de sélection est destinée aux services web composants dont l'enchainement est défini dès l'étape de conception. « Jaeger et Muhl » et « Liu, Ngu et al. » appliquent la technique SAW (Simple Additive Weighting) pour modéliser le problème de sélection de services web basée sur la QOS. Etant donné un ensemble de services ayant la même fonctionnalité mais des valeurs de critères de QOS différents, le principe de la technique SAW consiste à affecter un vecteur de QOS à chaque service web candidat dans la sélection. Ce vecteur contient les valeurs des critères de QoS qui caractérisent chaque service web. Sur la base de ce vecteur, il s'agit d'appliquer les deux phases suivantes :

⋆ Phase d'ajustement : Elle consiste à ajuster les critères de qualité de dimension croissante (comme la disponibilité) et décroissante (comme le temps de réponse) inclus dans chaque vecteur en appliquant des équations de normalisation afin de rendre les valeurs des critères de QOS homogènes.

- Phase d'attribution de poids : Lors de cette phase, il s'agit d'attribuer un poids à chaque critère de QOS inclus dans le vecteur de QOS puis de calculer le score de chaque service web. Ce score représente une somme pondérée des valeurs normalisées de ces critères et le service web qui a le score le plus élevé est celui qui est attribué à la tâche en question.

⋆ Lorsqu'il s'agit d'appliquer la stratégie de sélection globale, le problème de sélection de services web composants peut être modélisé comme un problème d'optimisation combinatoire. Il s'agit, pour une composition fonctionnelle de services web et une QOS donnée, de trouver les services web qui sont disponibles et qui peuvent satisfaire une ou plusieurs contraintes de QOS tout en considérant comme objectif d'optimiser (maximiser ou minimiser) une fonction coût. Ainsi, lorsqu'il s'agit de considérer une seule contrainte de QOS, « Yu et Lin » proposent de modéliser le problème de sélection comme un problème de sac à dos. Et lorsqu'il s'agit de considérer plus qu'une contrainte de QOS, « Yu et Lin » et « Zeng, Benatallah et al. » proposent de modéliser le problème de sélection comme un problème de programmation linéaire.

5 Conclusion

Nous avons présenté dans ce chapitre les problèmes d'optimisation difficile, qu'ils soient mono-objectif ou multi-objectif. Une famille de méthodes de résolution de ces problèmes a été présentée : les métaheuristiques. Les métaheuristiques sont des méthodes stochastiques qui visent à résoudre un large panel de problèmes, sans pour autant que l'utilisateur ait à modifier leur structure. Ces méthodes sont inspirées d'analogies avec des domaines aussi variés que la physique, la génétique ou encore

l'éthologie. Les métaheuristiques ont vite rencontré un vif succès grâce à leur simplicité d'emploi mais aussi à leur forte modularité. On a vu suivant la nature du problème posé (continu/discret, mono-objectif/multi-objectif, etc.), les métaheuristiques sont facilement adaptables et/ou hybridables en vue d'obtenir les meilleures performances possibles. Ainsi on a vu comment peut sélectionner un ensemble des services web qui peut être selon deux stratégies local ou global et comment faire la modélisation de ce problème.

Chapitre 3

L'Algorithme de colonies de fourmis(ACO)

1 Introduction

L'informatique peut traiter plusieurs problèmes avec des méthodes mathématiques simples. Cependant ces méthodes ne sont pas adaptées pour des problèmes complexes, qui apparaissent par exemple dans le domaine de la robotique ou de la recherche opérationnelle dont la source d'inspiration utilisée dans la branche de l'intelligence artificielle est la modélisation de systèmes complexes naturels en particuliers les systèmes biologiques qui sont étudient d'après une observation d'un être vivant ou d'un groupe d'être vivant. Dans ce chapitre on doit s'intéresser aux modèles qui copient les comportements sociaux des animaux c'est-à-dire l'intelligence collective qu'il est possible d'observer chez les abeilles, les guêpes, certaines espèces de poissons, les termites ou plus particulièrement chez les fourmis qui sera la principale espèce étudiée. En recherche opérationnelle, et plus précisément dans le domaine de l'optimisation difficile, pour lesquels on ne connait pas de méthode classique plus efficace la majorité des méthodes sont inspirées par de telles études, et notamment par la biologie. Dans le cadre de l'optimisation, l'approche de colonies de fourmis a donné lieu à la création de nouvelles métaheuristiques « une classe métaheuristique ».

2 Caractéristiques des fourmis

- Selon [Dorigo 1999]. il y a 108 insectes vivants, uniquement 2% d'entre eux sont sociaux ,50% des insectes sociaux sont des fourmis.
- La taille d'une population de fourmis varie entre 30 et plusieurs millions d'individus
- Dans les algorithmes ACO [Dorigo & Di Caro, 1999, Dorigo & Stüzle, 2004], les fourmis se déplacent en appliquant une politique de décision stochastique locale qui se base sur l'utilisation des traces de phéromone et d'une information heuristique spécifique au problème traité.
- Des coeficients $\alpha et \beta$. permettent de contrôler l'importance relative des deux éléments.
- Chaque fourmi possède une forme de mémoire afin d'obliger celle-ci à former une solution admissible.
- En se déplaçant, les fourmis construisent incrémentalement des solutions au problème d'optimisation. Une fois qu'une fourmi a construit une solution, ou lorsque la solution est en cours de construction, la fourmi évalue la solution (partielle) et dépose la phéromone dans le composant ou la connexion qu'elle à utilisé.
- Cette information de phéromone dirigera la recherche des futures fourmis
- L'intensité de ces traces décroît dans le temps par un facteur constant appelé coefficient d'évaporation. D'un point de vue pratique, l'évaporation de phéromone est nécessaire pour éviter une convergence prématurée de l'algorithme vers une région sous-optimale. Ce processus implémente une forme utile d'oubli favorisant l'exploration de nouvelles aires de recherche.

3 Comportement des fourmis

3.1 Les taches des différents types d'individu des fourmis

Tache	Type d'individu
Repreduction	Reine
Défence	Fourmis Soldat
Collecte de Nouriture	Fourmis travailleuses spécialisées
Entretien des œufs	Fourmis travailleuses spécialisées
Ménage	Fourmis travailleuses spécialisées
Construction Maintenance du nid	Fourmis travailleuses spécialisées

TABLE 3.1 – table des taches des fourmis

chez les fourmis le comportement des femelles change beaucoup après la fécondation,au cours des recherches des biologistes sur la fondation des colonies de Messeor ,ils ont eu maintes fois l'occasion d'observer le comportement de la reine avant l'éclosion des premiers ouvrières et celui qu'elle a en leur présence. Ce comportement particulier apparait tout différent dans les relations de la jeune fondatrice avec la nourriture ,les autres femelles,les ouvieres et le couvain. maintes fois l'occasion d'observer le comportement de la reine avant l'éclosion des premiers ouvrières et celui qu'elle a en leur présence. Ce comportement particulier apparait tout différent dans les relations de la jeune fondatrice avec la nourriture ,les autres femelles,les ouvieres et le couvain.

- Les fourmis artificielles peuvent être caractérisées comme une procédure de construction stochastique construisant des solutions sur le graphe G= (C ; L). En général, les fourmis tentent d'élaborer des solutions faisables mais, si nécessaire, elles peuvent produire des solutions infaisables. Les composants et les connexions peuvent être associés à des pistes de phéromone (mettant en place une mémoire adaptative décrivant l'état du système) et à une valeur heuristique (représentant une information à priori sur le problème, ou venant d'une source autre que celle des fourmis ; c'est bien souvent le coût de l'état

en cours). Les pistes de phéromone et la valeur de l'heuristique peuvent être associées soit aux composants, soit aux connexions.

- Chaque fourmi dispose d'une mémoire utilisée pour stocker le trajet effectué, d'un état initial et de conditions d'arrêt.
- Les fourmis se déplacent d'après une règle de décision probabiliste fonction des pistes de phéromone locales, de l'état de la fourmi et des contraintes du problème. Lors de l'ajout d'un composant à la solution en cours, les fourmis peuvent mettre à jour la piste associée au composant ou à la connexion correspondante.
- Une fois la solution construite, elles peuvent mettre à jour la piste de phéromone des composants ou des connexions utilisées.
- Enfin, une fourmi dispose au minimum de la capacité de construire une solution du problème.
- Les fourmis ont la particularité d'employer pour communiquer des substances volatiles appelées phéromones. Cette figure montre le plus court chemin à trouver la capacité des colonies de fourmis. Entre la fourmilière et la seule source de nourriture existe deux chemins de longueurs différentes. Dans les quatre graphiques, les traces de phéromone sont représentées par des lignes en pointillés dont l'épaisseur indique la force des sentiers.

3.2 Le principe

Les fourmis[6] utilisent les pistes de phéromone pour marquer leurs trajets, par exemple entre le nid et une source de nourriture. Une colonie est ainsi capable de choisir (sous certaines conditions) le plus court chemin vers une source à exploiter [Goss et al. 1989, Beckers et al.1992], sans que les individus aient une vision globale du trajet et capable de s'adapter aux changements de l'environnement. Le moyen de communication est le phéromone. Une fourmi qui se déplace, laisse une quantité variable de phéromone sur son chemin, qui est détectée par les prochaines fourmis, et qui détermine avec une grande probabilité leurs chemins. Plus des fourmis suivent

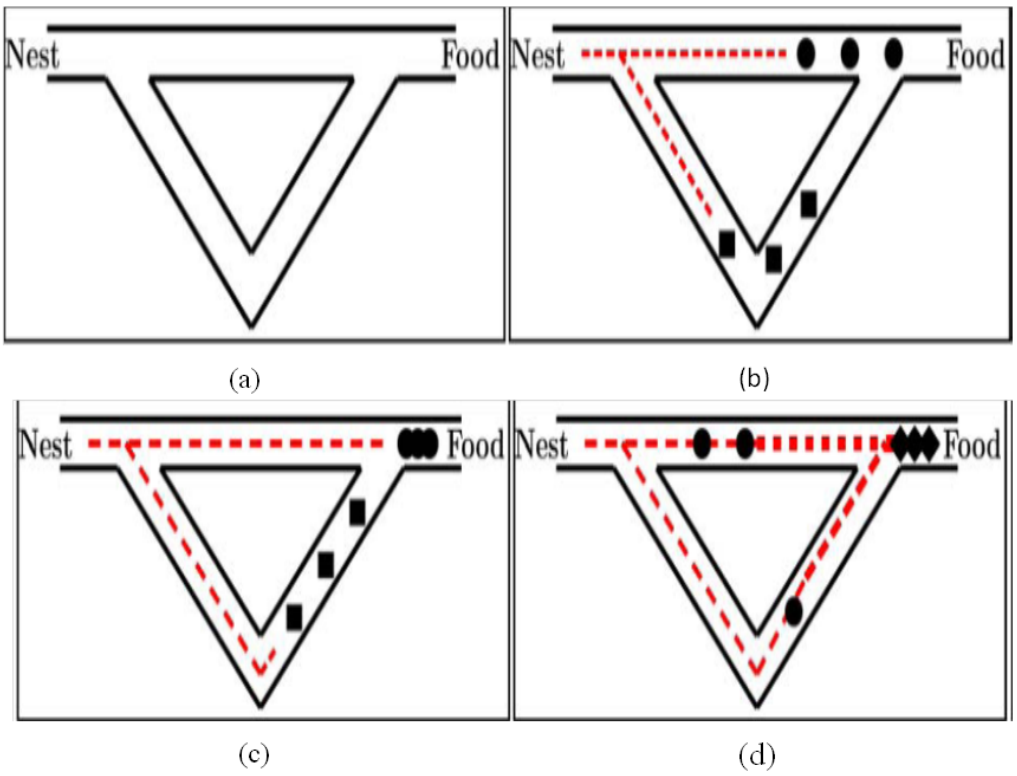

FIGURE 3.1 – Un cadre expérimental qui démontre le plus court chemin à trouver la capacité des colonies de fourmis

(a) :toutes les fourmis sont dans le nid. il n'ya pas de phéromone dans l'environnement (b) :les mises en chantier de recherche de nourriture dans une probabilité de 50% des fourmis prennent le chemin de courte durée (symbolisé par le cercle) et 50% prendre le long chemin vers la source de nourriture (symbolisée par losanges) (c) :les fourmis qui ont pris le chemin d'accès court sont arrivées plus tôt à la source de nourriture, par conséquent, lors du retour, la probabilité de reprendre le chemin d'accès court est plus élevé (d) :la trace de phéromone sur le chemin d'accès court reçoit, en probabilité, un plus fort renforcement et la probabilité qui prendre ce chemin se développe.

Finalement due faire l'évaporation de la phéromone sur le long chemin, toute la colonie sera la probabilité d'utiliser le chemin court.

le chemin plus le chemin devient attractif.

3.3 Exemple

il n'existe pas de trace de phéromone sur les deux chemins, elles vont choisir une des deux alternatives avec la même probabilité. Donc environ 15 fourmis décident d'aller à C et les 15 autres fourmis décident d'aller à H. A t=1, les 30 prochaines fourmis sortant de B (et de D) peuvent détecter les traces de phéromone. La trace de phéromone sur BCD est deux fois plus intense que celle sur BHD, car 30 fourmis sont passées par BCD (15 de A et 15 de D), tandis que 15 fourmis seulement sont passées par BHD. C'est pourquoi maintenant 20 fourmis prennent BCD et 10 prennent BHD (de même 20 prennent DCB et 10 prennent DHB à partir du point D). Une fois encore, plus de phéromone est déposée sur le plus court chemin. Ce processus est répété à chaque unité de temps et ainsi les traces de phéromone sont renforcées. Grâce à ce processus auto-catalytique, toutes les fourmis vont très rapidement choisir le chemin le plus court

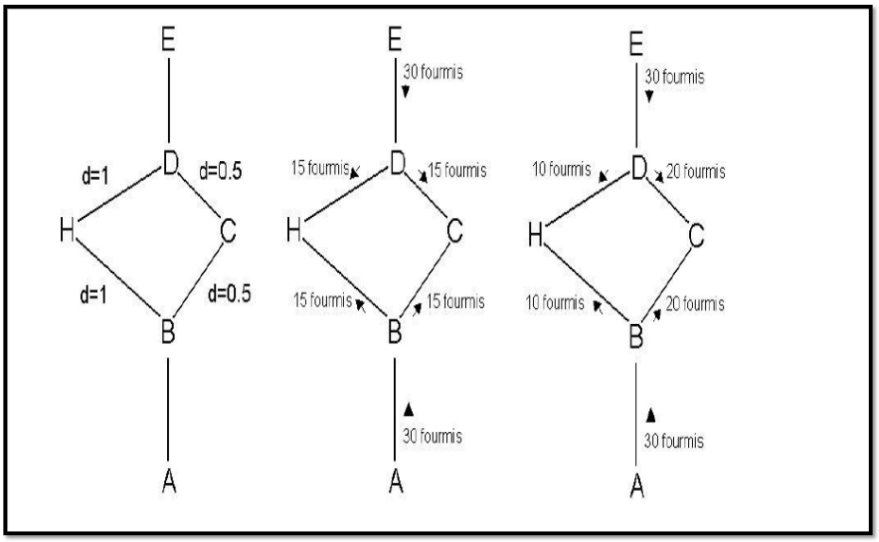

FIGURE 3.2 – comment les fourmis trouvent le plus court chemin

4 L'algorithme de colonies des fourmis

4.1 Représentation du problème

Le problème[7] est représenté par un jeu de solutions, une fonction objective assignant une valeur à chaque solution et un jeu de contraintes. L'objectif est de trouver l'optimum global de la fonction objectif satisfaisant les contraintes. Les différents états du problème sont caractérisés comme une séquence de composants. On peut noter que, dans certains cas, un coût peut être associe à des états autres que des solutions. Dans cette représentation, les fourmis construisent des solutions en se déplaçant sur un graphe G = (C ; L), ou les nœuds sont les composants de C et où l'ensemble L connecte les composants de C. Les contraintes du problème sont implémentées directement dans les règles de déplacement des fourmis (soit en empêchant les mouvements qui violent les contraintes, soit en pénalisant de telles solutions). Les algorithmes de colonies de fourmis utilisent généralement un trait particulier du comportement des fourmis réelles : le dépôt de piste. Dans l'algorithme , à chaque itération t avec

$$1 < t < tmax$$

, chaque fourmi k (k =1....m) parcourt le graphe et construit un trajet complet de n = |N| étapes (on note|N| le cardinal de l'ensemble N). Pour chaque fourmi, le trajet entre une ville i et une ville j dépend de :

1. la liste des villes déjà visitées, qui définit les mouvements possibles à chaque pas, quand la fourmi k est sur la ville i : J_i^k
2. l'inverse de la distance entre les villes :

$$N_{ij} = 1/d_{ij}$$

, appelée visibilité. Cette information statique est utilisée pour diriger le choix des fourmis vers des villes proches, et éviter les villes trop lointaines ;

3. la quantité de phéromone déposée sur l'arête reliant les deux villes, appelée l'intensité de la piste. Ce paramètre définit l'attractivité d'une partie du trajet global et change à chaque passage d'une fourmi. C'est, en quelque sorte, une mémoire globale du système, qui évolue par apprentissage. La règle de déplacement (appelée règle aléatoire de transition proportionnelle par les auteurs [Bonabeau et al.1999]) estla suivante :

   la Formule 1 :

$$P_i^k j = \begin{cases} \frac{(T_{ij})^\alpha \cdot (N_{ij})^\beta}{\sum_{l \in J_i^k} (T_{il})^\alpha \cdot (N_{ij})^\beta} & \text{si } l \in J_i^k \\ 0 & \text{si } l \notin J_i^k \end{cases}$$

   où $\alpha et \beta$ sont deux paramètres contrôlant l'importance relative de l'intensité de la piste $T_{ij}(t)$, et de la visibilité (nij) . Avec $\alpha = 0$, seule la visibilité de la ville est prise en compte ;la ville la plus proche est donc choisie à chaque pas. Au contraire, avec $\beta = 0$, seules les pistes de phéromone jouent. Pour éviter une sélection trop rapide d'un trajet, un compromis entre ces deux paramètres, jouant sur les comportements de diversification et d'intensification. Après un tour complet, chaque fourmi laisse une certaine quantité de phéromone $\triangle T_{ij}^k(t)$ sur l'ensemble de son parcours, quantité qui dépend de la qualité de la solution trouvée :

   La Formule 2 :

$$\triangle T_i^k j = \begin{cases} \frac{Q}{L^k(t)} & \text{si } (i,j) \in T^k(t) \\ 0 & \text{si } (i,j) \notin T^k(t) \end{cases}$$

   où $T^k(t)$ est le trajet effectué par la fourmi k à l'itération t,$L^k(t)$ la longueur de la tournée et Q un paramètre fixé. L'algorithme ne serait pas complet sans le processus d'évaporation des pistes de phéromone. En effet, pour éviter d'être piégé dans des solutions sous-optimales, il est nécessaire de permettre au système d'oublier les mauvaises solutions. On contrebalance donc l'additivité des pistes par une décroissance constante des valeurs des arêtes à chaque itération. La règle de mise à jour des pistes est donc :

La Formule 3 :

$$T_{ij}(t+1) = (1-\rho) \cdot T_{ij}(t) + \sum_{k=1}^{m} \triangle T_{ij}^{k}(t)$$

où m est le nombre de fourmis et le taux d'évaporation. La quantité initiale de phéromone sur les arêtes est une distribution uniforme d'une petite quantité $T0 \geq 0$. $T_{ij}(t)$ Intensité de phéromone sur l'arrêt (i, j)
au moment t.

• Définitions

– Itération = déplacement de m fourmis dans (t,t+1)

– Cycle = n itérations

– Observation : après un cycle chaque fourmi a parcouru une tournée complète.

– Mise à jour de l'intensité de phéromone :

$$T_{ij}(t+n) = \rho \cdot T_{ij}(t) + \triangle T_{ij}$$

La quantité de phéromone par unité de distance, laissé sur l'arrêt (i,j), par la fourmi k pendant l'intervalle (t,t+n) est définit comme suit :

$$\triangle T_{ij} = \triangle \sum_{k=1}^{m} T_{ij}^{k}$$

où :

– Q : Est une constante.

– L_k :Longueur de la tournée pour la fourmi k.

– ρ : doit être <=1 pour limiter l'accumulation de phéromone.

– T_{ij} :est une constante positive (intensité initiale de phéromone)

– tabou k : vecteur dynamique qui contient la liste des villes visitées par la fourmi k.

– tabou k (s) : la ville de rang s qui se trouve dans la liste de fourmi k.

– d_{ij} :distance entre les villes i et j visibilité :$T_{ij} = 1/d_{ij}$

– α=paramètre qui contrôle l'importance de l'intensité de phéromone.

– β= paramètre qui contrôle l'importance de la visibilité. Donc le choix dépend de distance et de l'intensité de la phéromone.

- Algorithme de base :

  Pour t=1,...,t_{max}

  Pour chaque fourmis k=1,...,m

  Choisir une ville au hasard

  Pour chaque ville non visitée i

  choisir une ville j,dans la liste J_i^k des villes restantes,selon la formule 1

  Fin pour

  Déposer une piste $\triangle T_{ij}^k$ sur le traget $T^k(t)$ confórmement a l'équation 2

  Fin pour

  Evaporer les pistes de selon la formule 3

  Fin pour

5 Les domaines d'application

Il existe plusieurs domaines d'applications dont on cite :

⁕ Applications au problème symétrique et asymétrique de voyageur de commerce.

⁕ Applications au problème d'ordonnancement séquentiel.

⁕ Applications aux problèmes d'affectation quadratique.

⁕ Applications aux problèmes de tournées des véhicules.

⁕ Applications aux problèmes d'établissement d'horaires.

⁕ Applications aux problèmes de coloration de graphe.

⁕ Applications aux problèmes de partitionnement.

⁕ Applications aux réseaux de télécommunications.

6 Conclusion

Lorsqu'on s'attaque à des problèmes réels, il faut se résoudre à un compromis entre la qualité des solutions obtenues et le temps de calcul utilisé. Ces méthodes assurent un compromis entre diversification quand il est possible de déterminer que la recherche se concentre sur de mauvaises zones de l'espace de recherche et intensification on recherche les meilleures solutions dans la région de l'espace de recherche en cours d'analyse. Ces algorithmes ont été appelés « méta-heuristiques » et ont pour objectif de trouver des solutions dont la qualité est au-delà de ce qu'il aurait été possible de réaliser avec une simple heuristique. Les algorithmes de colonies de fourmis révèlent certains avantages par rapport aux autres métaheuristiques dans le cas où le graphe étudié peut changer dynamiquement au cours de l'exécution. La colonie de fourmis s'adaptera de façon relativement flexible aux modifications au cours du temps. De plus, les algorithmes de colonies de fourmis ont été appliqués à un grand nombre de problèmes d'optimisation combinatoire comme le repli de protéine ou le routage de véhicule.

Chapitre 4

Conception et Implémentation

1 Introduction

Dans ce travail nous proposons une architecture et une stratégie pour la sélection des services web par une application cliente . Nous utilisons un modèle de points et de poids pour un ensemble de paramètres de QdS (i.e. Qualité de Service) pour la sélection du meilleur service web. Les paramètres de QdS sont définis selon les besoins des applications clientes. L'administrateur d'une application cliente choisit les paramètres de QdS qui doivent peser le plus au moment de sélectionner le meilleur service web. La stratégie est basée sur qu'au début on choisit le meilleur fournisseur d'un service web. Une application cliente peut substituer un service web en panne par un autre équivalent et disponible. L'objectif de ce chapitre est de concevoir et simuler l'optimisation à base de colonies de fourmis sur le problème de sélection de service web.

2 Présentation de la base

Notre base de données se compose d'un ensemble de dix classes, chaque classe contient quarante opérations : Chaque opération est un tableau qui contient cinq colonnes, chaque valeur représente la somme de QOWS La Latence qui est une valeur aléatoire entre 0 et 100, La Fiabilité qui est une valeur aléatoire entre 0 et

100, La Disponibilité qui est une valeur aléatoire entre 0 et 1, Coût qui est une valeur aléatoire entre 0 et 1, et la Réputation qui est une valeur aléatoire entre 0 et 1), et les lignes représentent les fournisseurs, en tenant compte que chaque service a exactement quatre cent quarante fournisseurs. Notre objectif est de choisir une composition de services qui maximise la disponibilité, la fiabilité la réputation, et qui minimise le cout et le temps d'exécution En plus elle doit aussi satisfaire les contraintes de l'utilisateur si elles existent.

3 Conception

3.1 Diagramme de cas d'utilisation

Un cas d'utilisation est un résumé des scénarios pour un but ou une tâche unique. Un acteur est la personne ou l'objet qui engage les événements impliqués dans cette tâche, Et notre diagramme de cas d'utilisation sera présenté comme suit :

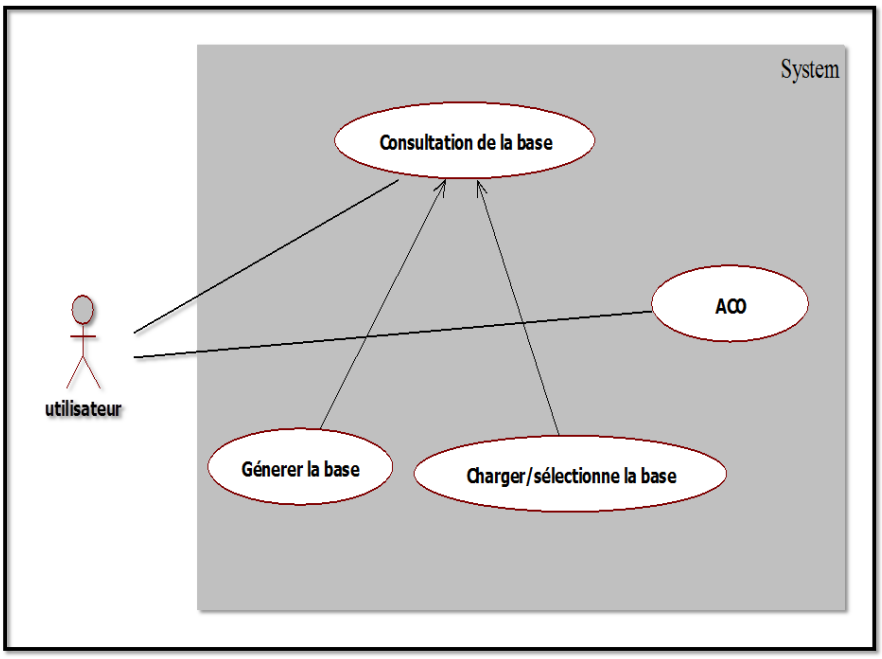

FIGURE 4.1 – Diagramme de Cas d'utilisation

3.2 Diagramme de Classe

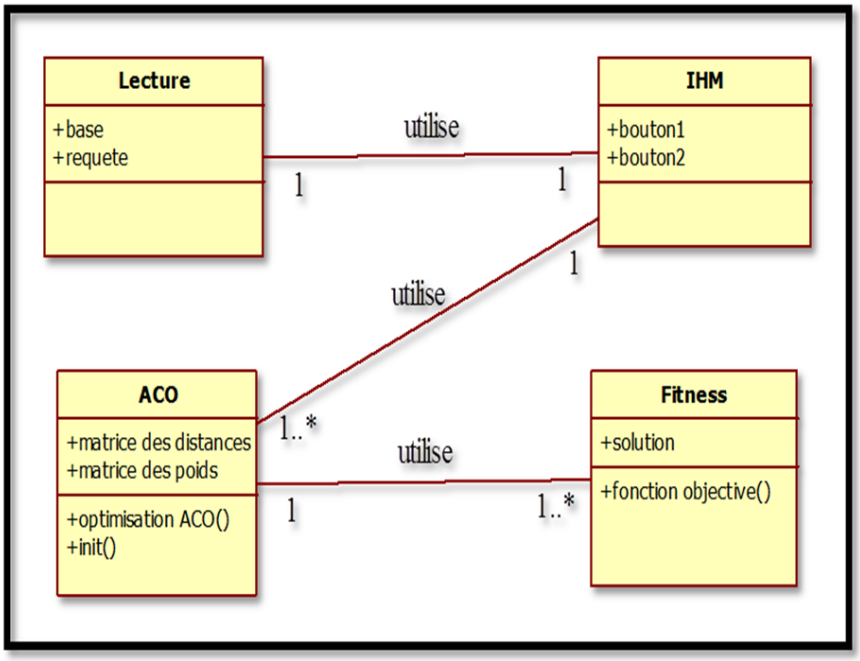

FIGURE 4.2 – Diagramme de Classe

4 Implémentation

4.1 La structure de L'ACO

L'algorithme est conçu de la manière suivante : Les fourmis sont guidées lors de la construction d'une solution par l'information heuristique spécifique au problème qui est inversement proportionnelle à la longueur du parcourt (les fourmis préfèrent le choix des lignes courtes) et le taux de phéromone (expérience des autres fourmis) Chaque fourmi représente une configuration ou bien une structure (arbre) . Alors chaque fourmi construit une solution , une fourmi placée sur un noeud i choisie une destination vers un autre noeud j. Le processus de recherche s'achève quand la fourmi atteint le dernier noeud sans qu'il revienne au noeud de départ en formant une connexion avec l'ensemble des noeuds (arrêt au noeud n-1) toute en formant l'arbre .La mise à jour de la phéromone consiste en deux phases :Mise à jour locale

la quantité de phéromone sur les arêtes (lignes) visitées par l'application des règles de mise à jour. La mise à jour locale est introduite afin d'éviter la convergence prématurée et réduite la quantité de phéromone sur l'arête reliant un noeud i avec un autre noeud j (la ligne ij) de manière à décourager la fourmi suivante de choisir la même destination durant le même cycle.

Voici des captures d'écran de notre application :

FIGURE 4.3 – Les Contraintes de QOS

FIGURE 4.4 – Chargement/Génération de la base

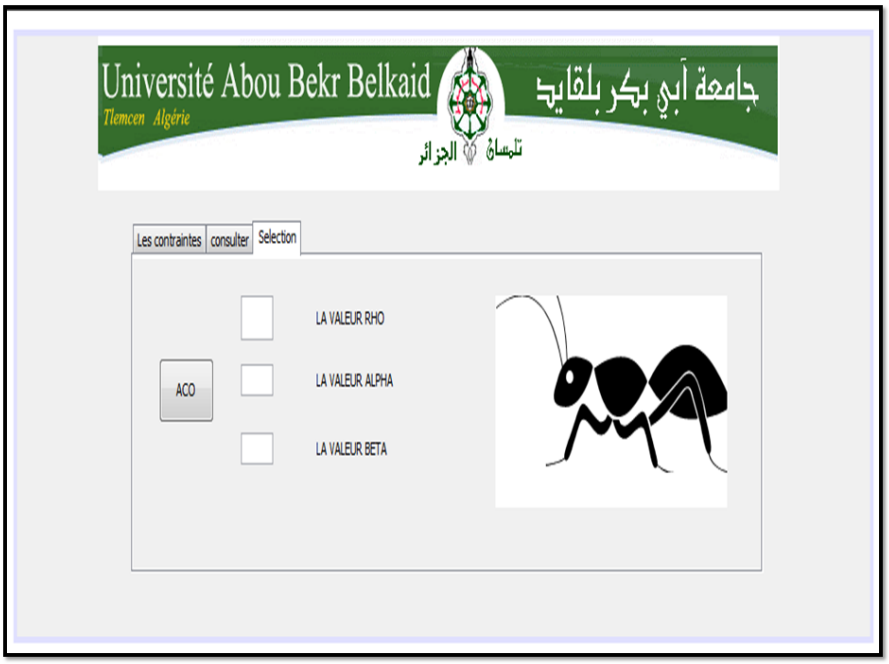

FIGURE 4.5 – La Sélection

et Mise à jour globale.Pendant la construction d'une solution, la fourmi modifie

4.2 Le chargement

Tout d'abord nous avons commencé par charger les données de la BDD , qui représentent les distances, en créant une classe connexion sachant que nous avons configuré les fichiers texte. Le bouton la base dans mon application affiche les lignes de la BDD , le bouton générer affiche la somme des cinq contraints ,enfin le bouton lecture affiche notre BDD finale qui contient tous les 40 lignes par 10 classe, et une onzième classe pour la résolution de notre probleme.

4.3 La sélection

La sélection se fait par ACO qui travaille avec la technique de voisinage ,avec α etβ qui sont deux paramètres qui contrôlent l'importance relative entre phéromones et visibilité. et rho qui est une valeur<=1 pour la mise ajour des phéromones.a la fin de traitement ACO nous donne le pourcentage entre le cout initiale et le cout finale qui est calculé par la formule suivante :

$$pourc = \frac{CoutSolOpt - \mid CoutSolOpt - CoutSolCourante \mid}{CoutSolOp} * 100$$

- par exemple si on comme Cout(sol*)=35 et on obtient une solution egale a 40 alors le calcule se fait comme suit :

$$pourc = \frac{35 - \mid 35 - 40 \mid}{35} * 100$$

$$pourc = 87\%$$

5 Expérimentation

Nous avons mené une expérience pour évaluer la performance de l'approche proposée. D'après le scenario qu'on a proposé , Le but est de démontrer comment notre approche peut aider le patient X a sélectionner la meilleure offre. Nous courons notre expérience sous NetBeans IDE 7.0.1 de Sun Microsystems sous le système d'exploitation Windows7, Processeur $IntelPentiumCPUP6200@2.13GHz, 3GigadeRAM$. nous avons 10 classes de services, et chaque classe contient 40 cas, le nombre total de solutions = 440 car on a ajouté une classe de plus pour éviter tous problème. Le nombre d'attributs QOS est fixé à 5. En chargeant notre base et on obtient les résultats suivants :

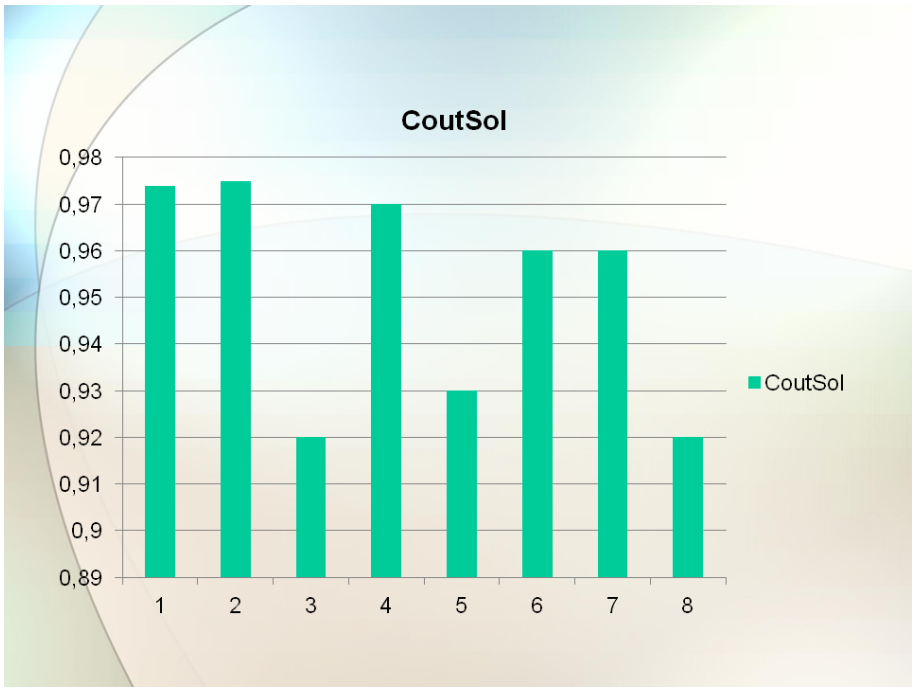

FIGURE 4.6 – Performance de l'ACO sur la base des services

Avec $\alpha = 0.5, \beta = 0.5, \rho = 0.01, Q = 2$ Plusieurs simulations ont été faites avec des configurations différentes cette figure montre les taux d'optimalité lorsque nous négligeons les contraintes globales.

6 Conclusion

Dans ce chapitre nous avons proposé un exemple d'application pour illustrer notre travail. Nous avons présenté les résultats de la validation de notre approche, afin d'évaluer l'efficacité de cette dernière. D'après les résultats qu'on a obtenus, on a confirmé l'efficacité de l'algorithme de colonies de fourmis dans le domaine de sélection de services.

Conclusion

Dans ce mémoire, nous avons abordé le sujet de l'optimisation . Après avoir exposé les méthodes d'optimisation statiques existant dans la littérature, nous avons choisissez la méthode de colonie de fourmis. Les algorithmes à colonies de fourmis fournissent de très bons résultats pour les problèmes de d'optimisations avec plusieurs centaines de nœuds, mais ne concurrencent pas les meilleures méthodes publiées dans le domaine.

Perspectives :

Comme perspective à ce travail nous proposant d'utiliser d'autres méthodes qui aide a la sélection de service web dont on site :

•la programmation par contraintes (ppc) qui permettent de d'écrire des problèmes de façon déclarative, en termes de contraintes, la résolution de ces problèmes étant prise en charge par des algorithmes de résolution intégrés au langage, sans que l'utilisateur n'ait besoin de « programmer » cette résolution.

•SAT Solver : Le problème SAT est un problème de décision qui consiste à déterminer si une formule booléenne mise sous forme normale conjonctive admet ou non une évaluation qui la rend vraie.

• Les algorithmes génétiques dont on site Vector Evaluated Genetic Algorithm(VEGA), Multiple Objective Genetic Algorithm(MOGA),Niched Pareto Génétic Algorithm(NPGA).

Bibliographie

Résumé

L'approche par service est devenue la plus utilisée dans le monde du developpement software. Ceci à impliqué un nombre important de service web dans le monde. dans ce travail nous nous intéressons à la sélection du service web pour cela nous avons opté une optimisation mono-objective et en particulier des algorthmes de colonies de fourmis. Les expérimenttations ont montré l'efficacité de l'ACO dans le domaine de sélection de service.

mots clés : SOA, service web, sélection de service, ACO, méta-heurestiques.

Abstract

The web service has become the most used in the world of software development. This involved to a significant number of web service. In this work we focus on selecting the web service that we opted for a single-objective optimization and especially ant colonies algorithm. The expérimenttations showed the effectiveness of the ACO in service selection. Keys Words : SOA, web service, service selection, ACO, meta-heurestiques. **keywords :** SOA,Web Service,selection of service,ACO,metaheuristics.