REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

Université Abou Bekr Belkaid
Tlemcen Algérie

جامعة أبي بكر بلقايد
تلمسان الجزائر

Faculté des Sciences – Tidjani HADDAM
Département de l'Informatique

Mémoire

Pour l'obtention du diplôme de

MASTER

Spécialité :
Informatique

Option :
Modèles Intélligents et Décision (MID)

THÈME

Sélection de sérvices Web à base de QoWS

Présenté par :
Mr : DALI YAHIA Mohammed

Membres du jury

Mr	**: LAHSAINI Mohammed**	**Président**
Mr	**: HADJILA Fethallah**	**Encadreur**
Mr	**: BELABED Amine**	**Examinateur**
Mr	**: BENAMAR Abdelkrim**	**Examinateur**
Mr	**: BENAZZOUZ Mourtada**	**Examinateur**
Mme	**: ILES Nawel**	**Examinateur**
Mr	**: SMAHI Ismail**	**Examinateur**

2010/2011

REMERCIEMENTS

Louange à Dieu qui ma donné la force, le courage, et l'espoir nécessaire Pour accomplir ce travail et surmonter l'ensemble des difficultés.

J'exprime ma gratitude, mes remercîements à mes parents qui ont fait de leur mieux pour m'aider.

Je tiens à remercier vivement :

- **Mon encadreur Mr. HADJILA Fethallah pour ces conseils et son suivi durant la réalisation de mon projet.**
- **A Mrs. M.LAHSAINI, A.BELABED, A.BENAMAR, M.BENAZZOUZ, I.SMAHI, et Mme. N.ILES qui ont bien voulu faire partie du jury.**

Je remercie aussi les personnes qui m'ont aidé et encouragé le long de ce travail.

DEDICACE

Je dédie ce travail à:

Mes parents

Vous vous êtes dépensés 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 mon frère et mes sœurs.

A mes oncles, tantes, cousins et cousines.

Spécial dédicace à mon colégue et ami AMARA Mohammed qui ma bien aidé le long de mon parcours universitaire.

A tous les members de ma promotion.

A mes amis.

A tous mes proffésseurs.

Table de matières

Liste de figures

Liste des tableaux

Introduction générale

Contexte

Le Web a évolué pour englober diverses sources d'information accessible mondialement. Les organisations de tous les spectres ont déjà déplacé leurs opérations principales au Web, ce qui a entraîné une croissance rapide des différentes applications Web. Les services Web sont devenus le facteur le plus significatif technologique par produit.

Le problème de la recherche dans les services Web a attiré l'attention des chercheurs au cours de la dernière décennie. La raison à cela est que la technologie évolue et que beaucoup de services sont disponibles, il devient important d'être en mesure de localiser les services qui répondent à nos besoins dans un grand nuage dense d'offres. Plusieurs propositions ont été avancées pour résoudre ce problème et plusieurs normes ont été définies, mais aucun d'entre eux a été efficace ou est maintenant acceptée comme le moyen d'effectuer une recherche de service.

Ce mémoire se penche sur une perspective plus large, une solution concrète et applicable

Problematique

La capacité d'accéder efficacement aux services Web est nécessaire. En outre, comme les services Web avec des fonctionnalités similaires sont censés être fournis par des fournisseurs concurrents, le défi majeur est de concevoir des stratégies d'optimisation pour trouver les "meilleurs"services Web ou la composition de celle-ci est à l'égard de l'utilisateur qui s'attend fourni de qualité.

Contribution

Dans ce mémoire nous proposons une approche qui utilise les algorithmes génétiques qui sont une abstraction de la théorie de l'évolution. L'utilisation de la théorie de l'évolution comme modèle informatique pour trouver une solution optimale peut se justifier par le fait que la théorie de l'évolution permet de rechercher la solution parmi un très grand nombre de possibilités dans un laps temps raisonnable (le processus de l'évolution se déroule en parallèle).

Plan du mémoire

Le mémoire est composé de trois chapitres et une conclusion générale, qui sont organisés comme suit:

Chapitre I

Il présente la technologie des services Web et les principaux standards qu'elle supporte, parmi les quels on cite les protocoles soap, wsdl, uddi, il montre aussi leurs avantages et leurs inconvenient.

Chapitre II

Ce chapitre expose les concepts fondamentaux des algorithmes génétiques. Après un historique très riche, et une terminologie pour définir le vocabulaire employé par notre travail; ce chapitre explique le fonctionnement de l'approche et les différentes méthodes de sélection

Chapitre III

Ce chapitre explique une application à partir du domaine de courtage de la voiture pour illustrer notre travail avec l'approche proposée. La première partie du chapitre énonce le sénario et mentionne les paramètres, la fonction et les données initiaux de notre application, et la seconde partie explique la conception de notre systeme et l'implementation pour évaluer la performance de l'approche proposée

Conclusion générale

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

Chapitre I

Les services Web

1. Introduction

Depuis les années 70, nous remarquons un besoin de communication entre les systèmes d'information des entreprises (fournisseurs ou clients). Ce type de communication est appelé intégration B2B. Dans un sens technologique, l'intégration B2B se réfère à « la technologie de logiciel utilisée pour connecter n'importe quel système d'information d'une entreprise à tous ses associés de commerces » [1]. Les protocoles assurant ces besoins s'appellent protocoles B2B. Des exemples de ces protocoles sont EDI [8] et RossetaNet [9].

Avec l'essor du Web qui a eu dans les dernières années, il a surgi le besoin de permettre qu'une application cliente invoque un service d'une application serveur en utilisant l'Internet. Ce besoin a été l'origine de ce qui se connaît comme services Web. En tenant en compte que les services Web permettant de connecter des applications différentes, l'utilité de cette technologie devient évidente et importante. Pour cette raison les activités de recherche et développement autour du sujet services Web ont un dynamisme très haut. Dans ce chapitre, nous décrivons les technologies liées aux services Web.

2. Définition et description de service Web

Le consortium W3C (http://www.w3.org/2002/ws/) définit un service Web comme étant une application ou un composant logiciel qui vérifie les propriétés suivantes :

- Il est identifié par un URI.

- Ses interfaces et ses liens (binding) peuvent être décrits en XML.

- Sa définition peut être découverte par d'autres services Web.

- Il peut interagir directement avec d'autres services Web à travers le langage XML et en utilisant les protocoles d'Internet.

La définition exacte du terme "service Web" est encore fortement discutée. Les différentes définitions proposées s'accordent au moins sur l'idée qu'un service Web est un nouveau type de composant logiciel ayant la capacité de publier ses fonctions sur Internet sous forme de services, de rendre ces services facilement invocables et de les mettre à disposition par l'intermédiaire de protocoles Internet standardisés (HTTP, XML).

3. L’intérêt des services Web

Les services Web comportent de nombreux avantages, ils sont utilisables à distance via n'importe quel type de plate-forme, ils peuvent servir au développement d’applications distribuées et sont accessibles depuis n’importe quel type de clients. Les services Web appartiennent à des applications capables de collaborer entre elles de manière transparente pour l’utilisateur.

4. Les application des services Web

Les technologies des services Web peuvent être appliquées à toutes sortes d’applications auxquelles elles offrent des avantages considérables en comparaison aux anciennes API propriétaires, aux implémentations spécifiques à une plate-forme et à quelques autres restrictions classiques que l’on peut rencontrer (multi-plateforme, multi-langage, disponible sur Internet avec une information actualisée disponible en temps réel, ...).

L’application des services Web est multiple, autant dans les domaines du **B2C, B2B** que pour des domaines de gestion, par exemple gestion de stock, gestion commerciale, etc...

B2C (Business to Consumer) : Qualifie une application, un site Internet destiné au grand public.

B2B (Business to Business) : Qualifie une application, un site Internet destiné au commerce de professionnel à professionnel.

Les services Web peuvent être utiles dans la plupart des scénarios applicatifs lorsque la communication peut être établie sur un modèle bidirectionnel (requête/réponse). C’est néanmoins loin d'être aussi limitatif, beaucoup d’autres modèles peuvent avoir recours aux services Web, sans même que vous vous en rendiez compte. Les entreprises qui mettent à disposition leurs services Web permettent aux développeurs intéressés par ses fonctionnalités de les réutiliser sans avoir à les recoder.Le principe des services Web permet d’avoir un partage des fonctionnalités et facilite grandement le développement. [4]

5. L’architecture générale des services Web

Jusqu'ici, l'accès via Internet à une ressource applicative ou à une base de données s'effectuait par l'envoi d'une requête s'appuyant sur des langages tels que (PHP, JSP, …).

Il s'agissait donc d'un dialogue entre une couche de présentation reposant sur HTML (protocole http) et des applications installées sur un serveur distant.

Avec les services Web, un dialogue est désormais instauré entre applications qui peuvent être installées sur des machines distantes, et ceci grâce à des standards XML. En effet, afin de dialoguer via Internet, ces applications doivent « parler » le même langage, langage basé sur le XML.

Dans la figure (I.1), l’architecture de référence des services Web se base sur les trois concepts suivants :

- Le fournisseur de service : c’est le propriétaire du service.
- Le client (ou le consommateur de service) : c’est un demandeur de service. D’un point de vue technique, il est constitué par l’application qui va rechercher et invoquer un service.
- L’annuaire des services : c’est un registre de descriptions de services offrant des facilités de publication de services à l’intention des fournisseurs ainsi que des facilités de recherche de services à l’intention des clients.

Les interactions de base qui existent entre ces trois éléments sont les opérations de publication, de recherche, d’invocation et de lien (bind).

Pour bien comprendre le fonctionnement de cette architecture, nous expliquons ci-dessous 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. [7]

Figure I.1. Structure des services Web

6. Les principales technologies de développement de service Web

Une caractéristique qui a permis un grand succès de la technologie des services Web est qu'elle est construite sur des technologies standards de l'industrie. Dans cette section, nous allons définir ces technologies.

6.1. XML – eXtensible Markup Language

6.1.1. Définition

XML est un standard promulgué par le W3C, l'organisme chargé de standardiser les évolutions du Web. On retrouve dans XML une généralisation des idées contenues dans HTML, on n'utilise les balises que pour décrire l'aspect graphique que doit revêtir la page dans le navigateur Web. Dans XML, les balises permettent d'associer toute sorte d'informations au fil du texte. [2]

XML a été conçu pour des documents arbitrairement complexes, tout en s'appuyant sur cinq grands principes simples et clairs :

- Lisibilité à la fois par les machines et par les utilisateurs ;
- Définition sans ambiguïté du contenu d'un document ;
- Définition sans ambiguïté de la structure d'un document ;
- Séparation entre documents et relation entre document ;
- Séparation entre structure du document et présentation du document.

Le contenu d'un document XML est décrit par une succession d' «éléments», blocs de texte encadré par des paires de balises ouvrantes et fermante, qui sont les « unités de contenue». Ces éléments sont liée entre eux par une hiérarchique, certain éléments apparaissant imbriqués dans d'autres. XML et DTD permettent aussi la séparation effective du contenu de la structure de document.

6.1.2. XML Namespaces

XML Namespaces est une recommandation du W3C qui été rapidement adopté après XML0.1, visant à résoudre le problème de l'ambiguïté éventuelle des balises dans un document XML. [3]

Les namespaces permettent ainsi de résoudre le problème des différences éventuelles d'interprétation du même document XML par des applications différentes. En s'appuyant sur le dispositif des URI, qui en assure l'unicité, et au prix d'une écriture un peu plus «bavarde», les balises et les attributs XML sont alors dotés d'une interprétation spécifique, nom ambiguë.

6.1.3. XML schéma

La recommandation XML schéma, adoptée après de longues discussions dans les comités technique du W3C, représente un réel tour de force et une innovation dans l'utilisation de XML, rompant tout net avec son usage original de la publication des documents. XML schéma précise comment représenter en XML la structure des données en générale – ce qu'on a l'habitude d'appeler les métadonnées dans le monde des bases de données relationnelles (description des tables, des colonnes, de leurs types etc. …). [3]

XML schéma est adapté à la description des structures de données des documents XML, des bases de données relationnelles, mais à celle des modèles objet des langages de programmation orientés objet comme JAVA et C++.

Un schéma XML définit, d'une part, l'imbrication des éléments entre eux, ce qui s'apparente aux DTD, et d'autre part, le type des éléments et de leurs attributs. L'information fournie par le schéma est donc plus riche que celle dans le DTD.

La spécification définit deux namespaces pour XML schéma :

- xsd ou xs associé à l'URL [14] ;
- xsi associé à l'URL [15] ;

Ce deux namespaces sont utilisés dans les documents XML liés à des schémas.

Nous avons défini jusqu'à maintenant les éléments importants de XML qui sont utilisables dans le service Web. Nous avons abordé la notion des namespace et la notion de XML schéma car la définition de la technologie suivante demande la connaissance des ces notions.

6.2. Le protocol SOAP (simple Object Access Protocol)

6.2.1. Présentation

SOAP [10] est un protocole adopté par le Consortium W3C. Le Consortium W3C crée des standards pour le Web : son but est donc de créer des standards pour favoriser l'échange d'information.

SOAP veut dire Simple Object Access Protocol. Si l'on voulait traduire cette définition en français cela donnerait Protocole Simple d'Accès aux Objets. En effet, le protocole SOAP consiste à faire circuler du XML généralement, via du http sur le port 80 (ou en utilisant un autre protocole). Cela facilite grandement les communications, car le XML est un langage standard et le port utilisé est le port 80, qui ne pose donc pas de problème pour les firewalls (pare-feu) de l'entreprise, contrairement à d'autres protocoles.

Tout comme la technologie des services Web, le protocole SAOP est très jeune. Le protocole SOAP a été crée en septembre 98, avec la version 0.9, par trois grand entreprise :

Microsoft, UserLand et DevelopMentor. IBM n'a participé au protocole SAOP qu'à partir de la version1.1 en Avril 2000. C'est cette même année que SOAP a été soumis au W3C. Depuis septembre 2000, SAOP 1.1 est en refonte complète pour donner jour à la version 1.2 avec un groupe de travail de plus de 40 entreprises, on trouve bien sur Microsoft, IBM mais aussi HP, Sun, INTEL… . (www.w3.org/tr/soap)

6.2.2. Ses caractéristiques

- SOAP permet une normalisation des échanges de données. Les données sont encodées en XML et échangées par des appels de procédures à distance [11] en utilisant HTTP/SMPT/POP comme protocole de communication.
- Simple, extensible et permet le diagnostic des erreurs.
- Message unidirectionnel (Requête→ Réponse).
- Fonctionne de manière synchrone et asynchrone.
- Indépendant de la plate-forme et du langage.
- N'est pas perturbé par un pare-feu.

6.2.3. Fonctionnement

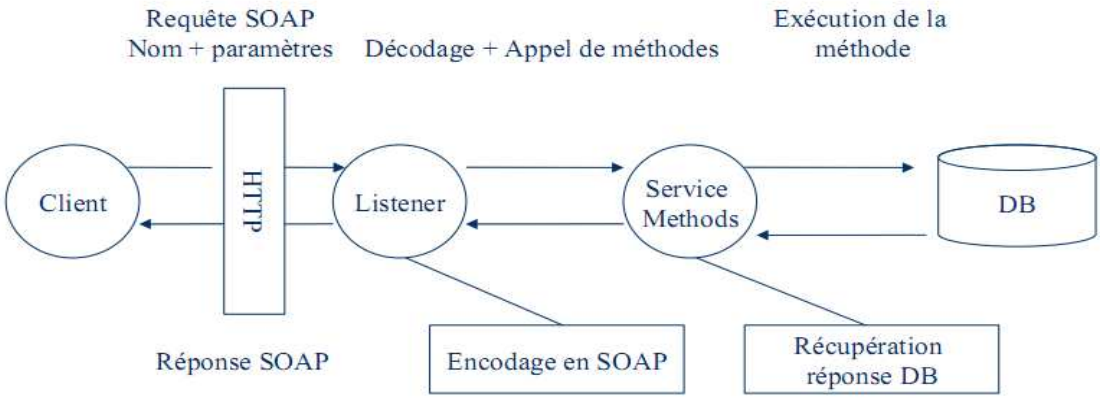

Figure I.2 Traitement d'un message SOAP

La requête SOAP transmise par le client à l'aide des protocole de transmissions (exemple HTTP) passe pare un Listener() que le permet de adresser (cas de java) aux bibliothèques, ces bibliothèque donnée au client l'accède aux méthode des services. Si l'exécution des

méthodes a besoin a des informations de la base de donné alors il doit connecter avec la base de donnée.

6.2.4. Modèle d'échange de messages en SOAP

Les messages SOAP sont des transmissions fondamentalement à sens unique d'un expéditeur à un récepteur comme le montre la **Figure I.3**, lorsqu'une transmission d'un message commence (e.g. invocation d'un service Web), un message SOAP (ou document XML) est génère. Ce message est envoyé à partir d'une entité appelée le SOAP sender, localisé dans un SOAP nœud. Le message est transmis à zéro ou plusieurs nœuds intermédiaires (SOAP intermediates) et le processus finit lorsque le message arrive au SOAP receiver.

Le chemin suivi par un message SOAP est nomme message path. les éléments qui participent au processus.

Quand un message SOAP arrive dans une entité SOAP, il doit suivre le processus décrit ci-dessous :

1. Identifier toutes les parties du message SOAP destiné à cette entité
2. Vérifiez que toutes les parties obligatoires identifiées dans le pas 1 sont soutenues par l'entité et traitez-les en conséquence. Si ce n'est pas le cas rejeter le message
3. Si l'entité n'est pas la destination suprême du message enlève alors toutes les parties identifiées dans le pas 1 avant l'expédition du message

Le traitement d'un message ou une partie d'un message exige que le processeur de SOAP comprenne, parmi d'autres choses, le modèle d'échange étant utilisé (e.g. sens unique, multicast), le rôle du destinataire dans ce modèle, l'emploi de mécanismes RPC, la représentation ou le codage de données, aussi bien que d'autre sémantique nécessaire pour traitement correct

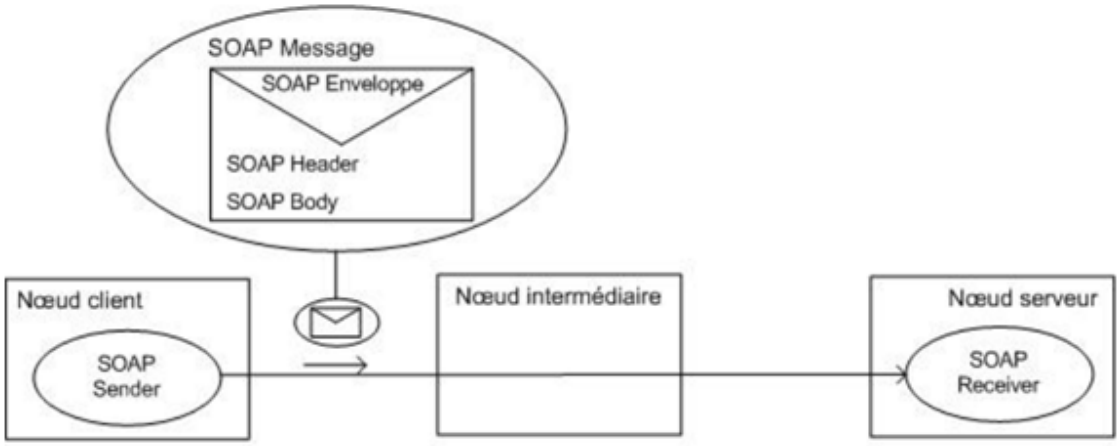

Figure I.3 Chemin emprunté par un message SOAP

6.2.5. Structure d'un message SOAP

Tout d'abord un message SOAP est un document XML qui doit avoir la forme suivante : [11]

- La structure de l'enveloppe SOAP qui définit une structure générale décrivant le contenu, le destinataire, et la nature du message
- Les règles d'encodage SOAP qui définissent le mécanisme de sérialisation utilisé pour échanger des objets
- SOAP RPC qui définit, pour les utilisations synchrones, une convention de représentation des appels et des réponses des procédures distantes

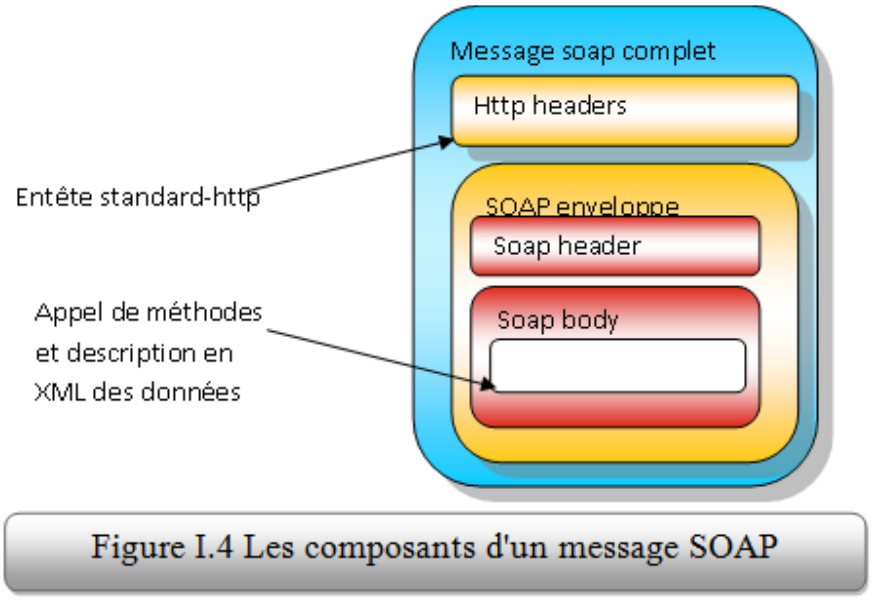

Figure I.4 Les composants d'un message SOAP

Un message SOAP est composé des parties suivantes :

6.2.5.1. Le HTTP Header

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

6.2.5.2. L'enveloppe SOAP

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

6.2.5.3. Le header SOAP

Cette partie du message est optionnelle. Elle sert à transmettre des informations nécessaires pour l'exécution de la requête SOAP aux dictionnaires qui recevront le message.
On y précise généralement des informations liées aux transactions, à l'authentification, etc.
Le header est composé d'un ou plusieurs champs : l'attribut actor, désignant le destinataire du header, l'attribut mustUnderstand, qui indique si le processus est optionnel.

L'attribut actor permet de préciser l'application à laquelle est destinée l'information contenue dans le header. L'URI http://schemas.xmlsoap.org/soap/actor/next précise en particulier que ces informations sont destinées à la première application qui reçoit le message. Dans le cas où l'actor n'est pas précisé, le header est analysé par le destinataire final du message.

Dans l'exemple suivant, on précise des informations sur l'identification de l'utilisateur et la transaction à laquelle appartient le message :

```
<SOAP-ENV :Header
SOAP-ENV :actor="http://schemas.xmlsoap.org/soap/actor/next"
SOAP-ENV:mustUnderstand="1">
<identifiant numero="124527"/>
<transaction type="compensees" numero="YU75X"/>
</SOAP-ENV:Header>
```

6.2.5.4. Le Body SOAP

Le body SOAP contient toutes les informations que l'on veut transmettre à l'application distante. Le contenu du Body est normalisé dans SOAP RPC, pour modéliser une requête et sa réponse. Le body de la requête contient l'identifiant de l'objet distant, le nom de la méthode à exécuter et les éventuels paramètres. Le body de la réponse contient le résultat de l'exécution de la requête.

6.2.6. SOAP RPC

SOAP RPC définit les conventions permettant d'utiliser SOAP comme un RPC, et le format des messages pour effectuer une requête ou envoyer une réponse. SOAP RPC se base sur les spécifications de XML-RPC. La figure I.5 donne une vue global sur SOAP RPC :

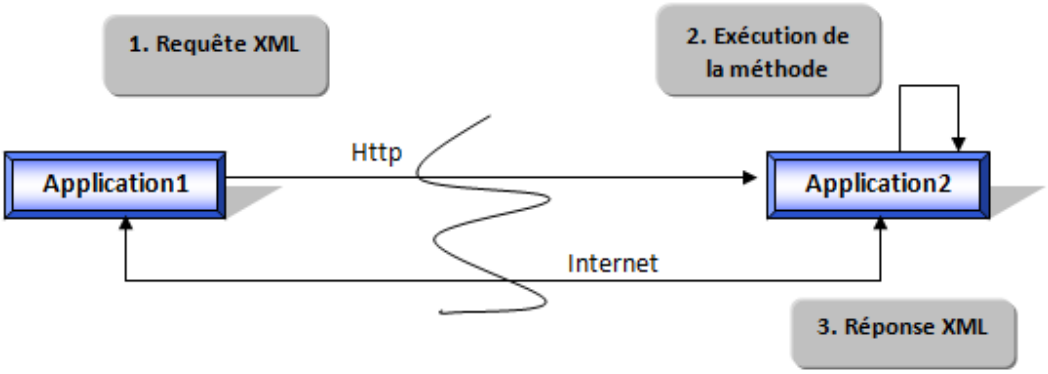

Figure I.5 SOAP RPC

6.2.6.1. Préambule

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

En Java toute classe peut être rendue accessible via SOAP (en utilisant Apache SOAP par exemple).

Lors de l'appel d'une méthode sur le service déployé, c'est la méthode correspondante de la classe Java qui est exécutée.

Nous donnons un exemple de classe java : « **Calculatrice.java** » que l'on désire rendre accessible via SOAP avec l'identifiant "urn: Calculatrice " :

```
public class Calculatrice {
public int plus(int a,int b) {
return a+b;         }
public int moins(int a,int b) {
return a-b;         }
}
```

6.2.6.2. Requêtes / réponses SOAP

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

```
<SOAP-ENV:Body>
<m:plus xmlns:m="urn:Calculatrice">
<a type="xsd:int">5</a>
<b type="xsd:int">8</b>
</m:plus>
</SOAP-ENV:Body>
```

Détails de la requête

- <m:plus xmlns :m="urn: Calculatrice">: l'espace de nommage défini, a pour URN l'identifiant du service (« urn : Calculatrice »). Le nom de l'élément (plus) correspond au nom de la méthode à exécuter.

- Les paramètres d'appel de la méthode sont ensuite ajoutés les uns à la suite des autres. Ici on n'a deux paramètres a et b : de type Int. On a affecté à ces deux paramètres les valeurs 5 et 8 respectivement.

Le message SOAP est alors envoyé au service SOAP. Le service exécute la méthode précisée dans la requête, ici c'est la méthode « plus », et retourne ensuite un message SOAP dont le Body contient le résultat de l'opération :

```
<SOAP-ENV:Body>
<m:plus xmlns:m="urn:Calculatrice">
<return type="int">13</return>
</m:plus>
</SOAP-ENV:Body>
```

Le résultat est contenu dans un élément <return> :

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

6.2.6.3. La gestion des erreurs

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

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

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

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

```
<SOAP-ENV:Body>
 <SOAP-ENV:Fault>
  <faultcode>SOAP-ENV:Client</faultcode>
  <faultstring>
   Method signature does not match.
  </faultstring>
 </SOAP-ENV:Fault>
</SOAP-ENV:Body>
```

6.2.6.4. Pattern d'utilisation

La norme SOAP ne définit pas les patterns d'utilisation. Ceci dit, à l'image des RPC traditionnels, on pourra être amené à utiliser un objet distant par le biais d'un proxy effectuant les requêtes SOAP (figure I.6), ce qui permet de manipuler l'objet distant comme un objet local :

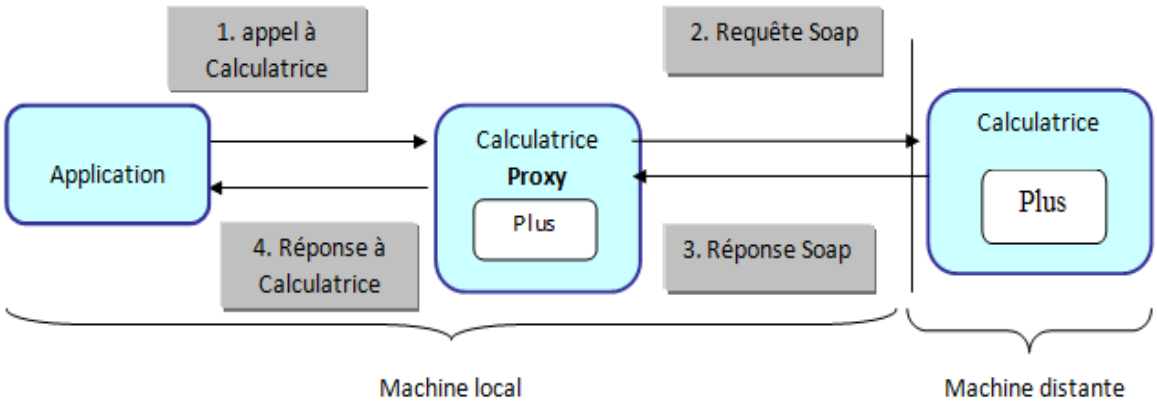

Figure I.6 Proxy SOAP

1. Une application locale veut appeler la méthode plus de l'objet distant Calculatrice : elle instancie un proxy et appelle sa méthode plus ;

2. Le proxy est un « raccourci » vers l'objet distant : il effectue la requête SOAP destinée à l'objet distant.

3. L'objet distant décode le message XML reçu et exécute sa méthode plus. Il envoie sa réponse au proxy.

4. Le proxy transmet la réponse reçue à l'application.

L’avantage de cette méthode est que l’application n’a pas à se soucier de la localisation de l’objet distant et de la construction de la requête SOAP : elle l’utilise comme un objet local. [10]

6.3. La technologie WSDL (Web Service Description Language)

6.3.1. Définitions générale

Le WSDL [3] est un langage dérivé d'XML permettant de fournir les spécifications nécessaires à l’utilisation d’un service Web en décrivant les méthodes, les paramètres et ce qu'il retourne.

Le WSDL est aussi l’équivalent de IDL (Interface Definition Language) pour la programmation distribuée (CORBA). Ce langage permet de décrire de façon précise les services Web, en incluant des détails tels que les protocoles, les serveurs, les ports utilisés, les opérations pouvant être effectuées, et les formats des messages d’entrée et de sortie.

Il y eut d’autres tentatives de langages pour résoudre le problème de la définition des services Web. Microsoft a d’abord proposé SDL (Service Definition Language) avec une implémentation fournie dans leur Toolkit SOAP, puis IBM a proposé NASSL (Network Accessible Service Specification Language), dont une implémentation est fournie dans SOAP4J. Microsoft modifia sa première spécification et proposa SCL (SOAP Contract Language), puis les différents acteurs s’accordèrent sur WSDL [11].

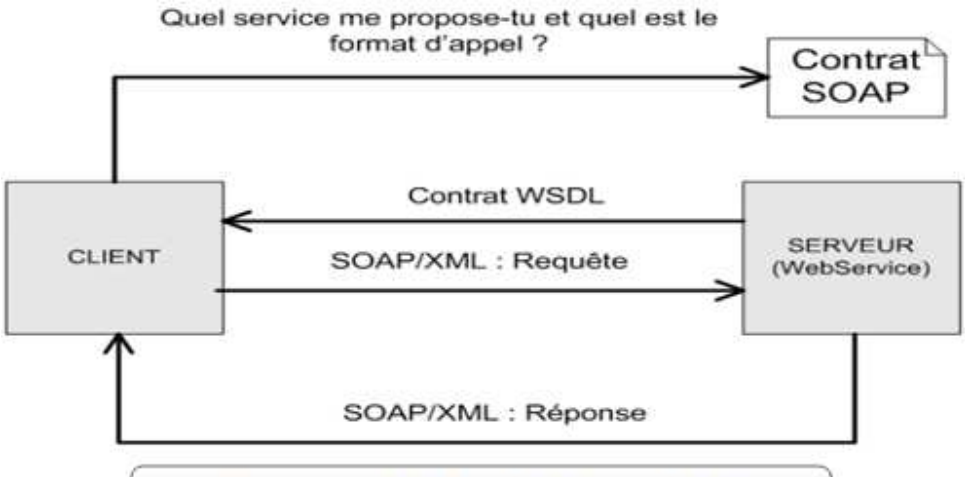

Figure I.7 Rôle générale de WSDL

6.3.2. Structure d’un document WSDL

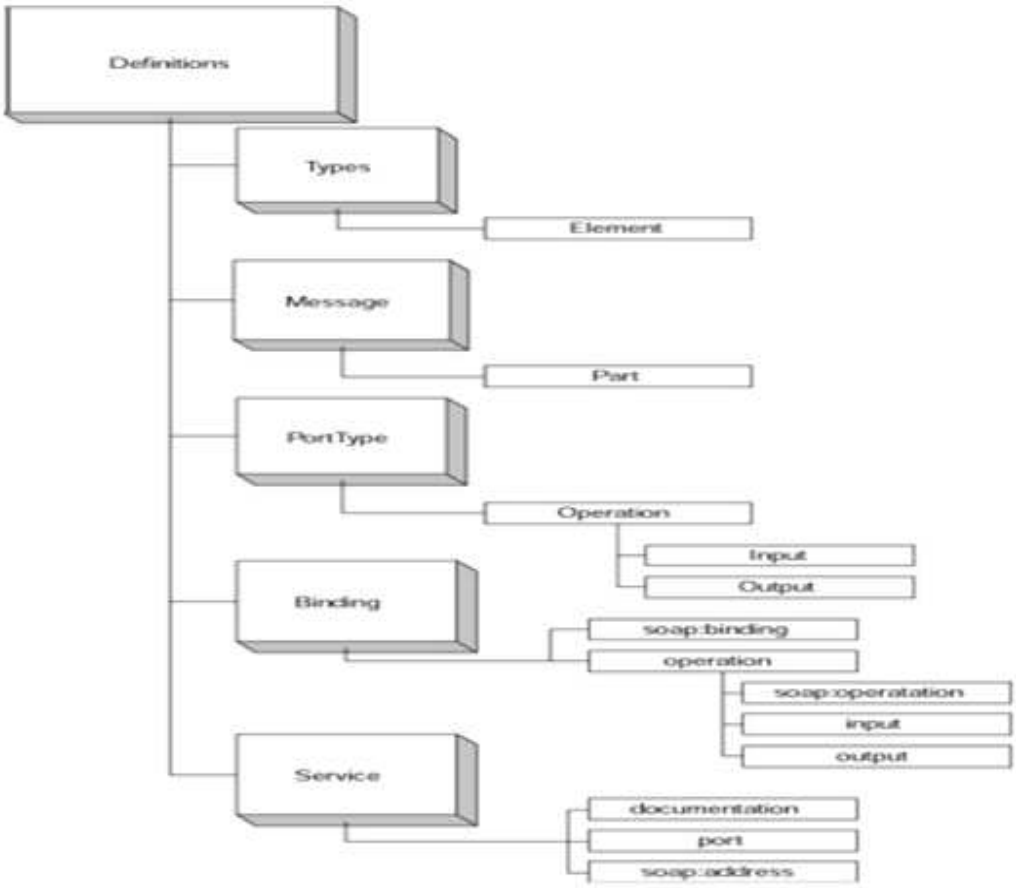

Figure I.8. La structure d'un document

La figure I.8 défini la structure générale d'un document WSDL.

- <**definitions**> : Cet élément contient la définition du service. C'est la racine de tout document WSDL. Cette balise peut contenir les attributs précisant le nom du service, et les espaces de nommage. <definitions> contient trois types d'éléments :
- <**message**> et <**portType**> : Ces éléments définissent les opérations offertes par le service, leurs paramètres d'entrée et de sortie, etc. En particulier, un <message> correspond à un paramètre d'entrée ou de sortie d'une <operation>. Un <portType> définit un ensemble d'opérations. Une <**operation**> définit un couple message -entrée / message -sortie. Par exemple, dans le monde Java, une opération est une méthode et un portType une interface.
- <**binding**> : Cet élément associe les <portType> à un protocole particulier. Les bindings possibles sont SOAP, CORBA ou DCOM. Actuellement seul SOAP est utilisé. Il est possible de définir un binding pour chaque protocole supporté.

<service> : Cet élément précise les informations complémentaires nécessaires pour invoquer le service, et en particulier l'URI du destinataire. Un <service> est modélisé comme une collection de ports, un <port> étant l'association d'un <binding> à un URI.
Il est aussi possible de définir des types de données complexes dans une balise <types> juste avant la balise <message>. Enfin, chaque élément WSDL peut être documenté à l'aide de l'élément <documentation>. Cet élément contient des informations liées à la compréhension du document par les utilisateurs humains du service.

L'exemple suivant montre un document XML qui décrit le service Calculatrice défini précédemment :

```
<?xml version="1.0" encoding="UTF-8" ?>
- <wsdl:definitions targetNamespace="http://localhost:8080/axis/Calculatrice.jws"
xmlns:apachesoap="http://xml.apache.org/xml-soap"
xmlns:impl="http://localhost:8080/axis/Calculatrice.jws"
xmlns:intf="http://localhost:8080/axis/Calculatrice.jws"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:wsdlsoap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">


- <wsdl:message name="moinsRequest">
<wsdl:part name="a" type="xsd:int" />
  <wsdl:part name="b" type="xsd:int" />
</wsdl:message>
- <wsdl:message name="plusRequest">
<wsdl:part name="a" type="xsd:int" />
  <wsdl:part name="b" type="xsd:int" />
  </wsdl:message>
- <wsdl:message name="moinsResponse">
<wsdl:part name="moinsReturn" type="xsd:int" />
  </wsdl:message>
- <wsdl:message name="plusResponse">
<wsdl:part name="plusReturn" type="xsd:int" />
  </wsdl:message>
- <wsdl:portType name="Calculatrice">
- <wsdl:operation name="plus" parameterOrder="a b">
```

```
26.   <wsdl:input message="impl:plusRequest" name="plusRequest" />
27.   <wsdl:output message="impl:plusResponse" name="plusResponse" />
28. </wsdl:operation>
29. - <wsdl:operation name="moins" parameterOrder="a b">
30. <wsdl:input message="impl:moinsRequest" name="moinsRequest" />
31. <wsdl:output message="impl:moinsResponse" name="moinsResponse" />
32. </wsdl:operation>
33. </wsdl:portType>

34. - <wsdl:binding name="CalculatriceSoapBinding" type="impl:Calculatrice">
35. <wsdlsoap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http" />
36. - <wsdl:operation name="plus">
37.   <wsdlsoap:operation soapAction="" />
38. - <wsdl:input name="plusRequest">
39. <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
40. namespace="http://DefaultNamespace" use="encoded" />
41.   </wsdl:input>
42. - <wsdl:output name="plusResponse">
43.   <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
45. namespace="http://localhost:8080/axis/Calculatrice.jws" use="encoded" />
46.   </wsdl:output>
47. </wsdl:operation>
48. - <wsdl:operation name="moins">
49. <wsdlsoap:operation soapAction="" />
50. - <wsdl:input name="moinsRequest">
51. <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
52. namespace="http://DefaultNamespace" use="encoded" />
53.   </wsdl:input>
54. - <wsdl:output name="moinsResponse">
55. <wsdlsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
56. namespace="http://localhost:8080/axis/Calculatrice.jws" use="encoded" />
57.   </wsdl:output>
58.   </wsdl:operation>
59.   </wsdl:binding>

60. - <wsdl:service name="CalculatriceService">
61. - <wsdl:port binding="impl:CalculatriceSoapBinding" name="Calculatrice">
62. <wsdlsoap:address location="http://localhost:8080/axis/Calculatrice.jws" />
```

```
63.  </wsdl:port>
64.  </wsdl:service>
65.  </wsdl:definitions>
```

Expliquons chaque partie du document WSDL précédent :

- **(10 – 23)** La première partie du fichier : définit les paramètres d'entrée(Request) et de sortie(Response) de toutes opérations du service : **moinsRequest** et **moinsResponse** : pour l'opération moins, et **plusRequest** et **plusResponse** : pour l'opération plus .
- **(24 – 33**) Ensuite, la définition abstraite du service Web est faite par la définition du portType, qui encapsule les définitions des opérations plus et moins. On fait ici référence aux messages définis précédemment (paramètres d'entrée et de sortie de l'opération). On obtient une description abstraite du service, indépendante de tout protocole de communication. C'est l'interface du service définissant les méthodes exportées, et leurs paramètres d'entrée et de sortie.
- **(34 – 59)** Il est ensuite possible d'associer ce service à un protocole existant, par la définition d'un binding (dans ce cas c'est le protocole : SOAP). Le binding définit les paramètres d'invocation du service spécifiques au protocole utilisé. On définit ici les paramètres nécessaires à l'utilisation du service via SOAP (lien vers les spécifications du transport utilisé, règles d'encodage pour la sérialisation des messages échangés, etc.).
- **(60 – 65)** La définition du service se termine par la définition des paramètres restants. Par exemple pour un binding SOAP, il reste à définir l'adresse URL du service à invoquer. Notons qu'il est possible de définir plusieurs bindings, et d'associer ces bindings à plusieurs URL, en utilisant la même définition abstraite du service.

6.4. UDDI - Universal Description Discovery and Integration

6.4.1. Principe

UDDI est une spécification définissant la manière de publier et de découvrir les services Web sur un réseau [12]. Ainsi, lorsque l'on veut mettre à disposition un nouveau service, on crée un fichier appelé Business Registry qui décrit le service en utilisant un langage dérivé d'XML suivant les spécifications UDDI.

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

• **Les pages blanches** incluant l'adresse, le contact et les identifiants relatifs aux services Web.

• **Les** pages **jaunes** identifiant les secteurs d'affaires relatifs aux services Web.

• **Les pages** vertes donnant les informations techniques.

En utilisant l'API UDDI l'utilisateur peut alors stocker ces informations dans un nœud (node) UDDI. Elles sont ensuite répliquées de nœud en nœud (principe assez proche dans l'idée des réplications de DNS).

Une fois ceci est fait, le service Web peut alors être connu de tous ceux qui le recherchent [13]. Le modèle UDDI comporte 5 types de structures de données décrites sous forme de schéma XML.

Business entity : elle contient des informations sur l'entreprise qui publie les services dans l'annuaire, cette structure contient les autres éléments de l'UDDI.
Business service : ensemble d'informations sur les services publiés par l'entreprise (nom du service, les catégories…).
Binding Template : ensemble d'informations sur le lieu d'hébergement du service (c.à.d. l'adresse du point d'accès du service).
Tmodel : ensemble d'informations sur le mode d'accès au service (définitions wsdl), il peut s'agir aussi d'une spécification abstraite ou d'une taxonomie.
Publisher Assertion : ensemble d'informations contractuelles entre les partenaires.

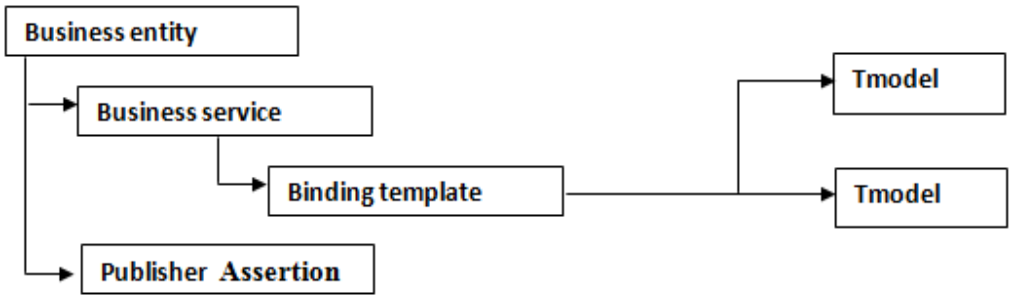

Figure I.9. Une synthèse sur les informations constituant l'annuaire

6.4.2. La recherche d'un service Web

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

- Nom de l'entreprise
- La localisation de l'entreprise
- Identifiant de l'entreprise
- Le nom du service Web …

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

Les annuaires UDDI ont pour but de localiser virtuellement des services Web hébergés sur les serveurs du monde entier. Lors de votre recherche vous pouvez ainsi vous renseigner sur tous les services mis à disposition d'une entreprise, avoir des informations sur l'entreprise,
Les opérateurs UDDI vous certifient la sécurité et l'intégrité des services Web contenus dans leurs annuaires.

Les appels aux sites des opérateurs donnent accès aux différents éléments de l'enregistrement d'un service Web dans un annuaire UDDI:

• **find_binding :** récupère la liaison du service considéré.

• **find_business :** récupère l'identité de l'entreprise productrice du service Web.

• **find_relatedbusiness :** récupère la liste des entreprises étant reliées (filiale, département. partenaire, ...) à l'entreprise productrice du service Web.

• **find_service :** récupère la définition du service.

• **find_tmodel :** récupère le modèle de données associé.

• **get_bindingDetail :** récupère, par une liaison précédemment établie par find_binding les champs individuels.

• **get_businessDetail, getbusinessDetailExt :** récupère une entité précédemment établie par find business les attributs individuels.

• **get serviceDetail :** récupère un service précédemment établi par find_service les attributs individuels du service (prototypes des méthodes).

• **gel_tmodelDetail :** récupère un modèle établie par find_tmodel les champs individuels. [4]

6.4.3. La publication d'un service Web

Comme dans WSDL, la liaison UDDI regroupe, pour un protocole de communication donné, les données techniques nécessaires à l'exploitation du service Web (adresse IP, noms de domaines, les informations sur les modalités d'usage du service, ...)
La publication par une entreprise d'un service Web requiert que celle-ci s'authentifie auprès du site de l'opérateur UDDI. L'entreprise doit s'enregistrer chez l'opérateur si cela n'est pas déjà le cas.

Une fois le site de l'opérateur choisi, les modifications ultérieures ou la mise à jour de cet enregistrement doivent être faites auprès du même opérateur.

Lors de l'enregistrement, vous pouvez enregistrer simultanément un ensemble d'entreprises affiliées, des relations entres entreprises indépendantes décrivant des accords croisés. L'API se décompose en 3 groupes:

- La manipulation (save et delete).
- L'authentification des commandes par jeton (get_authToken et discard_authToken).
- L'ajout de relations inter entreprises (joint_ventures). [8]

7. Les avantages et inconvénient des services Web

7.1. Avantages

• Les services Web fournissent l'interopérabilité entre divers logiciels fonctionnant sur diverses plates-formes.
• Les services Web utilisent des standards et protocoles ouverts.
• Les protocoles et les formats de données sont au format texte dans la mesure du possible, facilitant ainsi la compréhension du fonctionnement global des échanges.
• Basés sur le protocole HTTP, les services Web peuvent fonctionner au travers de nombreux pare-feux sans nécessiter des changements sur les règles de filtrage.

• Les outils de développement, s'appuyant sur ces standards, permettent la création automatique de programmes utilisant les services Web existants. [13]

7.2. Inconvénients

• Les normes de services Web dans certains domaines sont actuellement récentes.

• Les services Web ont de faibles performances par rapport à d'autres approches de l'informatique répartie telles que le RMI, CORBA, ou DCOM.

• En l'utilisation du protocole HTTP, les services Web peuvent contourner les mesures de sécurité mises en place à travers les firewalls. [15]

8. Conclusion

Les services Web sont des applications accessibles par l'échange de documents XML entre deux URL. Ils permettent une souplesse d'utilisation, et une accélération du développement d'applications. En adoptant cette technologie, nous faciliterons l'interopérabilité et la réutilisation du code. Le chapitre suivant présente une étude de cas, dans laquelle nous montrons des applications hétérogènes qui échangent des données selon le paradigme des services Web.

Chapitre II

Les algorithmes génétique

1. Introduction

Les algorithmes génétiques [16] sont des algorithmes d'optimisation s'appuyant sur des techniques dérivées de la génétique et des mécanismes d'évolution de la nature: croisements, mutations, sélections, ….etc. Ils appartiennent á la classe des algorithmes évolutionnaires.

2. Historique

1860

Charles Darwin publie son livre intitulé L'origine des espèces au moyen de la sélection naturelle ou la lutte pour l'existence dans la nature. Dans ce livre, Darwin rejette l'existence «de systèmes naturels figés», déjà adaptés pour toujours à toutes les conditions extérieures, et expose sa théorie de l'évolution des espèces : sous l'influence des contraintes extérieurs, les êtres vivants se sont graduellement adaptés à leur milieu naturel au travers de processus de reproductions.

20ième siècle

Mise en évidence de l'existence de mutations génétiques. Les problèmes de traitement de l'information sont résolus de manières figés : lors de sa phase de conception, le système reçoit toutes les caractéristiques nécessaires pour les conditions d'exploitations connues au moment de sa conception ce qui empêche une adaptation à des conditions d'environnement inconnues, variables ou évolutives. Les chercheurs en informatique étudient donc des méthodes pour permettent aux systèmes d'évoluer spontanément en fonction de nouvelles conditions.

1966

Programmation évolutionnaire L. J. Fo gel.

1973

Stratégie d'évolution I. Rechenberg.

1975

Dans les années 1960, John Holland étudient les systèmes évolutifs et, en 1975, il introduit le premier modèle formel des algorithmes génétiques (the canonical genetic algorithm AGC) dans son livre Adaptation in Natural and Artificial Systems. Ce modèle servira de base aux recherches ultérieures.

1989

David Goldberg publie un ouvrage de vulgarisation des algorithmes génétiques.

années 1990

Programmation d'une panoplie d'algorithmes génétiques transcris en C++, appelée GAlib. Cette librairie contient des outils pour des problèmes d'optimisation en utilisant les AG. Elle est conçue pour servir de support de programmation.

3. Terminologie

Avant d‘aborder comment un problème soit résolu à l‘aide d‘un AG, nous devons également définir le vocabulaire employé par notre travail

- ***Chromosome*** **:** En biologie, il est défini comme le porteur de l‘information génétique nécessaire à la construction et au fonctionnement d‘un organisme. Dans le cadre des AG, il correspond à un élément représentant une solution possible d‘un probleme donné.
- ***Gène*** **:** En biologie, il représente une partie du chromosome, chaque chromosome est constitué d‘un certain nombre de gènes. Pour un AG, chaque chromosome est divisé en un ensemble d‘unités le constituant dites gènes.
- ***Génotype*** *:* Dans les systèmes naturels, l‘ensemble du matériel génétique est appelé le génotype. Dans les AG, l‘ensemble des chaînes est appelé structure.
- ***Phénotype :*** Dans les systèmes naturels, l‘organisme formé par l‘interaction de l‘ensemble du matériel génétique avec son environnement est appelé le phénotype.

Dans les AG, les structures décodées forment un ensemble de paramètres donné, ou une solution ou un point dans l‘espace des solutions.

- ***Allèle :*** Dans les systèmes naturels, l‘allèle est une composante du gène. Les alleles sont les différentes valeurs que peuvent prendre les gènes. Dans les AG, l‘allèle est également appelé valeur caractéristique.
- ***Locus :*** Le locus est la position d‘un gène dans le chromosome.
- ***Individu :*** En biologie un individu est une forme qui est le produit de l‘activité des gènes. Pour un AG, il est réduit à un chromosome et on l‘appelle donc chromosome ou individu pour désigner un même objet.

• ***Population*** **:** Dans un système naturel, une population est simplement un ensemble d'individus. Par analogie, elle se définit comme l'ensemble des chromosomes. Elle est aussi appelée une génération.

• ***Parents :*** Dans un système naturel, les individus peuvent se reproduire en créant de nouveaux individus formant une nouvelle génération afin d'assurer la continuité de la vie. Dans le cadre d'un AG, les parents correspondent aux individus pouvant s'imposer ce qui donne naissance à de nouveaux individus descendants afin de former une nouvelle génération.

• ***Descendants :*** Dans un système naturel, la continuité de la vie se fait par la reproduction des individus de la population, il en résulte la naissance d'un certain nombre d'individus dits descendants et qui participent à la création de la nouvelle génération. Dans un AG, les descendants peuvent être considérés comme étant le résultat direct du processus de la reproduction des parents, chaque descendant hérite des caractéristiques issues de ses parents.

Les autres concepts comme le croisement et la mutation seront discutés par la suite de se chapitre

4. Fonctionnement de l'approche

4.1. But

Le but des AG est de déterminer les extrêmes d'une fonction $f : X \mapsto IR$, où X est un ensemble quelconque appelé espace de recherche et f est appelée fonction d'adaptation ou fonction d'évaluation ou encore fonction fitness. La fonction agit comme une «boite noire» pour l'AG. Aussi des problèmes très complexes peuvent être approchés par programmation génétique sans avoir de compréhension particulière du problème.

4.2. Codage d'une population

Le premier pas dans l'implantation des algorithmes génétiques est de créer une population d'individus initiaux. En effet, les algorithmes génétiques agissent sur une population d'individus, et non pas sur un individu isolé. Par analogie avec la biologie, chaque individu

de la population est codé par un chromosome ou génotype (Holland, 1975). Une population est donc un ensemble de chromosomes. Chaque chromosome code un point de l'espace de recherche. L'efficacité de l'algorithme génétique va donc dépendre du choix du codage d'un chromosome.

4.3. Fonction d'évaluation et fonction fitness

Pour calculer le coût d'un point de l'espace de recherche, on utilise une fonction d'évaluation. L'évaluation d'un individu ne dépendant pas de celle des autres individus, le résultat fournit par la fonction d'évaluation va permettre de sélectionner ou de refuser un individu pour ne garder que les individus ayant le meilleur coût en fonction de la population courante : c'est le rôle de la fonction fitness. Cette méthode permet de s'assurer que les individus performants seront conservés, alors que les individus peu adaptés seront progressivement éliminés de la population.

4.4. L'hybridation (ou cross over)

A partir de deux individus, on obtient deux nouveaux individus (enfants) qui héritent de certaines caractéristiques de leurs parents. L'hybridation sélectionne des gènes parmi deux individus appelés parents. A partir de ces gènes sont générés les enfants. La probabilité d'hybridation représente la fréquence à laquelle les hybridations sont appliquées.

- S'il n'y a pas d'hybridation, les fils sont l'exacte copie des parents.
- S'il y a hybridation, les fils sont composés d'une partie de chacun de leur parents.
- Si la probabilité est de 0%, la nouvelle génération est la copie de la précédente.
- Si la probabilité est fixée à 100%, tous les descendants sont générés par hybridation.

L'hybridation est mis en place pour que les nouveaux chromosomes gardent la meilleur partie des chromosomes anciens. Ceci dans le but d'obtenir, peut-être, de meilleurs chromosomes. Néanmoins, il est quand même important qu'une partie de la population survive à la nouvelle génération.

4.5. La mutation

La mutation génère des «erreurs» de recopie, afin de créer un nouvel individu qui n'existait pas auparavant. Le but est d'éviter à l'AG de converger vers des extrema locaux de la fonction et de permettre de créer des éléments originaux. Si elle génère un individu plus faible l'individu est éliminé. La probabilité de mutation représente la fréquence à laquelle les gènes d'un chromosome sont mutés.

- S'il n'y a pas de mutation, le fils est inséré dans la nouvelle population sans changement.
- Si la mutation est appliquée, une partie du chromosome est changée.

La mutation est prévue pour éviter au AG de s'enliser dans des optima locaux. Mais si elle est trop fréquente, le AG est orienté vers une recherche aléatoire de la bonne solution.

4.6. Itération

A partir d'une population initiale de i individus, l'AG sélectionne une population intermédiaire de i' individus en faisant une sélection sur la population initiale (un même individu peut être sélectionné plusieurs fois ou peut ne pas être sélectionné du tout, en fonction de la valeur de sa fonction d'évaluation). Les i' individus de la population se croisent deux à deux (les couples se forment aléatoirement) pour construire i nouveaux individus. Ces individus passent par un opérateur de mutation (qui agit aléatoirement avec une possibilité faible 2-3% de bits) pour former une nouvelle population. On réitère ensuite le procédé à partir de cette population jusqu'à obtenir une solution que l'on juge satisfaisante.

Le fonctionnement de tout AG peut être décrit par le principe illustré par la figure II.1

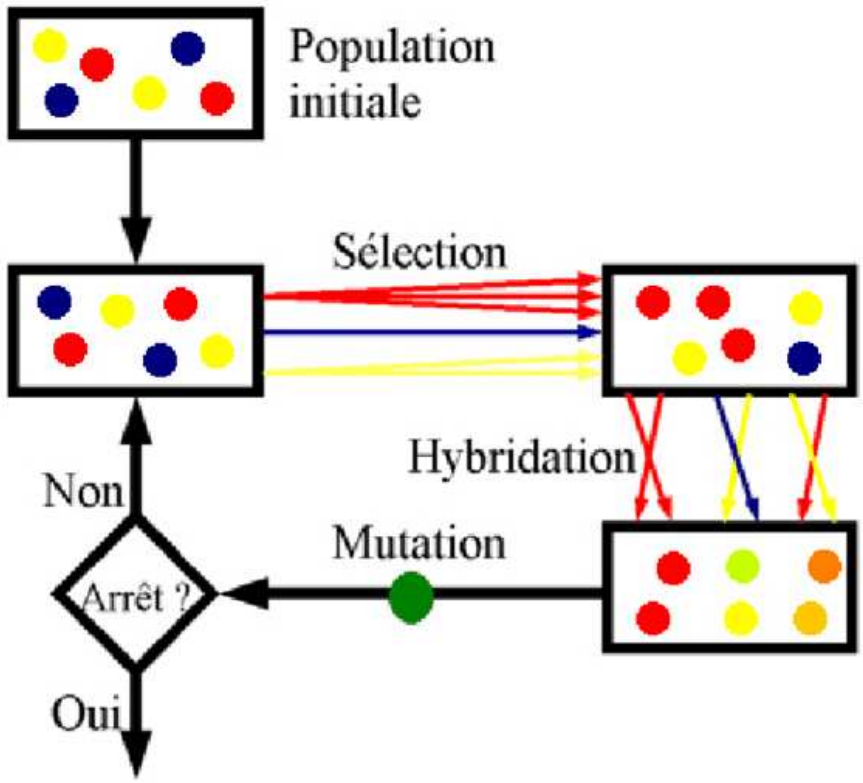

Figure II.1 Schéma résumant le fonctionnement des AG

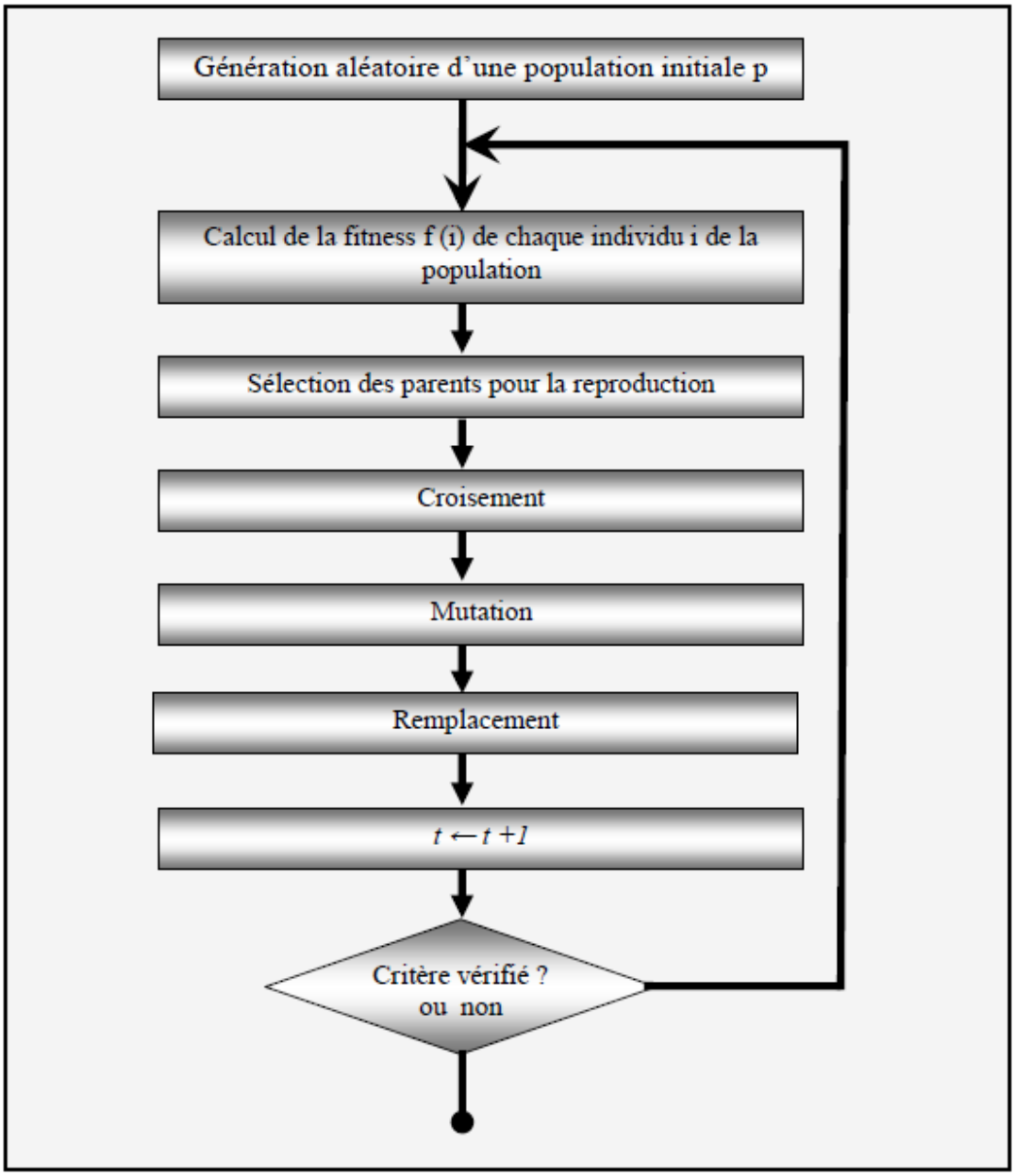

Figure II.2 Organigramme des AG

4.7. L'algorithme

* Initialisation: Générer un ensemble de solutions initiales.

* Boucle Principale

1) Evolution:

- Sélection: Choisir avec une probabilité proportionnelle à leur qualité une liste d'individus.

- Reproduction: Générer à partir de cette liste de nouveaux individus à l'aide des opérateurs génétiques
- Remplacement: Eliminer avec une probabilité inversement proportionnelle à leur qualité certains individus.

2) Réactualisation de la meilleure solution

3) Allez en 1) tant que Nombre de Générations <= Valeur Prédéterminée [5]

4.8. Analyse de l'algorithme

1- On génère une population aléatoire de taille de la population contrôleurs.

2- On évalue les contrôleurs de la génération 0 (on calcule leur fitness) en les injectant dans la grille 2D.

3- On sélectionne les contrôleurs qui vont servir de géniteurs.

4- On crée la nouvelle génération en copiant, mutant et croisant les géniteurs, selon des proportions à définir.

5- On évalue les contrôleurs de la nouvelle génération (on calcule leur fitness) en les injectant dans la grille 2D.

6- On remplace l'ancienne population par la nouvelle.

7- On recommence 3, 4, 5 et 6 jusqu`a ce qu'on ait atteint une fitness suffisante ou le nombre maximum de générations.

Le mode de sélection et la fitness sont à définir par l'utilisateur.

Les paramètres taille de la population, proportion de copies, proportion de mutations, proportion de cross-over, et nombre de générations, sont à choisir au lancement de l'expérimentation.

5. Les différentes méthodes de sélection

5.1. Sélection par roulette (wheel)

Les parents sont sélectionnés en fonction de leur performance. Meilleur est le résultat codé par un chromosome, plus grandes sont ses chances d'être sélectionné. Il faut imaginer une sorte de roulette de casino sur laquelle sont placés tous les chromosomes de la population, la

place accordée à chacun des chromosomes étant en relation avec sa valeur d'adaptation. Cette roulette est représentée par la figure II.3

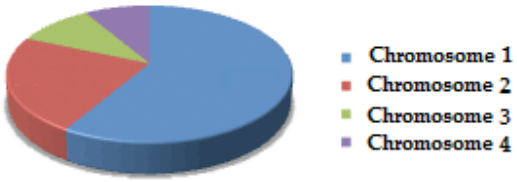

Figure II.3 Exemple de sélection par roulette

Ensuite, la bille est lancée et s'arrête sur un chromosome. Les meilleurs chromosomes peuvent ainsi être tirés plusieurs fois et les plus mauvais ne jamais être sélectionnés. Cela peut être simulé par l'algorithme suivant:

1. On calcule la somme S1 de toutes les fonctions d'évaluation d'une population.
2. On génère un nombre r entre 0 et S1.
3. On calcule ensuite une somme S2 des évaluations en s'arrêtant dès que r est dépassé.
4. Le dernier chromosome dont la fonction d'évaluation vient d'être ajoutée est sélectionné.

5.2. Sélection par rang

La sélection précédente rencontre des problèmes lorsque la valeur d'adaptation des chromosomes varient énormément. Si la meilleure fonction d'évaluation d'un chromosome représente 90% de la roulette alors les autres chromosomes auront très peu de chance d'être sélectionnés et on arriverait à une stagnation de l'évolution.

La sélection par rang trie d'abord la population par fitness. Ensuite, chaque chromosome se voit associé un rang en fonction de sa position. Ainsi le plus mauvais chromosome aura le rang 1, le suivant 2, et ainsi de suite jusqu'au meilleur chromosome qui aura le rang N (pour une population de N chromosomes). La sélection par rang d'un chromosome est la même que par roulette, mais les proportions sont en relation avec le rang plutôt qu'avec la valeur de l'évaluation. Le tableau II.1 fournit un exemple de sélection par rang. Avec cette méthode de sélection, tous les chromosomes ont une chance d'être sélectionnés. Cependant,

elle conduit à une convergence plus lente vers la bonne solution. Ceci est dû au fait que les meilleurs chromosomes ne diffèrent pas énormément des plus mauvais.

Chromosomes	1	2	3	4	5	6	Total
Probabilités initiales	89 %	5 %	1 %	4 %	3 %	2 %	100 %
Rang	6	5	1	4	3	2	21
Probabilités finales	29 %	24 %	5 %	19 %	14 %	9 %	9 %

Tableau II.1 Exemple de sélection par rang pour 6 chromosomes

5.3. Sélection steady-state

Ce n'est pas une méthode particulière de sélection des chromosomes parents. L'idée principale est qu'une grande partie de la population puisse survivre à la prochaine génération. L'algorithme génétique marche alors de la manière suivante. A chaque génération sont sélectionnés quelques chromosomes (parmi ceux qui ont le meilleur coût) pour créer des chromosomes fils. Ensuite les chromosomes les plus mauvais sont retirés et remplacés par les nouveaux. Le reste de la population survie à la nouvelle génération.

5.4. Sélection par tournoi

Sur une population de m chromosomes, on forme m paires de chromosomes. Dans les paramètres de l'AG, on détermine une probabilité de victoire du plus fort. Cette probabilité représente la chance qu'a le meilleur chromosome de chaque paire d'être sélectionné. Cette probabilité doit être grande (entre 70% et 100%). A partir des m paires, on détermine ainsi m individus pour la reproduction.

5.5. Elitisme

A la création d'une nouvelle population, il y a de grandes chances que les meilleurs chromosomes soient perdus après les opérations d'hybridation et de mutation. Pour éviter cela, on utilise la méthode d'élitisme. Elle consiste à copier un ou plusieurs des meilleurs

chromosomes dans la nouvelle génération. Ensuite, on génère le reste de la population selon l'algorithme de reproduction usuel. Cette méthode améliore considérablement les algorithmes génétiques, car elle permet de ne pas perdre les meilleurs solutions.

6. Conclusion

Les algorithmes génétiques fournissent des solutions proches de la solution optimale à l'aide des mécanismes de sélection, d'hybridation et de mutation. Ils sont applicables à de nombreux problèmes, dont le problème du voyageur de commerce par exemple.

Chapitre III

Conception et implémentation

1. Introduction

Les requêtes orientées services permettant aux utilisateurs d'accéder à plusieurs services Web de manière transparente et efficace. En outre, comme les services Web avec des fonctionnalités similaires devraient être fournis par les fournisseurs concurrents, un défi majeur est de concevoir des stratégies d'optimisation pour trouver les meilleurs services Web ou de la composition de celle-ci à l'égard de la qualité attendue fournie par l'utilisateur (au niveau par exemple de temps, de coût, et de réputation), Les normes existantes à base de technologies de découverte de services sont nettement insuffisants pour la construction d'une infrastructure de service à part entière de requête, le paradigme actuel de recherche par mots clés ne peut pas toujours localiser précisément les services Web en partie à cause de la sémantique riche énoncés dans ces services. Le traitement des requêtes sur les services Web est un nouveau concept qui va au-delà de la vision traditionnelle centrée sur les données du traitement des requêtes, qui est principalement axé sur la performance. Il met l'accent sur les paramètres de l'utilisateur de sélectionner la qualité des services multiples qui sont l'équivalent des fonctionnalités différentes mais présentent la qualité de service Web (QoWS).[6]

Dans se chapitre nous présentons un algorithme d'optimisation pour sélectionner efficacement les (SEP : Service Execution Plans) avec les meilleurs QoWS. Une fonction de score est alors présentée pour évaluer l'ensemble des SEP.

1.1. Sénario : courtage de la voiture

Comme une façon d'illustrer ce travail, nous utilisons une application à partir du domaine de courtage de la voiture [6]. Un scénario typique serait d'un client X, a l'intention d'acheter une voiture d'occasion ayant un modèle spécifique, la marque, et le kilométrage. Il veut naturellement obtenir la meilleure affaire. Supposons que le client X a accédé à une infrastructure de services Web où les différentes entités qui jouent un rôle dans l'achat de voitures sont représentées par les services Web. Exemples de services Web qui doivent être accessibles comprennent achat d'un véhicule (CP : Car Purchase), une assurance automobile (CI : Car Insurance) et le financement (FI : FInancing).

Un seul service Web peut fournir de multiples opérations. Différentes opérations peuvent aussi avoir des relations de dépendance. Par exemple, les opérations paymentHistory et financingQuote sont tous deux offerts par le service de financement. Cette dernière opération dépend de la première opération, c'est-à-dire, l'historique des paiements décide le financingQuote.

Pour acheter une voiture, le client X voudrait tout d'abord connaître le prix de la voiture choisie et le rapport historique du véhicule. Il doit ensuite obtenir le devis d'assurance. Enfin, parce que le client X a besoin de l'assistance du financement, il veut aussi savoir le financingquote. En outre, le client X peut avoir des exigences particulières sur la qualité des opérations de service. Par exemple, il veut passer moins de 20 dollars pour obtenir le rapport de l'historique du véhicule.

Le client X peut spécifier sa voiture par une requête de service déclaratif. La requête est formée de service basée sur le modèle de service. La requête de service déclaratif est une expression de calcul de service. Il spécifie les fonctionnalités que l'utilisateur souhaite récupérer en termes d'opérations de services. Il n'est pas nécessaire pour l'utilisateur d'avoir la connaissance des contraintes de dépendance entre les opérations de service.

En outre, la requête de service permet également à l'utilisateur de spécifier les exigences relatives à la qualité des opérations de service.

1.2. Les paramètres QoWS (Quality of Web Service)

Nous utilisons cinq paramètres QoWS pour évaluer les opérations de service: temps de latence, de fiabilité, disponibilité, le coût, et la réputation.

1.2.1. Latence

Définit par: Timeprocess(op) + Timeresults(op), où Timeprocess(op) est le temps pour traiter op, et Timeresults(op) est le temps pour transmettre et recevoir les résultats

1.2.2. Fiabilité

Définit par: Nsuccess(op)/Ninvoked(op), où Nsuccess(op) est le nombre de fois que op a été exécutée avec succès et Ninvoked(op) est le nombre total d'appels

1.2.3. Disponibilité

Défini par: UpTime(op)/TotalTime(op) où UpTime est le temps où op a été accessible pendant le temps de mesure total TotalTime(op)

1.2.4. Coût

Définit par: Le montant en dollars pour exécuter l'opération

1.2.5. Réputation

$\sum_{u=1}^{n}$ Rankingu (op)/n, 1 ≤ Réputation ≤ 10, où Rankingu est le classement par u utilisateur et n est le nombre de fois où op a été classé

Les utilisateurs peuvent avoir de préférence particulière pour certains paramètres QoWS sur les autres. Par exemple, le client X veut utiliser le SEP avec la réputation la plus élevé, c'est-à-dire il préfère acheter la voiture auprès des fournisseurs les plus réputés. La préférence est sur l'ensemble de SEP qui contient habituellement de multiples opérations. Elle est différente de l'exigence de qualité dans la requête de service sur certaines opérations individuelles (par exemple, obtenir le rapport d'historique pour moins de 20 dollars).

L'optimisation QoWS est «centrée sur l'utilisateur ». Le rôle de l'utilisateur est de donner le poids des paramètres QoWS. Le but est de trouver le SEP avec la meilleure qualité en fonction des préférences de l'utilisateur sur l'ensemble des SEP.

La qualité d'un SEP peut être définie comme un vecteur de QoWS :

Qualité(SEPi) = (latence(SEPi), fiabilité(SEPi), disponibilité(SEPi), coût(SEPi), réputation(SEPi)).

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].

1.3. La fonction « Score »

Nous définissons une fonction de score pour calculer une valeur hors du vecteur de QoWS des SEP. Cela peut faciliter la comparaison de la qualité des SEP. Comme les utilisateurs peuvent avoir des préférences sur la façon dont leurs requêtes sont traitées, elles peuvent indiquer l'importance relative des paramètres QoWS. Nous attribuons des poids allant de 0 à 1à chaque paramètre QoWS pour refléter le niveau des importances.

Paramètres QoWS	Fonction d'agrégation
Latence	$\sum_{i=1}^{n}$ latence (op i)
Fiabilité	$\prod_{i=1}^{n}$ fiabilité (op i)
Disponibilité	$\prod_{i=1}^{n}$ disponibilité (op i)
Coût	$\sum_{i=1}^{n}$ coût (op i)
Réputation	$\frac{1}{n}\sum_{i=1}^{n}$ réputation (op i)

Tableau III.1 Les fonctions d'agrégation des QoWS pour un SEP

Nous utilisons la fonction score S suivante pour évaluer la qualité des plans d'exécution du service. En utilisant la fonction de score, l'optimisation QoWS est de trouver le plan d'exécution avec le score maximum.

$$S = \left(\sum_{Qi \in Neg} Wi \frac{Qi\,max - Qi}{Qi\,max - Qi\,min} + \sum_{Qi \in Pos} Wi \frac{Qi - Qi\,min}{Qi\,max - Qi\,min} \right)$$

Où Qi est la iéme qualité de service, Qi est calculée par la function d'agrégation, Neg et Pos sont les ensembles de QoWS négatifs et positifs respectivement. En négative (resp. positive) des paramètres, le plus élevé (resp. inférieure) de la valeur, le pire est la qualité.Qi max est la valeur maximale pour le ième paramètre QoWS pour tous les plans possibles d'exécution du service et Qi min est le minimum.

2. Description de la base

2.1. Description de la base

La première base de données(A) se compose d'un ensemble de trois services, chaque service contient deux opérations: CP (OP_1, OP_2), CI (OP_3, OP_4) et FI (OP_5, OP_6), où OP_1, OP_2, OP_3, OP_4, OP_5, OP_6 représentent les opérations carQuote, historyReport, paymentHistory,financingQuote, drivingHistory, et ins uranceQuote, respectivement. Chaque opération est un tableau de cent lignes et cinq colonnes, les colonnes représentent les QoWS (Latence, Fiabilité, Disponibilité, Coût, Réputation), et les lignes représentent les fournisseurs, en tenant compte que chaque service a exactement cent fournisseurs.

Les données de chaque tableau sont générées aléatoirement d'une façon où la latence est prise dans l'intervalle comprit entre 0 et 300 seconde, la fiabilité entre 0.5 et 1 fois, la disponibilité entre 0.7 et 1, le coût entre 0 et 30$, et la réputation entre 0 et 5. Pour générer aléatoirement ces valeurs nous avons utilisé la fonction =ALEA.ENTRE.BORNES(min;max) sous Excel pour les valeurs de la latence, avec min est la borne inferieure de l'intervalle de la latence et max est la borne supérieure, la même fonction est utilisée pour le coût et la réputation, pour la fiabilité nous avons utilisé la fonction =(2,71828183)^(1-0,5*ALEA()), et pour la disponibilité nous avons utilisé la fonction =(2,71828183)^(1-0,3*ALEA()), lorsqu'on fait le logarithme népérien des valeurs résultantes nous obtiendrons les valeurs comprise dans l'intervalle de chaque paramètre (fiabilité et disponibilité), le but des deux dernières fonctions utilisées est pour avoir le même opérateur ($\sum$) pour toutes les fonctions d'agrégation.

La deuxième BDD (B) se compose de 40 lignes et 3 colonnes, chaque ligne représente le choix aléatoire d'un fournisseur de chaque service.

2.2. Description de la requête

Nous considérons une requête de service qui aide à obtenir la voiture du client X, y compris l'offre de prix et le rapport de l'historique d'une voiture d'occasion, devis d'assurance, devis de financement. La requête de service peut être exprimée comme une expression algébrique de service comme suit :

$$Q : X_{\{op1,\ op\ 2,\ op3,\ op4\}} [\delta_{\lambda 4\ (op2)\ \leq 20}\ (CP) \oplus_{\lambda 3\ (op1)\ <\ \lambda 3\ (op4)} \delta_{\lambda 3\ (op4)\ \leq 20} (CI) \otimes FI]$$

Où OP1, OP2, OP3, OP4 représentent les opérations carQuote, historyReport, paymentHistory, et financingQuote, respectivement.

3. Conception

3.1. Diagramme de cas d'utilisation

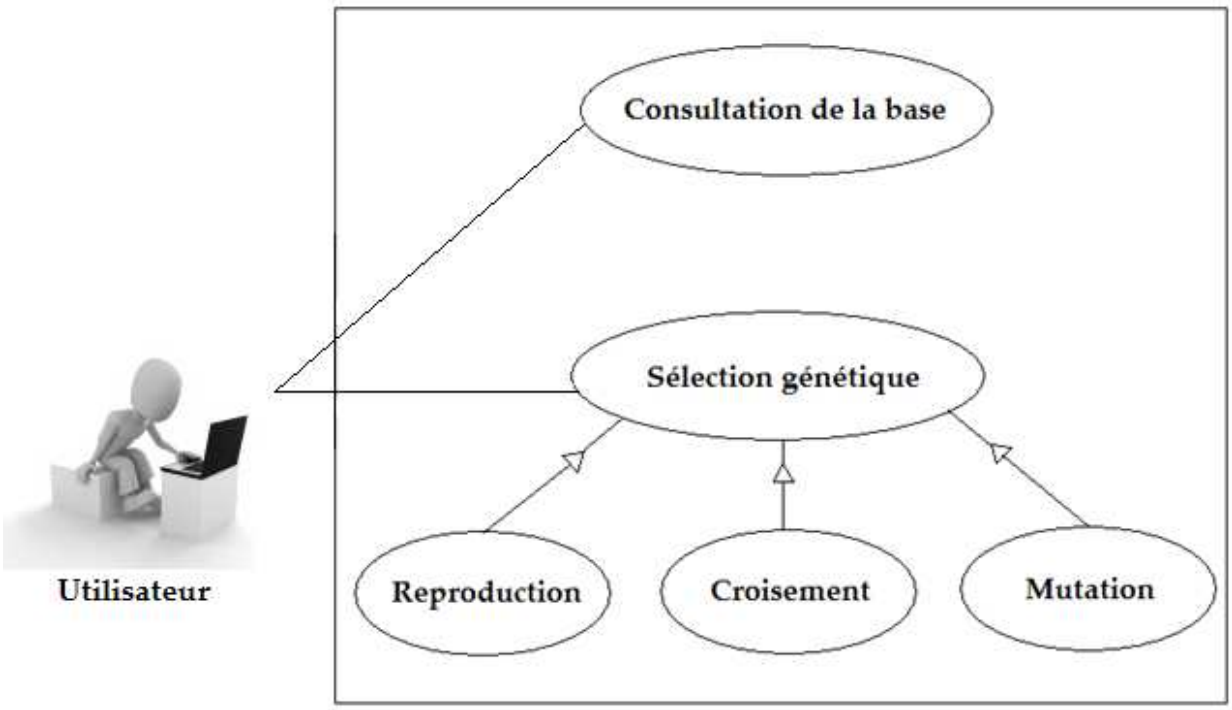

Figure III.1 Diagramme de cas d'utilisation

3.2. Diagramme de classe

Voici le diagramme de classe qui représente l'architecture conceptuelle de notre système:

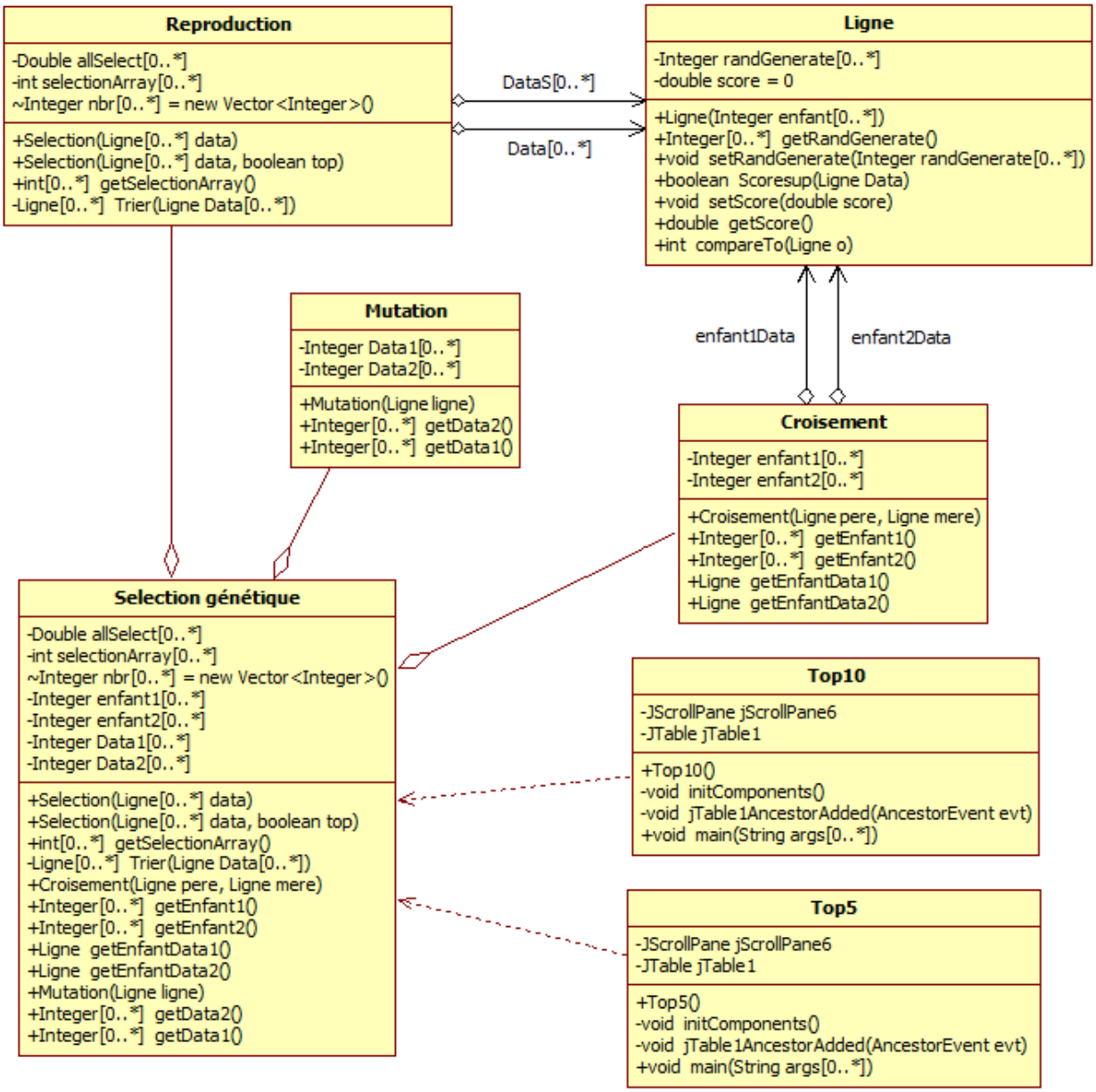

Figure III.2 Diagramme de classe

4. Implémentation

Voici quelques captures d'écran des différentes parties de notre application

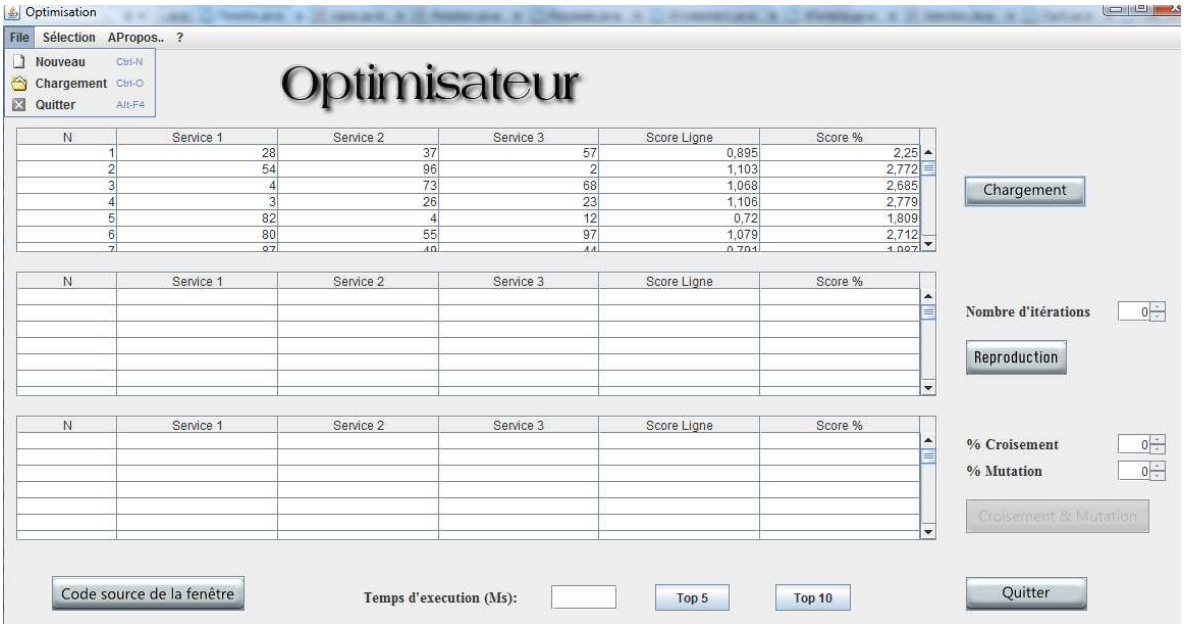

Figure III.3 Chargement

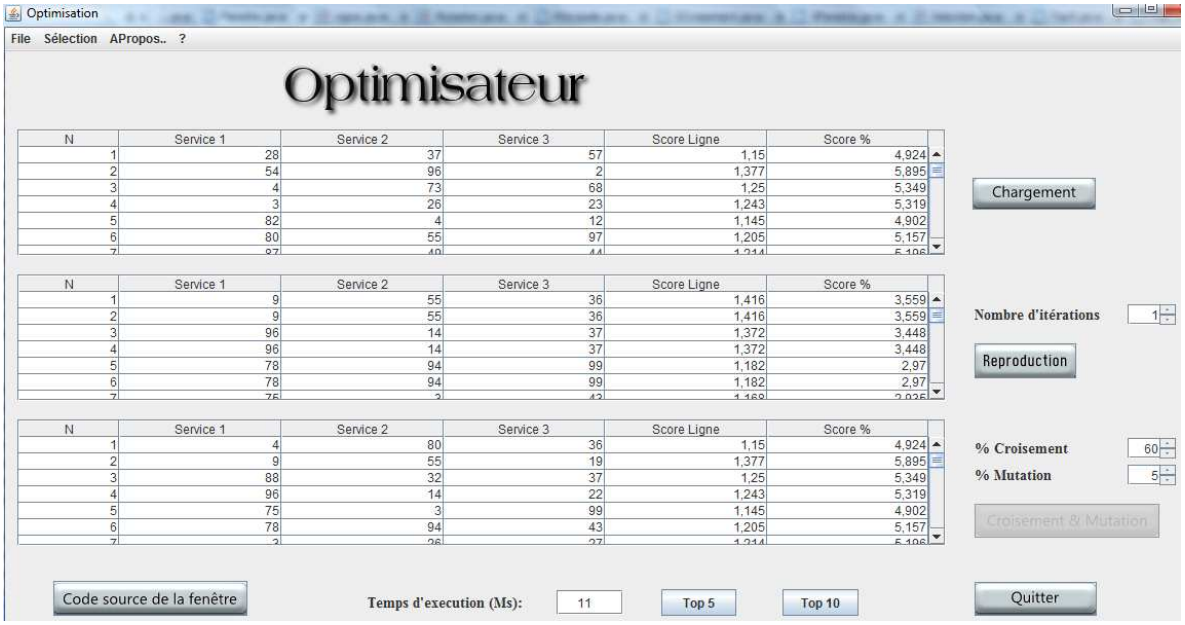

Figure III.4 Sélection génétique

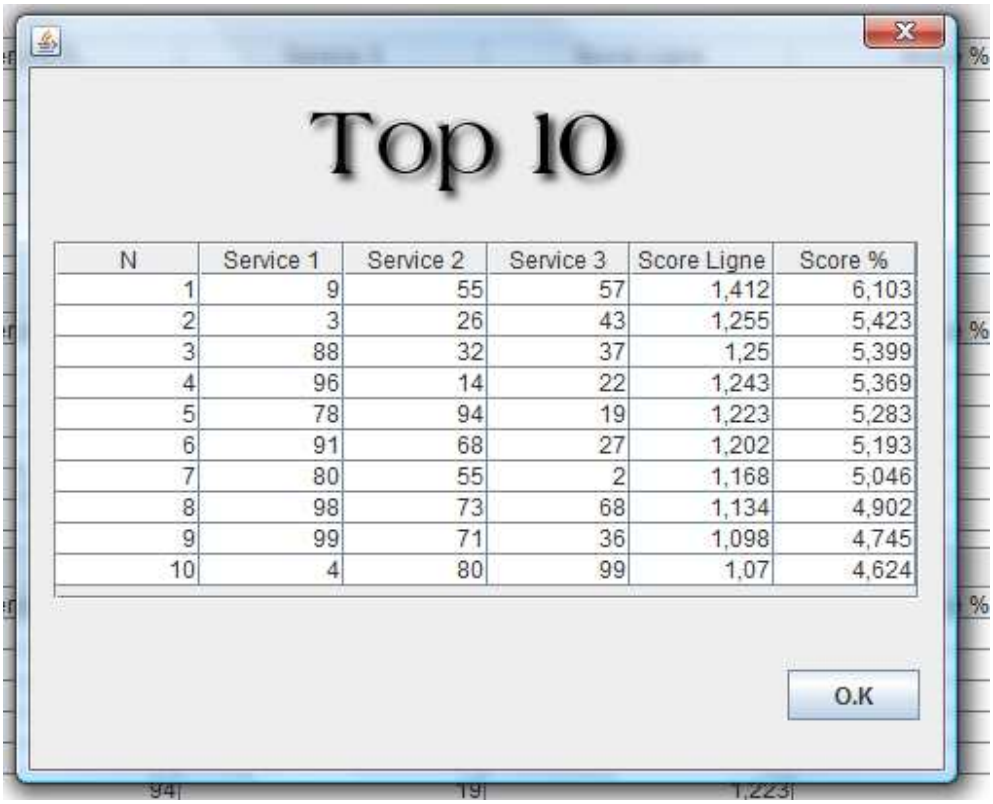

Figure III.5 Top 10 des résultats

4.1. Connexion et chargement

Tout d'abord nous avons commencé par charger les données de la BDD (A), et de la BDD (B) qui représentent la population initial, en créant une classe connexion sachant que nous avons configuré les fichiers Excel dans l'administrateur de sources de données ODBC en sélectionnant le chemin de mon classeur Excel qui contient la BDD(A et B). Le boutton chargement dans mon application affiche dans le premier tableau les lignes de la BDD (B) et leur score, le score est calculé avec la fonction score vue précédemment

4.2. La sélection génétique

Le boutton sélection génétique fait une sélection sur les données des services du tableau 1 avec le score correspondant a chaque ligne. La sélection est realisé de la manière suivante:
On prend a chaque fois le pourcentage du score le plus élevé, la ligne correspondante a ce score est dupliquée n fois dans le deuxieme tableau celui de la sélection génétique, le nombre n est l'arrondissement du résultat de l'opération %score $\times \frac{40}{100}$

On répette la même procedure jusqu'a se qu'on obtient 40 lignes dans le 2éme tableau

4.3. Le croisement et la mutation

Le boutton croisement et mutation fait le croisement, la mutation et le score de chaque ligne résultante, tout ça en récupérant les données des trois services du tableau 2 et le pourcentage de croisement et le pourcentage de mutation, notant que le pourcentage de croisement ne doit pas dépasser 80%, et le pourcentage de mutation ne doit pas dépasser 20%.

4.3.1. Le croisement

Un croisement est realisé en échangeant une partie d'une ligne de services du tableau de sélection génétique (gène du chromosome 1) avec une autre partie d'une autre ligne du même tableau (gène du chromosome 2) parrallèle a la première (le meme locus), sachant que ces deux lignes ne sont pas identiques et ne sont pas déjà prises auparavant. On répétte cette function jusqu'à se qu'on arrive a un nombre maximal de lignes croisées indiqué par le pourcentage de croisement.

4.3.2. La mutation

La mutation est faite en échangeant une partie d'une ligne de services du tableau de sélection génétique (gène du chromosome) qui n'est pas déjà prise (ni pour le croisement ni pour la mutation) avec une autre partie quelconque d'une ligne de la population. On répétte cette function jusqu'à se qu'on arrive à un nombre maximal de lignes mutées indiqué par le pourcentage de mutation

4.4. Reproduction

Le menu reproduction fait les trois operations vues précédemment, la sélection génétique, le croisement et la mutation, en lui attribuant le nombre d'itérations pour répetter le processus de la reproduction n fois, où n est le nombre d'itération et ne doit pas dépasser 40. On donne aussi le score correspondant a chaque ligne résultante.

Le champs "temps d'execution" calcule le temps d'exécution en mili seconde du processus de la reproduction ou du croisement et de la mutation

5. Expérimentation

Nous avons mené une expérience pour évaluer la performance de l'approche proposée. Nous utilisons le scénario du courtage de voiture comme notre environnement de test pour configurer les paramètres de l'expérience. Le but est de démontrer comment notre approche peut aider le client X a sélectionner la meilleure offre. Nous courons notre expérience sous NetBeans IDE 6.8 de Sun Microsystems sous le système d'exploitation Windows, Processeur Core 2 Duo, 3 Giga de RAM.

Nous avons mené les 12

expériences afin d'évaluer le système, les 6 premières expériences avec 1 itération et les 6 dernières expériences avec 3 itérations, en changeant a chaque fois les paramètres croisement et mutation. On a obtenu les résultats suivants:

Itération	Croisement	Mutation	Meilleur score	Temps d'exécution
1	60	5	1.435	14
1	40	5	1.396	17
1	80	5	1.412	18
1	90	5	1.4	19
1	60	10	1.431	14
1	60	20	1.355	11

Tableau III.2 Les résultats avec 1 itération

Itération	Croisement	Mutation	Meilleur score	Temps d'exécution
3	60	5	1.347	25
3	40	5	1.416	26
3	80	5	1.482	28
3	90	5	1.482	30
3	60	10	1.482	28
3	60	20	1.416	27

Tableau III.3 Les résultats avec 3 itérations

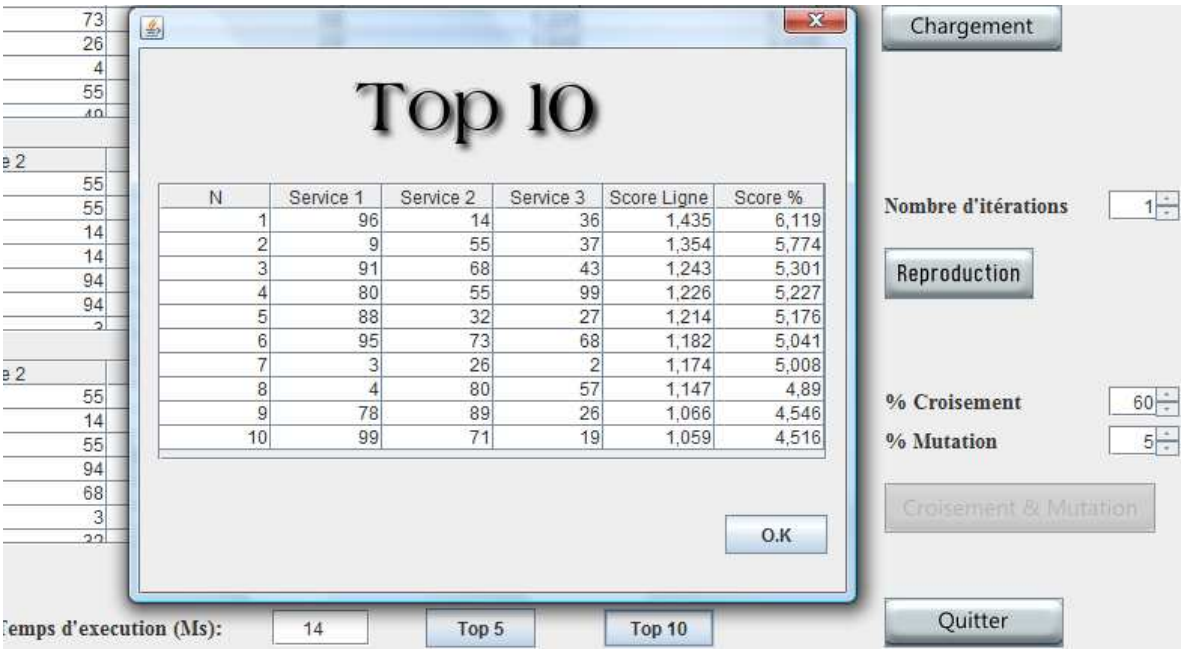

Figure III.6 Les résultats avec 1 itération, 60% croisement et 5% mutation

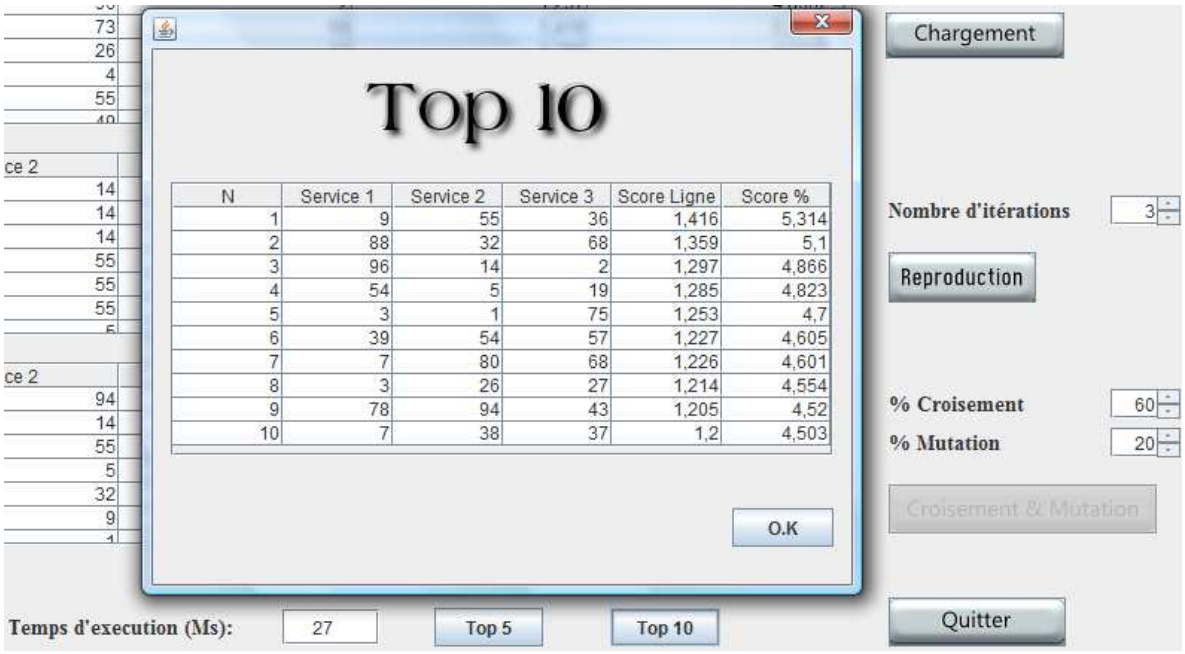

Figure III.7 Les résultats avec 3 itération, 60% croisement et 20% mutation

6. Discussion

D'après nos expériences on conclut les points suivants :

- Plus le nombre d'itérations est élevé mieux seront les résultats, jusqu'à se qu'on arrive a une certaine itération (itération4), les résultats à partir de cette itération seront les mêmes, on a donc une stabilité dans les résultats.

- Plus le pourcentage de croisement est élevé mieux seront les résultats, jusqu'à se qu'on arrive à 80%, les résultats à partir de se pourcentage de croisement seront les mêmes (stabilité).

- Plus le pourcentage de mutation est élevé mieux seront les résultats, jusqu'à se qu'on arrive à environ 18%, les résultats à partir de se pourcentage de mutation commencent a se dégradés.

D'après ces résultats on confirme l'efficacité des algorithmes génétiques

7. Conclusion

Dans ce chapitre nous avons proposé un exemple d'application pour illustré 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 obtenu, on a confirmé l'efficacité des algorithmes génétique.

Conclusion générale

Nous avons présenté dans ce mémoire les technologies liées aux services Web. Nous avons proposé aussi un algorithme de sélection qui se base sur les algorithmes génétiques pour l'optimisation des compositions.

Notre prototype selectionne les compositions de sérvices les plus satisfaisantes, en se basant sur 05 criteres de QOS : Latence, fiabilité, disponibilité, coût, reputation.

Comme perspective à ce travail, on propose de changer l'algorithme de sélection, pour cela, on peut utiliser alors

- Les colonies de fourmis
- Les colonies d'abeilles
- Spo
- ...

References

[1]: Bussler C "B2B Integration: Concepts and Architecture"; Springer-Verlag Berlin, 2003.

[2]: Chauvet J., ''Service Web avec SOAP, WSDL, UDDI, ebXML ''; jouve, Paris; mars 2002.

[3]: Chinnici R., **Gudgin M**., **Moreau J**. and **Weerawarana S**. "Web Services Description Language (WSDL) Version 1.2 ", W3C Working Draft January 2003,

[4]: Vialette M., Web services Communication inter langage, Version 2.0, Ecole supérieur d'Informatique de Paris, 8 mars 2006

[5]: Michel Van Caneghem, « Algorithme et Complexité 4», 2002

[6]: Qi Yu- Athman Bouguettaya, Foundation for Efficient Web Service Selection, 2010

[7]: Web services definition, www.laurenti.com/publis/SFT_Laurenti_04032004.PDF .

[8]: The Accredited Standards Committee (ASC) X12; Disponible sur: http://www.x12.org/

[9]: RosettaNet; Disponible à: http://www.rosettanet.org/

[10]: W3C Recommandation, www.w3.org/2002/07/**soap**-translation/**soap**12-part1.html.

[11]: W3C Recommandation, www.w3.org/2002/07/**soap**-translation/**soap**12-part0.html.

[12]: Site officiel d'UDDI: http://www.uddi.org/

[13]: Wikipedia® , http://fr.wikipedia.org/wiki/Service_web

[14]: http://www.w3.org/2001/XMLSchema

[15]: http://www.w3.org/2001/XMLSchema-instance

[16]: http://lsis.univ-tln.fr/~tollari/TER/AlgoGen1/node5.html#SECTION00051000000000000000

Résumé

Les services web permettent de connecter des applications différentes, l'utilité de cette technologie devient évidente et importante. Pour cette raison les activités de recherche et développement autour du sujet services Web ont un dynamisme très haut.

comme les services Web avec des fonctionnalités similaires sont censés être fournis par des fournisseurs concurrents, le défi majeur est de concevoir des stratégies d'optimisation pour trouver les meilleurs services Web respectant les exigences de l'utilisateur en terme de qualité.

Dans ce mémoire nous proposons un algorithme de sélection basé sur les algorithmes génétiques. Ces algorithmes appartiennent á la classe des algorithmes évolutionnaires et s'appuyant sur des techniques dérivées de la génétique et des mécanismes d'évolution de la nature : sélections, croisements, mutations…etc

Cette approche a été expérimentée afin d'évaluer les performances du prototype.

Mots clés : services Web, algorithmes de sélection, algorithme génétique, qualité de service, optimisation

Abstract

The usefulness of Web service technology has become evident and important, for this reason, the activities of research and development which endow Web services have been well flourished and dynamic.

As those Web service are fueled and mechanized by different providers which obey to the law of concurrence and market, we are obliged to conceive optimization strategies to select the best services, this will be the ultimate challenge which take into account user exigences, in terms of quality of service.

In this humble work, we propose a genetic based algorithm, in order to resolve with determinism this problem.

This approach has been experimented thanks to the need of the prototype performances validation

Keywords : Web service, selection algorithms, genetic algorithm, quality of service, optimization

ملخص

لقد أصبحت المنفعة المرجوة من تكنلوجيا خدمات الواب واضحة للعيان و من أجل ذلك زادت وتيرة البحث حول هذا الميدان.

و بما أن المؤسسات المانحة لهاته الخدمات أصبحت تشهد منافسة كبيرة فإنه أضحى لزاما علينا تصميم خطط لإنتقاء أفضل الخدمات المقدمة.

في هذه المذكرة، نقترح مقاربة إنتقاء نعتمد على الخوارزميات الوراثية.

لقد خضعت هذه المقاربة للتجربة من أجل تقييم أدائها ميدانيا.

الكلمات الرئيسية: خدمات الواب، خوارزميات الإختيار، الخوارزميات الوراثية، جودة الخدمة، التحسين