Sommaire

LISTE DES FIGURES

LISTE DES TABLEAUX

Résumé

L'augmentation continuelle des services web sur la toile mondiale crée de nouveaux défis pour le monde académique et industriel. En effet, la présence des services web similaires d'un point de vue fonctionnel mais différents d'un point de vue Qos, nous obligent à mettre en œuvre des techniques d'optimisation à fin de retenir les meilleures compositions de services. Dans ce travail, nous proposons une technique d'optimisation mono-objective basée sur la sélection clonale. Les résultats obtenus confirment l'efficacité des systèmes immunitaires artificiels dans le domaine de sélection de services web.

Mots clés: optimisation combinatoire, affectation sous contraintes, méta-heuristiques, système immunitaire artificiel, algorithmes évolutifs, sélection des services web.

INTRODUCTION GENERALE

I. Contexte

Le web évolue très rapidement, il n'est plus maintenant une source statique d'informations, mais un vrai réseau dynamique dans lequel les ressources sont partagées et l'information est produite sur demande.

Les organisations commerciales exploitent actuellement de plus en plus le web pour l'adaptation aux marchés à travers les applications B2B (intégration de Business-to-Business) et les applications B2C (Business-to-Consumer).

L'importance des standards dans ce contexte a sans doute accentué le phénomène. Mais il arrive très fréquemment que plusieurs services répondent à un même ensemble de besoins fonctionnels, d'où la nécessité de la naissance du domaine de la sélection de services Web a base de qualité de service.

II. Problématique

Notre projet a pour but d'assurer la sélection des services web à base de Qos (Quality of service) par exemple si on a une composition abstraite qui comporte n tâches et un ensemble de contraintes globales alors nous devons instancier ces n taches avec des services concrets de telle sorte à maximiser les critères positifs et minimiser les critères négatifs et satisfaire les contraintes globales de l'utilisateur.

III. Contribution

Pour arriver à ce but nous avons proposé dans ce travail un algorithme d'optimisation mono objectif qui se base sur les systèmes immunitaires artificiels et en particulier la sélection clonale.

IV. Plan du mémoire

Ce mémoire est constitué de quatre chapitres. Il est organisé comme suit :

Chapitre I

Nous avons présenté les concepts généraux des services web. Tout d'abord, nous commençons par l'architecture SOA (). Ensuite, définir et présenter les différentes architectures des services web. Après, nous détaillons les différents protocoles d'un service web sur lesquels il se repose. Enfin, nous citons quelques avantages et inconvénients des services web afin de conclure ce que nous avons fait.

Chapitre II

Nous avons présenté le problème de sélection avec un exemple de motivation, ensuite on a montré un état de l'art qui regroupe les différentes approches proposées pour résoudre ce problème.

Chapitre III

Nous avons présenté le SIN (Système Immunitaire Naturel) dans un premier grand titre: on a définit le SIN, donner ses caractéristiques, éléments et enfin ses théories afin d'achever les différentes théories et concepts nécessaires au développement d'un SIA (système immunitaire artificiel).

Chapitre IV

Il décrit la conception et la réalisation de notre application.

Chapitre I :

Les services web

I. Introduction

Ces dernières années, la recherche dans le domaine des services web a été très active. Une grande partie de cette recherche a été consacrée à la composition de services web. L'idée originale de « composition de service » n'est pas vraiment nouvelle dans la conception de logiciel. Elle peut être comparée au concept de bibliothèque exécutable dans beaucoup de langages de programmation. En appliquant cette idée au génie logiciel, il est possible de développer de nouvelles applications en utilisant des composants logiciels préexistants. Ce procédé est désigné par « composition » ou « réutilisation » de composants. Dans la communauté des services web, des efforts importants sont maintenant consacrés à ce problème, qui regroupe la découverte de services web, leur composition et leur exécution. [1]

II. L'architecture SOA

L'architecture **SOA** (**S**ervice **O**riented **A**rchitecture) fournit un ensemble de méthodes pour le développement et l'intégration de systèmes dont les fonctionnalités sont développées sous forme de services interopérables et indépendants. Une infrastructure SOA vise à permettre l'échange d'informations entre applications. Les concepts de SOA se basent sur des concepts plus anciens, en particulier l'informatique distribuée et la programmation modulaire. [2]

Afin de mettre en œuvre une architecture SOA avec succès, il ne suffit pas d'avoir les capacités technologies. Il convient également d'adopter un certain nombre de principes importants au niveau de la conception, du développement et de la gestion. Dans le cadre de SOA, ces principes constituent un frame work qui va permettre aux clients et aux services de collaborer et servir d'orientation pour la conception et le développement de services. Il existe de nombreux principes qui peuvent être appliqués à une SOA, on peut cependant dénoter quatre principes fondamentaux qu'il est nécessaire de respecter qui sont les suivants :

1. Couplage faible

Le couplage est une métrique indiquant le niveau d'interaction entre deux ou plusieurs composants logiciels. Deux composants sont dits couplés s'ils échangent de l'information. On parle de couplage fort (ou serré) si les composants échangent beaucoup d'information et de couplage faible (ou relaxé) dans le cas contraire. Dans une architecture SOA, le couplage entre les applications clientes et les services doit être faible. Cela signifie qu'il y a une séparation logique qui isole le client du service afin d'éviter une dépendance physique entre les deux. Pour ce faire, le client communique avec le service par échange de messages dans un format standard. Il est ainsi possible de modifier un service, par exemple pour y ajouter des fonctionnalités, sans briser la compatibilité avec les applications clients existantes. En cas de couplage fort, la possibilité d'évolution des services serait très limitée.

2. Interopérabilité

Tout comme le couplage faible, l'interopérabilité est un critère primordial pour mettre en œuvre une architecture SOA avec succès. L'interopérabilité se définie comme la capacité que possède un produit ou système, dont les interfaces sont intégralement connues, à fonctionner avec d'autres produits ou systèmes existants ou futurs. Il convient de distinguer la compatibilité et l'interopérabilité. En effet, la compatibilité est une notion verticale qui fait qu'un produit peut fonctionner dans un environnement donné en respectant ses caractéristiques. La notion d'interopérabilité est au contraire transversale qui permet à deux produits de communiquer de part une connaissance bilatérale de leur manière de fonctionner. L'interopérabilité est rendue possible par le respect de normes et formats par tout système ou produit. Dans le cas d'une architecture SOA, l'interopérabilité permet à des applications clients de communiquer avec des services programmés dans des langages de programmation différents et s'exécutant sur des plateformes (logicielles ou matérielles) différentes. Similairement au couplage faible, la communication par échange de messages joue ici un rôle très important. En effet, l'application ne connaît pas et n'a pas besoin de connaître le fonctionnement interne d'un service pour pouvoir l'utiliser. Il est simplement nécessaire de définir un format d'échange standard entre le client et le service. Le format XML est largement accepté et pris en charge pour l'encodage des messages. En effet, la plupart des

plateformes technologiques sont aptes à générer et à traiter des messages au format XML.

3. Réutilisabilité

Le principe de réutilisabilité permet de réduire les coûts de développement en favorisant la réutilisation de parties de codes qui ont déjà été réalisées. Dans le cas de SOA, l'objectif de la séparation des tâches en services autonomes est en effet de promouvoir leur réutilisation. Ainsi, lorsque les nouveaux clients définissent leurs besoins, il est généralement possible d'utiliser certains services existants pour satisfaire une partie des besoins.

Ceci permet un gain de temps considérable et une réduction importante de la taille des applications par évitement de la duplication. Ceci facilite également la maintenance et diminue les chances de bogues dans le programme. Dans une architecture SOA, la granularité du découpage des fonctionnalités en services est également importante. Il faut que celle-ci soit suffisante pour qu'un service soit significatif et utile.

4. Découverte

La découverte est une étape importante pour permettre la réutilisation des services. Il faut en effet être en mesure de trouver un service afin de savoir qu'il existe et pouvoir en faire usage. Ainsi, même si un service fournit une fonctionnalité importante, il serait très inefficace s'il n'était pas découvrable pour être réutilisé plus tard. Une solution typique pour ce genre de problème consiste à utiliser un annuaire de services. Un annuaire est très similaire à un catalogue ou un inventaire de services. L'annuaire contient des informations publiées concernant les services et fournit typiquement une possibilité de recherche basée sur une fonctionnalité recherchée. Il est alors nécessaire qu'un service pour SOA soit conçu pour être suffisamment descriptif pour qu'il soit facilement localisable et accessible via un mécanisme de découverte. Le développeur peut alors simplement consulter cet annuaire afin de trouver un service adéquat pour réaliser une tâche donnée et l'utiliser dans son code.

Dans la suite du chapitre, nous décrivons les technologies liées aux services Web.

III. Définitions

Définition 1 : Les services web sont des applications auto descriptives, modulaires et faiblement couplées qui fournissent un modèle simple de programmation et de déploiement d'application, base sur des normes, et s'exécutant a travers l'infrastructure web. Les services web réalisent des fonctions allant des simples requêtes aux processus métier sophistiqués. [3]

Définition 2: « A Web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-process able format (specifically WSDL). Other systems interact with the Web service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards. **»** [4]

IV. Architecture des services Web

1. Architecture de base

Le modèle des services Web repose sur une architecture orientée service. Celle-ci fait intervenir trois catégories d'acteurs : les fournisseurs de services (i.e. les entités responsables du service Web), les clients qui servent d'intermédiaires aux utilisateurs de services et les annuaires qui offrent aux fournisseurs la capacité de publier leurs services et aux clients le moyen de localiser leurs besoins en terme de services. La dynamique entre ces trois acteurs inclut donc les opérations de *publication*, de *recherche* et de *liens* (binding) d'opérations. Cette dynamique est normalisée à travers 3 standards : un protocole abstrait de description et de structuration des messages (SOAP), une spécification XML qui permet la publication et localisation des services dans les annuaires UDDI**,** et un format de description des services Web (WSDL). Un service WSDL est composé d'un ensemble d'opérations élémentaires, chacune décrite par un flux de messages échangés entre le client et le service**.** [5]

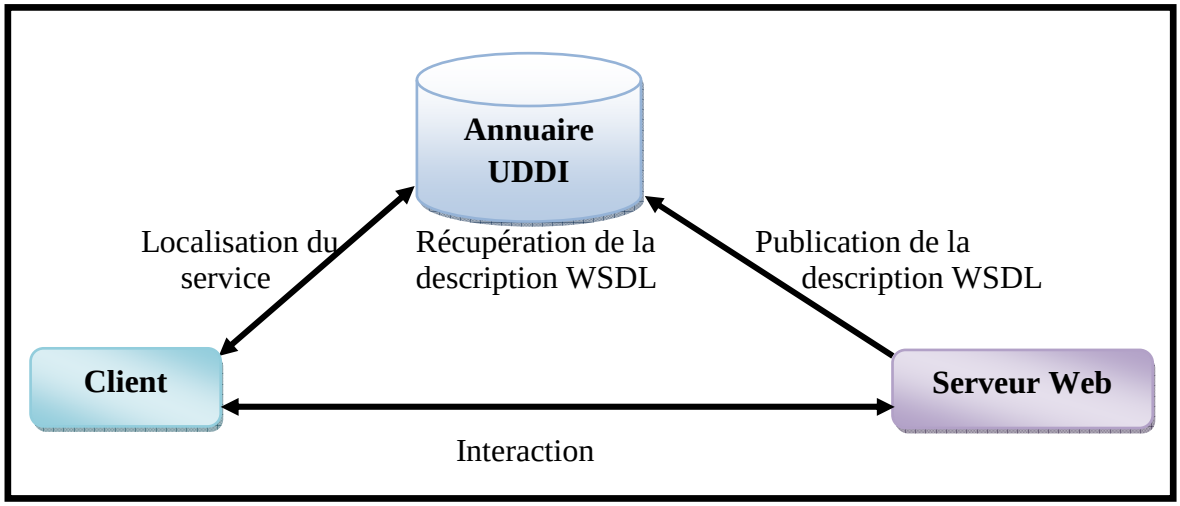


Figure I.1 Architecture de base d'un service web

Les services Web constituent un cadre robuste pour assurer l'interopérabilité entre des applications hétérogènes, accessibles en ligne, car ils proposent une représentation homogène du comportement observable du service (i.e. du point de vue du client).

Dans la suite de cette partie, nous allons présenter plus en détails SOAP, WSDL et UDDI qui constituent les langages de base des services Web.

2. Architecture en couche

La normalisation actuelle autour des Web Services est cependant une organisation complexe qui va bien au-delà de la simple invocation d'une méthode d'un objet distant. Différents travaux ont ainsi démarré pour permettre d'établir une véritable infrastructure distribuée, capable de satisfaire l'ensemble des besoins d'une application distribuée, aussi bien en termes de normalisation des échanges qu'en termes de services transverses. [6]

Cette organisation par comités de normalisation peut être schématisée selon le découpage matriciel suivant :

– Cette normalisation des services transverses se fait sur trois axes horizontaux :

- **Couche de transport :** Définition de la structure des messages utilisés par les applications pour se découvrir et dialoguer entre elles. Cette couche est à l'heure actuelle la seule réellement normalisée et qui ne souffre d'aucune contestation. Elle s'appuie sur le protocole SOAP pour l'échange des messages et sur le langage WSDL pour la définition du contrat de l'interface.

- **Couche de sémantique :** Normalisation des données participant aux échanges selon des critères métier. Les initiatives de définition de la couche de sémantiques des messages sont nombreuses et n'ont pour le moment pas conduit à une quelconque normalisation. Deux types d'organisation sont actuellement ouverts, l'une établie selon les différents corps de métier, l'autre suivant une approche plus globale autour de consortium tel qu'OASIS (initiateur d'ebXML) ou RosettaNet.
- **Couche de gestion des processus :** Standardisation de la gestion des processus métier qui s'étendent sur plusieurs applications disponibles sur Internet. L'orchestration de transactions B2B (Business to Business) complexes, fondée sur une architecture normalisée des messages est aussi une tentative qui n'avance pas assez rapidement et sur des standards non murs.

– Et trois axes verticaux :

- **Service d'annuaire :** Standardisation des moyens d'accès à un service à partir d'une requête portant sur le contenu d'un service ou sur un fournisseur. La première proposition d'annuaire UDDI aurait du apporter une solution définitive. Le constat est qu'il n'en est rien et que la trame, trop globale, du projet ne suffit pas à régler cette problématique d'échanges entre applications se connaissant. Une deuxième proposition d'annuaire, WS-Inspection, vient concurrencer celle-ci. Moins ambitieuse puisque consistant en une simple exposition, par agrégation, des services d'une application, elle est toutefois plus adaptée à cette seconde problématique.
- **Service de sécurité :** Normalisation des moyens permettant de couvrir les problématiques d'authentification et de gestion des droits d'accès. La gestion de la sécurité est actuellement le frein le plus important à la mise en place d'architectures distribuées à base de Web Services. Plusieurs organisations sont ouvertes mais aucune n'est réellement acceptée. Il semblerait que la norme XACML (eXtensible Access Control Markup Language) puisse supplanter SAML (Security Assertion Markup Language) et s'imposer à terme comme standard de sécurité.

- **Service de transaction :** Normalisation des moyens permettant de garantir l'intégrité des transactions longues impliquant plusieurs Web Services. Le problème reste le même que pour la sécurité. Les standards ne sont pas tout à fait établis. La lutte pour l'obtention d'une norme est beaucoup plus ouverte que pour celle de la sécurité, même si BTP (Business Transaction Protocol) semble plus soutenu actuellement.

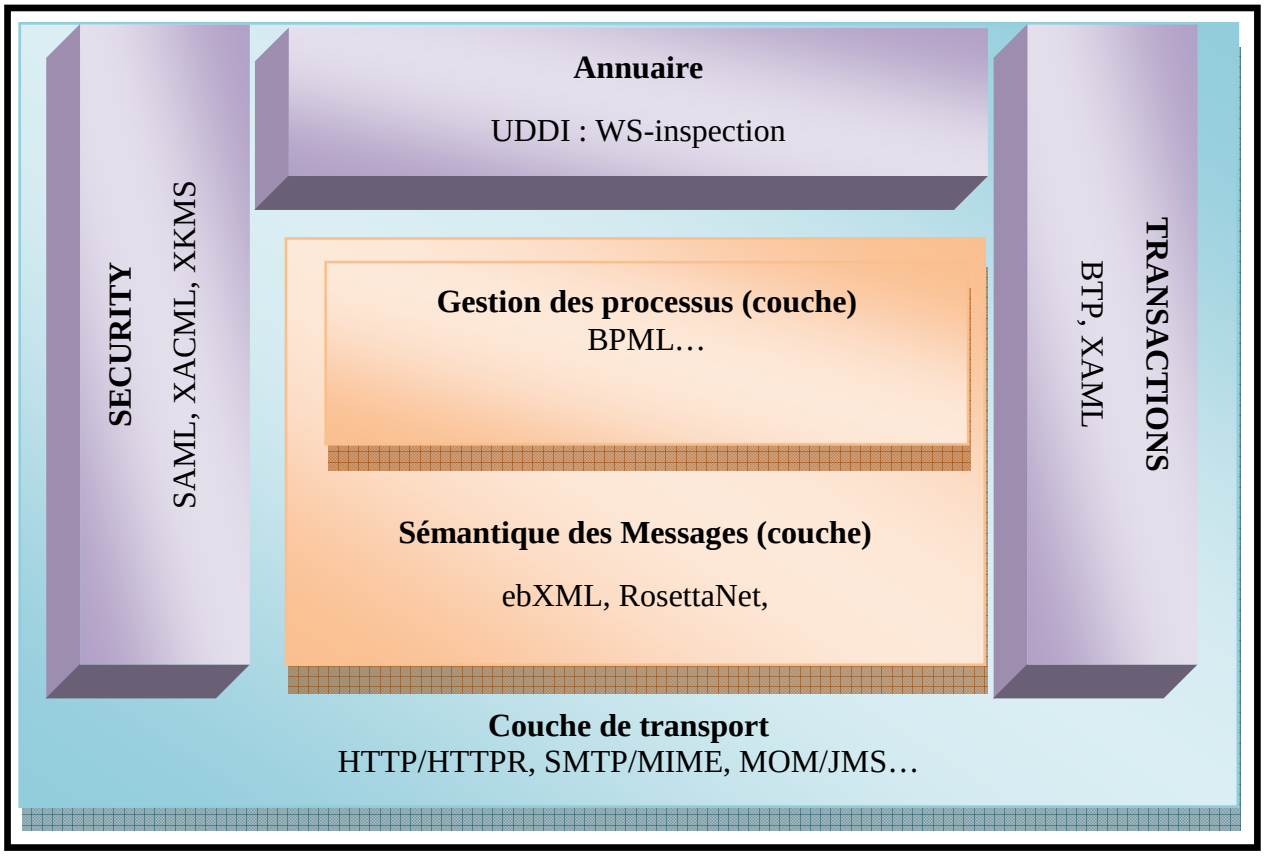


Figure I.2 Architecture en couche

Et puisque SOAP, WSDL, UDDI sont basés sur l'XML, on devrait avoir une connaissance sur l'XML, XML shemas et XML namespaces.

V. Cycle de vie

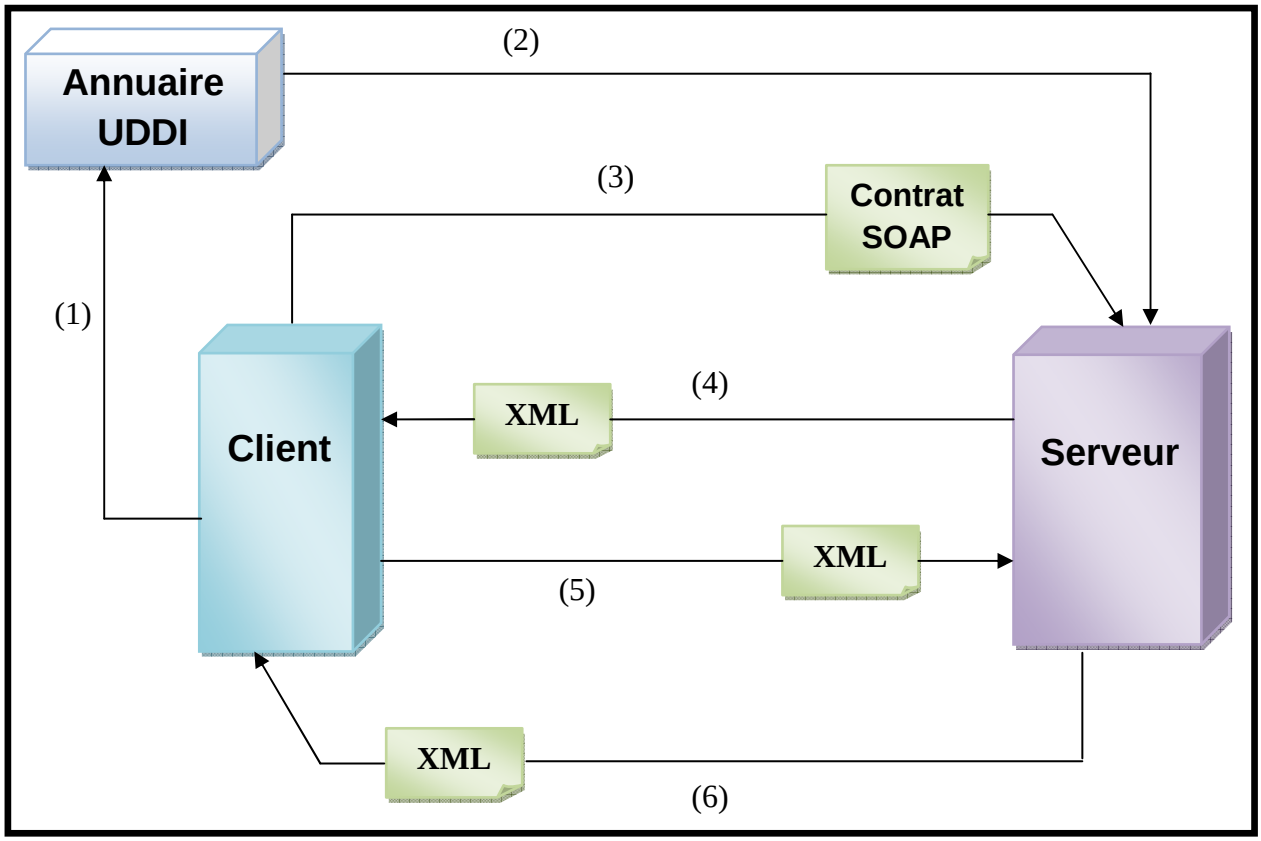


Figure I.3 Cycle de vie des Services Web

Dans un premier temps le fournisseur de service Web publie ses services web. Ensuite la Figure I.3 [6] explique les étapes du cycle de vie d'1 service web :

(1) Le client envoie une requête à l'annuaire de Service pour trouver le service Web dont il a besoin.

(2) L'annuaire a trouvé le service approprié, il envoie l'information du serveur qui l'héberge.

(3) Le client demande quel est le contrat du service web que tu proposes?

(4) Le serveur envoie sa réponse sous la forme établie par WSDL en langage XML.

(5) Le client appel le service web sous la forme établie par SOAP en langage XML.

(6) Le serveur envoie le résultat du service web sous la même forme normalisé.

VI. 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 il y a une description de ces technologies.

1. XML (eXtensible Markup Language)

XML est un format texte simple, très flexible tiré du SGML (l'ISO 8879). À l'origine conçu pour la publication électronique à grande échelle, XML joue aussi un rôle de plus en plus important dans l'échange d'une large variété de données sur le Web et ailleurs.

W3C recommande depuis 1998 XML en tant que standard de description de données. XML est un méta langage permettant d'identifier la structure d'un document. Un document est composé d'une définition de sa structure et d'un contenu. La structure d'un document XML est souvent représentée graphiquement comme un arbre. La racine du document constitue le sujet du document, et les feuilles sont les éléments de ce sujet. De ce fait, XML est alors flexible et extensible, et est devenu rapidement le standard d'échange de données sur le web. [7]

Nous allons prendre un exemple simple pour visualiser graphiquement quelle serait la hiérarchie du document XML correspondant et quelle est la structure du document. L'exemple choisi est celui d'un CD, élément racine du document, représenté par son titre, son artiste, sa société, le pays de la société, son prix et son année; tous les éléments du document XML. Le pays de la société est optionnel. L'artiste est lui même constituée d'un nom, un prénom et un site web. Le document XML devra suivre le schéma décrit ci-dessus. Cet exemple de document XML définit le CD "Tournée d'enfer", de l'artiste Renaud Létang, de la société française EMI MUSIC, son prix est 11 Euros et l'année est 2004.

Le schéma XML (nommé cd.xsd) en correspondance avec la définition est la suivante (cf. Extrait de code 1) :

```
<?xml version="1.0" encoding="UTF-8" ?>

<xs:schema xmlns : xs = "http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified" attributeFormDefault="unqualified">

<xs:element name= "CD">

      <xs:annotation>

            <xs:documentation>

            Un CD a un titre, une artiste, un pays, une société, un prix et
            une année

            </xs: documentation>

      </xs:annotation>

      <xs:complexType>

            <xs:sequence>

                  <xs:element name="titre" type="xs:string" />

                  <xs:element name="artiste">

                      <xs:complexType>

                        <xs:sequence>

                         <xs:element name="nom" type="xs:string" />

                         <xs:element name="prénom" type="xs:string" />

                         <xs:element name="siteweb" type="xs:string" />

                        </xs:sequence>

                     </xs:complexType>

                  </xs:element>

                  <xs:element name="société" type="xs:string" />

                  <xs:element name = "pays" type = "xs:string"minOccurs="0"

                  maxOccurs="unbounded"/>

                  <xs:element name="prix" type="xs:string" />

                  <xs:element name="année" type="xs:integer" />

            </xs:sequence>

      </xs:complexType>

</xs:element>

</xs:schema>
```

Extrait de code 1 Schéma XML: cd.xsd

Le fichier XML (nommé cd.xml) donné dans l'Extrait de code 2, instancie le CD défini au-dessus:

```
<CD>
<titre>Tournée d'enfer</titre>
      <artiste>
            <nom>Létang</nom>
            <prénom>Renaud</prénom>
            <siteweb>http://www.renaud.com</siteweb>
      </artiste>
      <société>EMI MUSIC</société>
      <pays>France</pays>
      <prix>11</prix>
      <year>2004</year>
</CD>
```

Extrait de code 2 Fichier XML: cd.xml

XML possède une galaxie de technologies. Parmi celles-ci nous pouvons citer XML Query (langage de requêtes) [8], XSL – eXtensible Stylesheet Language (définition de présentations de documents XML) [9], XSLT – XSL Transformations (langage permettant la transformation de documents XML) [9], XPath – XML Path (langage de requêtes permettant d'accéder à chaque élément d'un document XML) [10].

2. Le protocole SOAP (Simple Object Access Protocol)

2.1. Définition

SOAP est un protocole pour l'échange d'informations structurées avec les services Web, recommandé par le W3C. SOAP est le successeur de XML-RPC. Il est basé sur XML pour le format des messages et il s'appuie généralement sur des protocoles de la couche application pour le transport des messages (RPC, HTTP).

SOAP forme la couche inférieure de la pile des protocoles des services Web, fournissant un frame work pour l'échange de messages sur lequel les services Web peuvent se baser. [11]

Un message SOAP est structuré comme suit :(voir figure I.4).

2.2. Structure d'un message SOAP

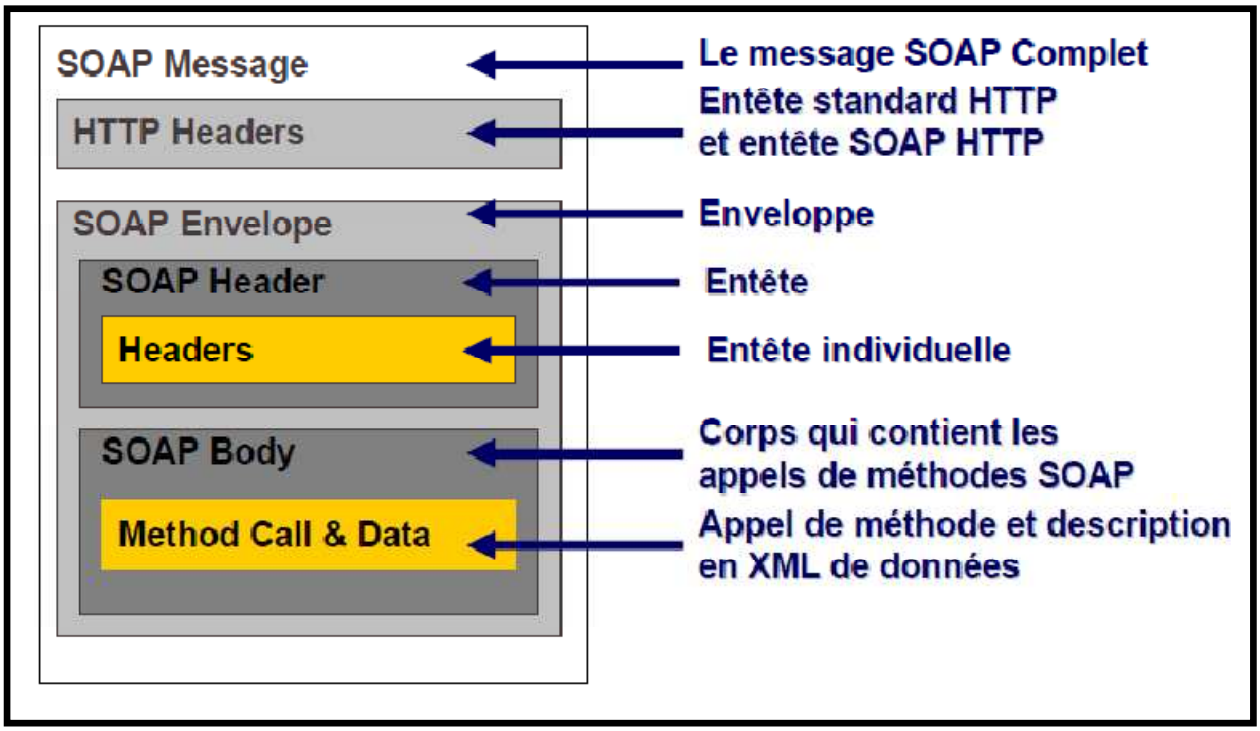


Figure I.4 Structure d'un message SOAP

- **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.

- **L'enveloppe SOAP**

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

- **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.

- **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.

2.3. Fonctionnement

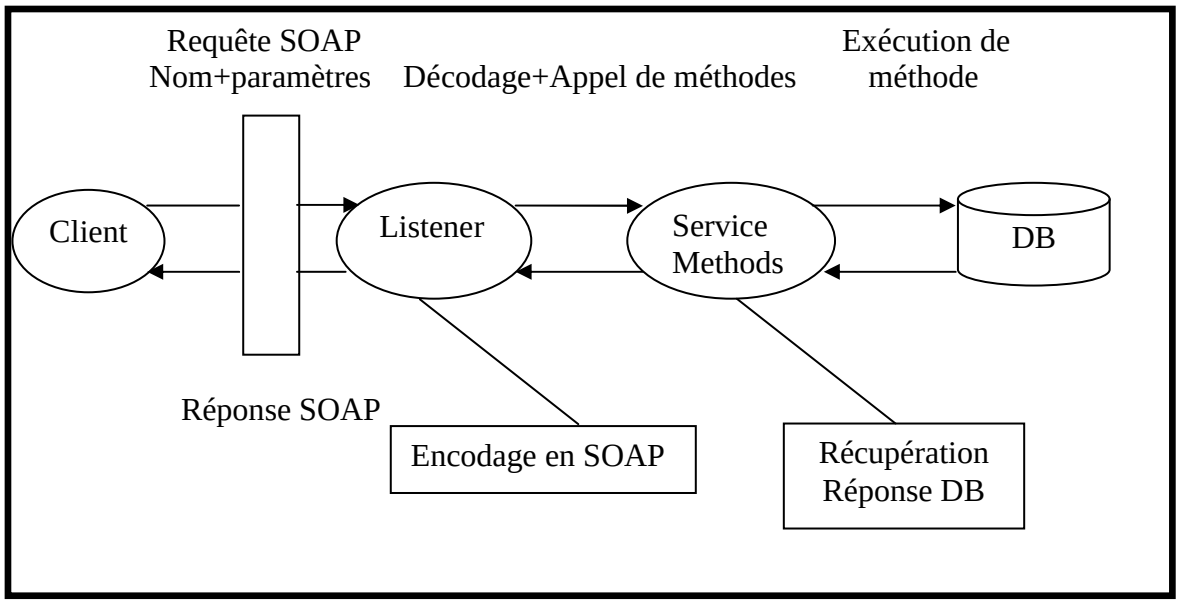


Figure I.5 Traitement d'un message SOAP

La requête SOAP transmise par le client à l'aide des protocoles de transmissions (exemple HTTP) passe par un Listener() que le permet d'adresser (cas de java) aux bibliothèques, ces bibliothèques donnée au client l'accède aux méthodes 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. [12]

2.4. Types de message SOAP

SOAP définit trois types de message

- Appel (Call) obligatoire
- Réponse (Response) optionnel
- Erreur (Fault) optionnel

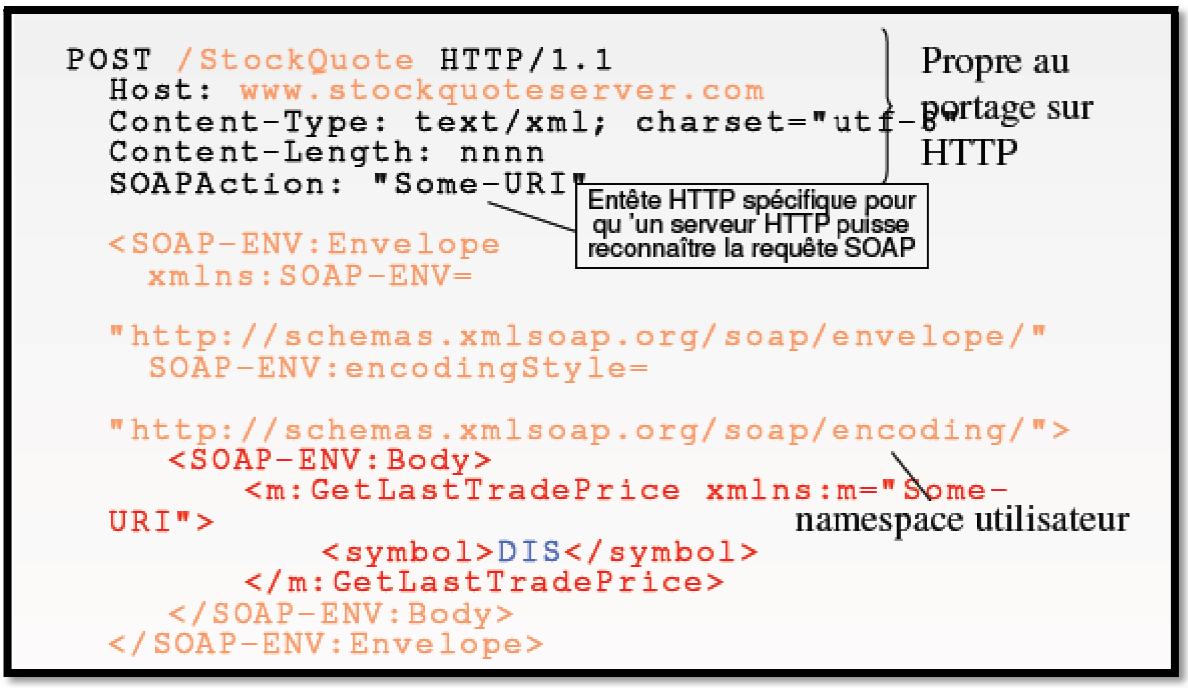
➢ **Requêtes / réponses SOAP**

- StockQuote est un ensemble de services qui permet d'obtenir des informations sur des actions boursières.

– GetLastTradePrice est le service qui permet de connaître la dernière valeur d'une action.

- Cet exemple présente un échange de messages entre un client qui veut savoir la valeur de l'action « DIS ».

Exemple de requête :



Exemple de repense :

```
HTTP/1.1 200 OK
  Content-Type: text/xml; charset="utf-8"
  Content-Length: nnnn

  <SOAP-ENV:Envelope
    xmlns:SOAP-ENV=

  "http://schemas.xmlsoap.org/soap/envelope/"
    SOAP-ENV:encodingStyle=

  "http://schemas.xmlsoap.org/soap/encoding/"/>
     <SOAP-ENV:Body>
         <m:GetLastTradePriceResponse
                 xmlns:m="Some-URI">
             <Price>34.5</Price>
         </m:GetLastTradePriceResponse>
     </SOAP-ENV:Body>
  </SOAP-ENV:Envelope>
```

- **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*.

```
<SOAP:Envelope xmlns:SOAP="urn:schemas-xmlsoap-
  org:soap.v1>
    <SOAP:Body>
        <SOAP:Fault>
            <faultcode>200</faultcode>
            <faultstring> SOAP Must Understand Error
  </faultstring>
            <runcode>1</runcode>
        </SOAP:Fault>
     <SOAP:Body>
</SOAP:Envelope>
```

La balise Fault est une balise permettant de signaler des cas d'erreur.

• La balise Fault contient les balises suivantes :

– faultcode : un code permettant d'identifier le type d'erreur

• Client, Server, VersionMismatch, MustUnderstand

– faultstring : une explication en langage naturel

– faultactor : une information identifiant l'initiateur de l'erreur

– detail : Définition précise de l'erreur

3. La technologie WSDL (Web Service Description Language)

3.1. Définition

« WSDL describes network services by using an XML grammar. It provides documentation for distributed systems and has the goal to enable applications to communicate with each other in an automated way. While SOAP specifies the communication between a requester and a provider, WSDL describes the services offered by the provider (an "endpoint") and might be used as a recipe to generate the proper SOAP messages to access the services. A WSDL document has a role similar to an IDL file in CORBA or the Remote Interface in a Java RMI implementation. » [13]

3.2. Structure d'un document WSDL

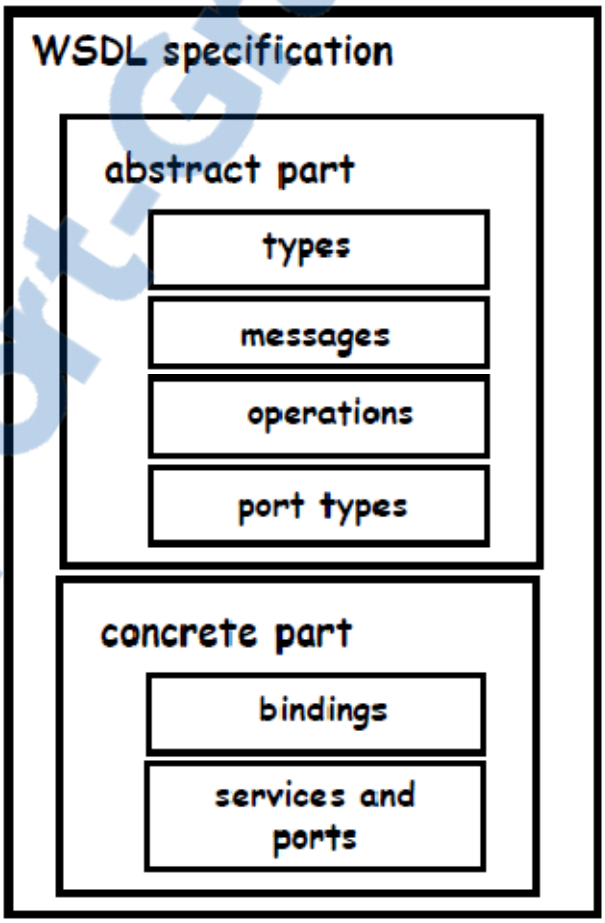
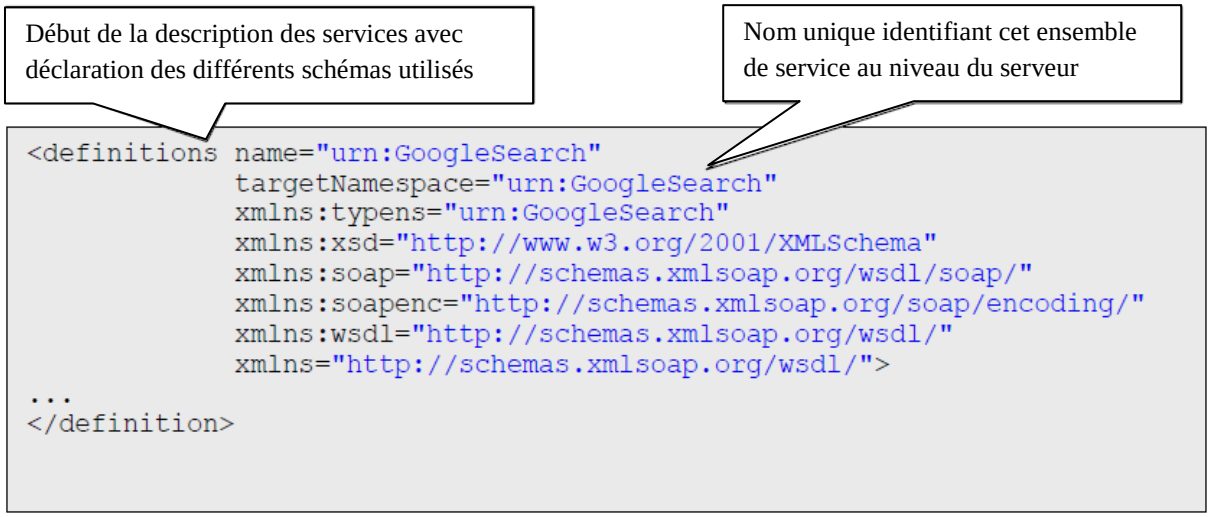


Figure I.6 Structure d'un document WSDL [14]

Un fichier WSDL commence par `<definition>`et finit par `</definition>`. C'est à l'intérieur de cette espace que l'on va déclarer tous les éléments constituant la description. Nous allons prendre l'exemple du Web Service de Google pour montrer comment se construit un fichier WSDL. [15]



a) Services : un service est la mise à disposition d'une ou plusieurs méthodes. On peut imaginer par exemple une classe avec plusieurs méthodes invocables à distance. La classe sera le service, et chaque méthode sera une opération sur ce service. La définition d'un service se fait par l'utilisation de `<service></service>`.

b) Port : pour chaque service on va définir des ports par lesquels ce service sera disponible. En effet, il est possible de rendre disponible un service sur plusieurs supports différents : HTTP GET, SOAP, SMTP…

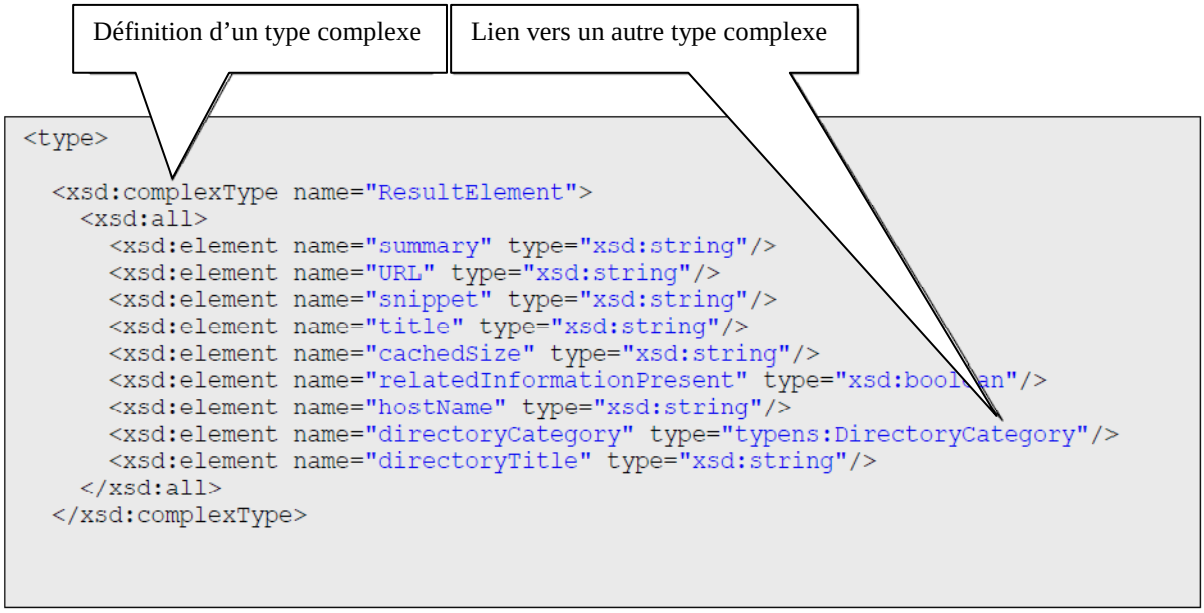
c) Message : les *messages* sont les éléments échangés entre le client et le serveur lors d'une *opération* sur le *port* d'un *service.* Ces messages sont constitués de plusieurs parties (*part*) qui représentent chacune une donnée avec un type associé.


```
<message name="doGoogleSearch">
    <part name="key"          type="xsd:string"/>
    <part name="q"            type="xsd:string"/>
    <part name="start"        type="xsd:int"/>
    <part name="maxResults"   type="xsd:int"/>
    <part name="filter"       type="xsd:boolean"/>
    <part name="restrict"     type="xsd:string"/>
    <part name="safeSearch"   type="xsd:boolean"/>
    <part name="lr"           type="xsd:string"/>
    <part name="ie"           type="xsd:string"/>
    <part name="oe"           type="xsd:string"/>
</message>

<message name="doGoogleSearchResponse">
    <part name="return"       type="typens:GoogleSearchResult"/>
</message>
```

Ainsi, il est possible définir ses propres types en se basant sur les types de base. En lisant cette définition dans le fichier WSDL, le client pourra générer éventuellement automatiquement une structure ou une classe reflétant cette description.

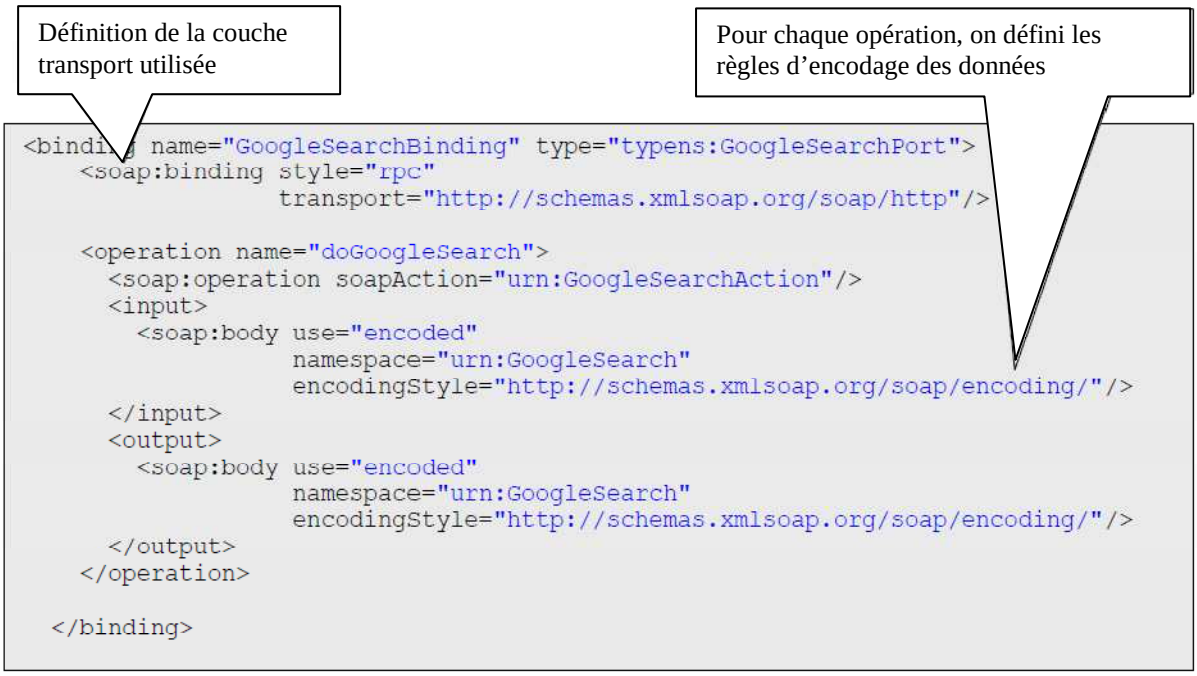
Voici un exemple de type complexe qui représente le résultat d'une recherche sur le Web service de Google™ :




```
<type>
  <xsd:complexType name="ResultElement">
    <xsd:all>
      <xsd:element name="summary" type="xsd:string"/>
      <xsd:element name="URL" type="xsd:string"/>
      <xsd:element name="snippet" type="xsd:string"/>
      <xsd:element name="title" type="xsd:string"/>
      <xsd:element name="cachedSize" type="xsd:string"/>
      <xsd:element name="relatedInformationPresent" type="xsd:boolean"/>
      <xsd:element name="hostName" type="xsd:string"/>
      <xsd:element name="directoryCategory" type="typens:DirectoryCategory"/>
      <xsd:element name="directoryTitle" type="xsd:string"/>
    </xsd:all>
  </xsd:complexType>
```

d) Port Type et Opérations : une méthode peut être représentée par une opération qui prend un message en entrée et renvoie un message en sortie. Ainsi chaque opération représentée par `<operation></operation>`indique les flux de messages en entrée et en sortie correspondants en définissant les éléments `<input/>`et `<output/>`. Enfin, la collection de toutes les opérations d'un service es assemblée dans un *port type.*

e) Bindings : enfin, pour compléter la description, il faut relier certains éléments entre eux. C'est le rôle des *bindings* représentés par les balises `<binding></binding>`. On va spécifier notamment tout ce qui concerne l'encodage des données dans les messages en indiquant les règles que l'on utilise.



4. UDDI - Universal Description Discovery and Integration

4.1. Définition

UDDI est un protocole basé sur XML qui sert à la publication et à l'exploration d'annuaires de Web Services. Il permet de situer un service Web, de le décrire, et de l'intégrer. Ce protocole a été réalisé par IBM, Microsoft et Ariba. Cependant UDDI est assez complexe et a été qualifié par certains « d'usine à gaz ». C'est la raison pour laquelle IBM et Microsoft travaille actuellement sur un autre protocole d'annuaire de services : Web Services Inspection (WS-Inspection). L'objectif de WS-Inspection est de permettre à une entreprise de décrire facilement les services Web dont elle dispose. [16]

4.2. La recherche d'un service Web

Pour simplifier la recherche de service, le standard UDDI offre trois catégories (Figure I.7) de classification pour les Services Web pour trois types de recherche différents

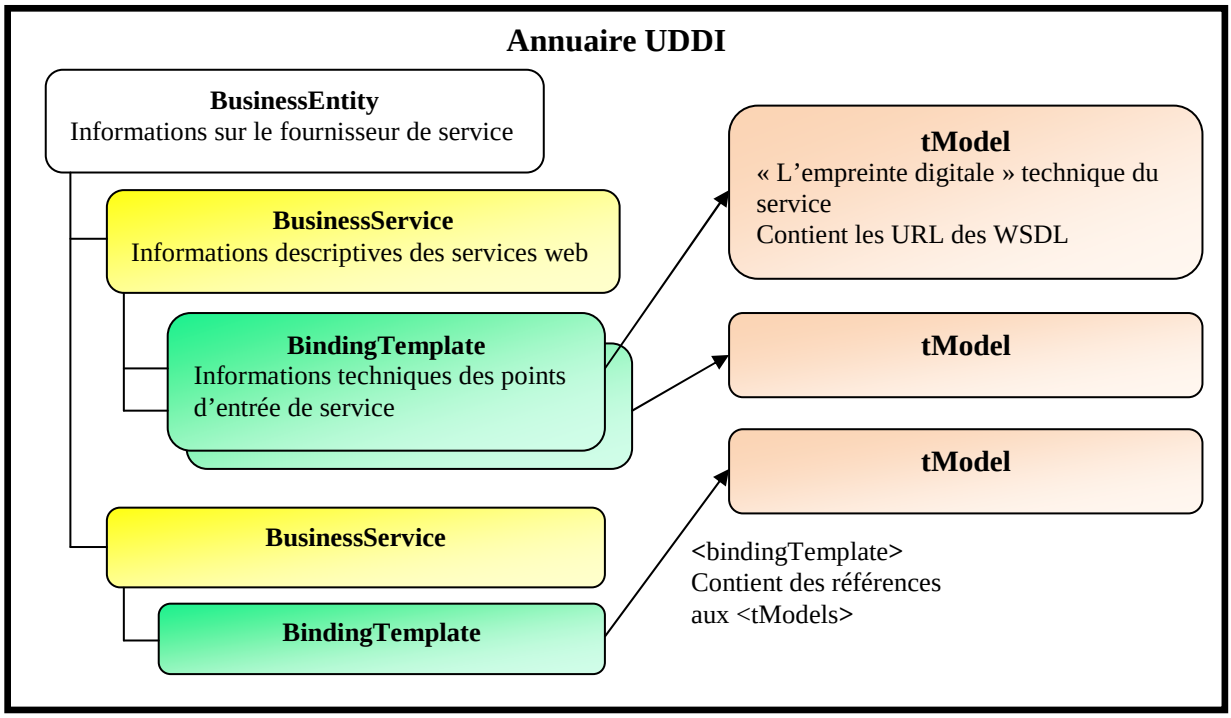


Figure I.7 Relation entre les structures de données UDDI

-pages blanches (BusinessEntity): contiennent des informations concernant les entreprises fournisseurs de service, la recherche se fait par contact, nom et adresse de fournisseur ;

- pages jaunes (BusinessService) : présentent les services selon leurs fonctionnalités en suivant une taxonomie industrielle standard ; le service est recherché par sujet et par domaine ;

- **pages vertes (BindingTemplate) :** offrent des informations techniques sur les services web ainsi que la référence de la description WSDL. Le service est recherché en fonction de ses caractéristiques techniques. [17]

4.3. La publication d'un service Web

L'enregistrement des services web dans un annuaire UDDI s'effectue auprès d'un opérateur en accédant au Site Web de ce dernier à partir d'un navigateur ou d'un outil intégré à un environnement de développement.

Grace à un jeu d'API XML basé sur SOAP, on peut interagir avec UDDI au moment de la conception d'exécution des S.W pour découvrir des données techniques et administratives sur les services et les entreprises qui publient, de manière que ces services puissant être invoquées et utilisés. UDDI repose sur le protocole de transport SOAP et assure que les requêtes et les réponses sont des objets UDDI envoyés sous forme de messages SOAP.

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). [18]

VII. Les avantages et les inconvénients des services Web

1. Avantages

L'utilisation des services web engendre plusieurs avantages dont on peut citer :

- ✓ **L'interopérabilité** : Les services Web fournissent l'interopérabilité entre divers logiciels fonctionnant sur diverses plates-formes.
- ✓ **La simplicité** : Les services web réduisent la complexité des branchements entre les participants. Cela se fait en ne créant la fonctionnalité qu'une seule fois plutôt qu'en obligeant tous les fournisseurs à reproduire la même fonctionnalité à chacun des clients selon le protocole de communication supporté.
- ✓ **Une composante logicielle légèrement couplée** : L'architecture modulaire des services Web, combinée au faible couplage des interfaces associées, permet l'utilisation et la réutilisation de services qui peuvent facilement être recombinés à différents autres applications.

- ✓ **L'hétérogénéité** : Les services web permettent d'ignorer l'hétérogénéité entre les différentes applications. En effet, ils décrivent comment transmettre un message (standardisé) entre deux applications, sans imposer comment construire ce message.
- ✓ **Auto-descriptivité** : Les services web ont la particularité d'être auto-descriptifs, c'est à dire capables de fournir des informations permettant de comprendre comment les manipuler. La capacité des services à se décrire par eux-mêmes permet d'envisager l'automatisation de l'intégration de services.
- ✓ Les outils de développement, s'appuyant sur ces standards, permettent la création automatique de programmes utilisant les services Web existants. [19]

2. Inconvénients

- Les services Web ont de faibles performances par rapport aux 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 miss en place à travers les firewalls. [20]
- La sémantique n'est pas prise en charge de façon efficace car le WSDL décrit les services de manière syntaxique.
- Les transferts reposent sur le XML, ce qui pose un problème sur la taille des fichiers échangés. Les fichiers XML sont le plus souvent de très gros fichiers et ceci entrainera une lourdeur considérable.
- Ils ne sont pas sécurisés à 100%.

VIII. 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.

Chapitre II

Sélection des services web

I. Introduction

Le domaine informatique a beaucoup évolué depuis ces dernières années. La taille et la complexité des logiciels sont en croissance continue. Cela est dû à des nouveaux besoins plus nombreux et plus complexes. Avec cette évolution, plusieurs approches ont vu le jour pour améliorer la productivité et l'efficacité des logiciels. Parmi ces approches, on peut citer l'approche à services, qui a une très grande popularité. L'émergence de l'approche à services a été accompagnée par une augmentation exponentielle du nombre de services et des fournisseurs de services. Dans cette mouvance, la sélection des meilleurs services représente un défi réel. Il faut aussi noter que la demande en dynamisme est devenue très forte avec l'apparition des services Web et des services liés aux équipements, où les services apparaissent et disparaissent à la volée. C'est ainsi que le besoin de méthodes avancées pour la sélection dynamique des meilleurs services, est plus grand que jamais. [21]

Pour résumer, nous considérons le problème de sélection de services comme un problème d'optimisation combinatoire qui sera détaillé dans la section V.

II. Exemple de motivation : (la création d'une entreprise)

Dans cette section, nous présentons un scénario dont l'objectif est d'illustrer et de motiver notre approche pour la sélection de services. Ce scénario concerne la création d'une entreprise, et pour cela nous avons besoins de 3 services (Figure II.1):

1/ Un service qui fournie les informations nécessaires pour la création d'une entreprise.

2/ Un service qui fournie des informations sur les concurrents du marché de même type d'activité.

Un service qui fournit des allocateurs pour l'hébergement de cette entreprise.

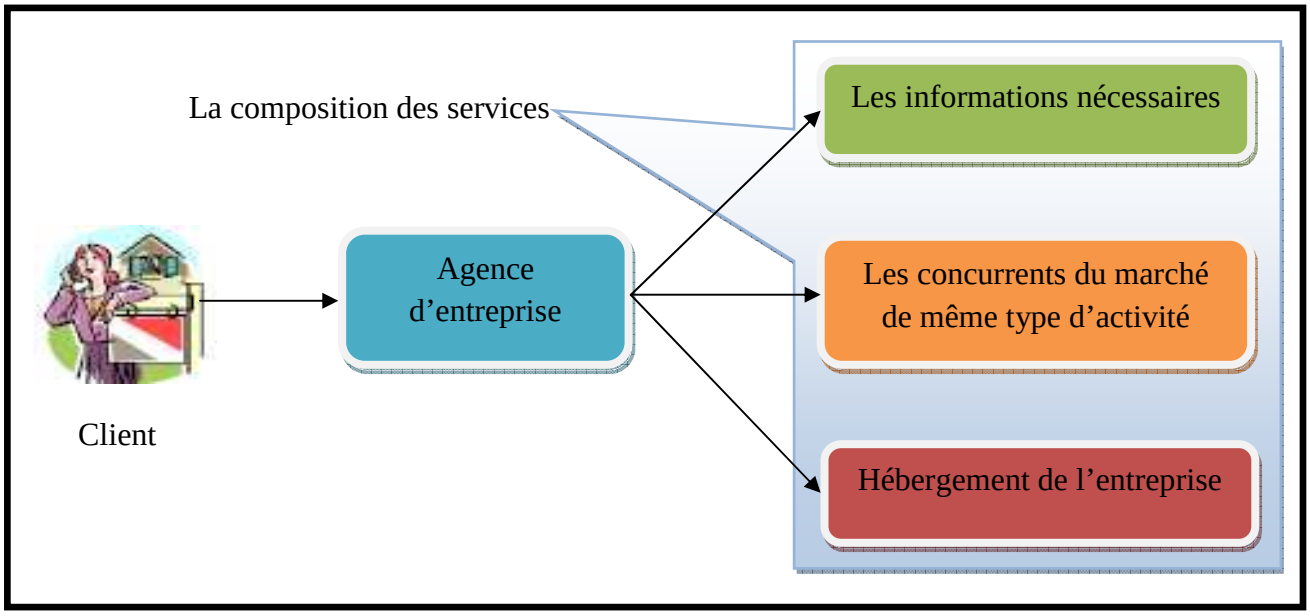


Figure II.1 Exemple de motivation

Chaque service est présenté par un ensemble de compagnie de telle sorte à respecter un ensemble de critères de qualité. Pour chaque critère, nous fournissons une définition, indiquons sa granularité, et fournissons des règles pour calculer sa valeur pour un service donné.

III. Critères de Qos (Quality of service)

Cinq critères de qualité génériques pour des services élémentaires sont considérés :

- ✓ **Latence :** Latence mesure le retard prévu en secondes entre le moment où l'exécution commence et le moment où l'exécution finit. La latence est calculé par:

$$\text{Latence (Sol)} = \sum_{i=1}^{n} lat\ (Ci)$$

- ✓ **Coût :** Le prix d'exécution est la somme d'argent qu'un demandeur de service doit payer pour exécuter une opération d'un service. Les allocataires de service annoncent directement le prix d'exécution de leurs opérations.

$$\text{Cout (Sol)} = \sum_{i=1}^{n} cout\ (Ci)$$

- ✓ **Réputation :** La réputation d'un service est une mesure de sa fidélité. Elle dépend principalement des expériences de l'utilisateur d'employer ce service.

Les différents utilisateurs peuvent avoir le service à peu près identique de différents avis. La valeur de la réputation est définie comme rang moyen indiqué au service par des utilisateurs.

$$\text{Rep (Sol)} = \frac{1}{n}\sum_{i=1}^{n} rep\ (Ci)$$

- ✓ **Disponibilité :** La disponibilité d'un service est la probabilité que le service est accessible.

$$\text{Disp (Sol)} = \log_2 \prod_{i=1}^{n} disp\ (Ci)$$

- ✓ **Fiabilité :** La fiabilité d'un service est la probabilité qu'une demande est correctement répondue dans le délai de temps prévu maximum (ce critère calcul le taux d'erreur). La fiabilité est une mesure liée à la configuration de matériel et/ou de logiciel des services et aux connexions réseau entre les demandeurs de service et les fournisseurs.

$$\text{Fiab (Sol)} = \log_2 \prod_{i=1}^{n} fiab\ (Ci)$$

On suppose que les domaines de valeurs des critères de Qos sont définit comme suit:

Critères	Domaine de valeurs
Cout	0..30 euros
Temps d'exécution	0..300s
Réputation	0..5
Disponibilité	0..1
Fiabilité	0..1

Tableau II.1 Les domaines de valeurs des critères

IV. La fonction objectif (score ou fitness)

Ces critères doivent optimiser une fonction globale qui maximise les critères positifs (Réputation, Disponibilité et Fiabilité) et minimise les critères négatifs (cout et temps d'exécution), et nous devons également satisfaire des contraintes globales par exemple :

- Cout (sol) < 200euros : Le cout d'une solution doit être mois de 200euros.
- Temps (sol) < 1000s : Le temps d'exécution doit être moins de 1000seconds.
- Rép (sol) > 3 : la réputation doit être supérieure à 3.
- Disp (sol) > 0.6 : la disponibilité doit être supérieure à 60%.

- Fiab (sol) > 0.6 : la fiabilité doit être supérieure à 60%.

Nous définissons une fonction score qui nous facilite la comparaison entre les qualités de chaque composition de web services :

Fctobj (Sol) = (Cout (Sol), Rep (Sol), Disp (Sol), Fiab (Sol), Temps (Sol))

$$= \sum_{Qi \in Neg} Wi \frac{Qi\max - Qi\,Sol}{Qi\max - Qi\,min} + \sum_{Qi \in Pos} Wi \frac{Qi\,Sol - Qi\,min}{Qi\max - Qi\,min}$$

Où $\boldsymbol{Wi} \in \mathfrak{R}^{+}_{0}$ et $\sum_{i=1}^{n} \boldsymbol{Wi} = \mathbf{1}$ est la somme des poids assignés pour chaque critère de qualité pour représenter les priorités de l'utilisateur.

V. Optimisation combinatoire

L'optimisation est une branche des mathématiques qui permet de résoudre des problèmes en déterminant le meilleur élément d'un ensemble selon certains critères prédéfinis. De ce fait, l'optimisation est omniprésente dans tous les domaines et évolue sans cesse depuis Euclide.

1. Définition

Un problème d'optimisation en général est défini par un espace de recherche *S* et une fonction objective *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 de cette section, nous aborderons les problèmes d'optimisation essentiellement sous l'aspect minimisation, maximiser une fonction *f* étant équivalent à minimiser -*f*. L'équation (1) résume la définition précédente.

$$S^* = \boldsymbol{min}\ (f(s) / s \in S) \qquad (1)$$

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-objectifs si plusieurs fonctions objectifs doivent être optimisées ou encore dynamique, si la *topologie* de *f* change au cours du temps. [22]

2. Approches d'optimisation combinatoire (approches de sélection)

Une grande quantité d'effort de recherches a été rapportée dans la littérature sur la sélection automatique de service [22]. Les travaux existants tombent dans deux types [23] :

1. **L'optimisation multi-objectif:** appelée aussi l'optimisation basée par horizon (the skyline based optimization) peut être manipulé en employant les techniques de base de donnée multiple [24]. Plus spécifiquement nous pouvons employer l'algorithme de clivage et conquérir, l'algorithme bitmap, l'algorithme basé par index (B tree, Hash table), et l'algorithme de plus proche voisin (R tree).

   D'ailleurs, il y a plusieurs travaux qui tient compte des préférences d'utilisateur pour choisir les k top horizons dominants [23 ; 24] certains d'entre eux utilisent la théorie des ensembles floues pour modeler les préférences et le rapport de dominance, les autres emploie le concept de Pareto-dominance pour ranger les services de Web.
2. **L'optimisation mono-objectif :** comporte plusieurs approches, elle peut employer un modèle de sélection global [25 ; 26 ; 27] ou un modèle de sélection local [28] ou un modèle de sélection hybride [22].
   - Le modèle de sélection global se concentre sur toute la composition (i.e. les n composants de la solution), il peut obtenir la solution optimale, mais il a une complexité exponentielle ;
   - Néanmoins, le modèle local a seulement une complexité linéaire mais ne peut pas manipuler les contraintes globales (il manipule seulement des contraintes locales).
   - Le troisième modèle est un compromis des deux approches, il commence la recherche par une optimisation globale, puis il continue le travail avec une optimisation local, sa complexité temporelle est inférieur que l'optimisation globale, et il peut également manipuler les contraintes globales.

Dans [29], les auteurs adoptent un algorithme génétique (sélection globale) pour sélectionner la composition optimale proche ; les résultats obtenus sont très satisfaites mais le temps de réponse est très élevé.

Zeng et al [30 ; 31] utilisent les techniques de programmation linéaire mélangées [32] pour trouver la sélection optimal des services composants. Près

de cet approche Ardagna et l'al, [26 ; 27] modifient le schéma de programmation linéaire pour inclure des contraintes locales. Les méthodes de programmation linéaire sont très robustes quand la taille du problème est petite, mais ces méthodes ne sont pas flexibles, à cause de sa complexité exponentielle de temps.

Notre contribution appartient à la catégorie mono-objectif et en particulier la sélection globale, et plus précisément la classe des méta-heuristiques.

La figure suivante montre l'état de l'art des différents travaux effectué sur le problème de sélection de service web.

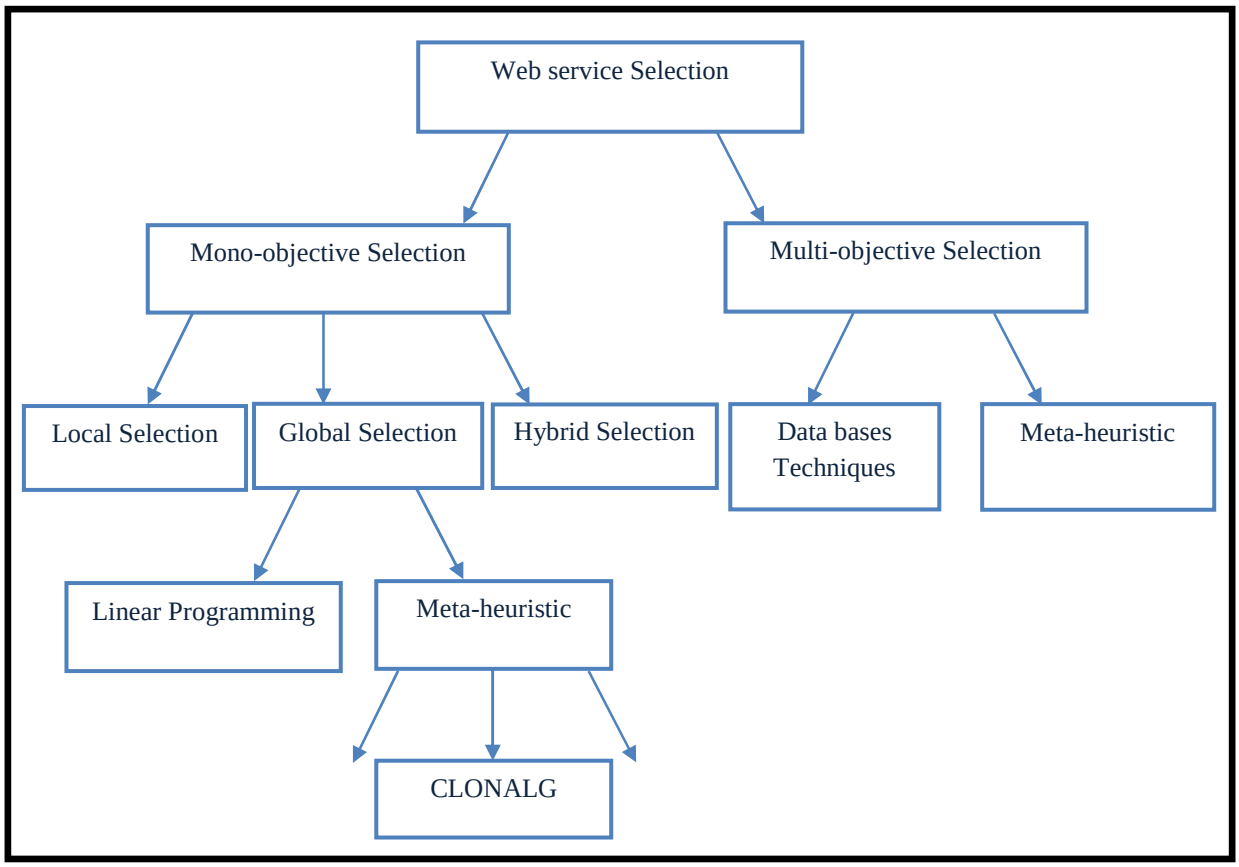


Figure II.2 Les approches de sélection

VI. Conclusion

Dans ce chapitre on a présenté le problème de sélection avec un exemple de motivation, ensuite on a montré un état de l'art qui regroupe les différentes approches proposées pour résoudre ce problème.

Dans le chapitre suivant nous proposons une méta-heuristique basée sur la sélection clonale afin de rechercher la meilleure composition qui satisfait l'ensemble des contraintes globales.

Chapitre III

Les Systèmes Immunitaires

I. Introduction

Dans le but de résoudre un problème complexe tel que la sélection des services web, des idées inspirées à partir des mécanismes naturels ont été exploitées pour développer des heuristiques inspirées de la nature. Le système immunitaire artificiel est un paradigme récent qui tente de capturer des caractéristiques intéressantes des systèmes immunitaires naturels, comme la mémorisation, la reconnaissance de formes, l'apprentissage, et les capacités d'adaptation, la détection d'intrusion dans les réseaux informatiques, la robotique, l'apprentissage machine, etc. [33]

Dans la section suivante, on va parler du système immunitaire naturel : ses types, ses caractéristiques, ses éléments et ses théories.

II. Le système immunitaire naturel (SIN)

Le système immunitaire est un système de défense remarquablement adaptatif. Il est capable de créer une très grande variété de cellules et de molécules susceptibles de reconnaître et d'éliminer spécifiquement un nombre pratiquement illimité d'envahisseurs étrangers. Ces cellules et ces molécules interviennent ensemble dans un réseau dynamique très précisément adaptable, dont la complexité rivalise avec celle du système nerveux. [34]

On distingue deux types principaux d'immunité naturelle, une innée et l'autre acquise, expliqué dans les points qui suit :

1. L'immunité innée (naturelle non spécifique ou naïve)

Est une immunité élémentaire qui est adaptée seulement à un certain nombre très réduit d'antigènes. On trouve ce type d'immunité chez les nouveaux nés pas encore vaccinés. Une immunité non adaptative pour longtemps peut conduire à des infections et à la mort car le corps n'est pas encore bien armé contre les antigènes de l'environnement [35].

2. L'immunité acquise (spécifique)

Est une immunité à mémoire (réponse secondaire), qui se développe lors de l'apparition du même antigène dans le même système immunitaire pour la deuxième fois ou plus, et qui engendre le développement et la génération des cellules B mémoire pour

ce type d'antigène déjà rencontré (mémorisé) dans le système. Cette réponse est plus rapide que celle innée [35]. Une réponse immunitaire engendre une augmentation de la température du corps, ce qui explique que les cellules B développées sont en train de lutter contre les antigènes introduits dans l'organe humain. La réponse immunitaire primaire est plus lente seulement mais elle garde les informations du passage des antigènes dans le système, le phénomène de mémorisation sera intéressant de l'utiliser pour la sélection des services web. De ce principe, le système immunitaire artificiel est développé dans ce sens.

3. Caractéristique du système immunitaire

Dans la section précédente, nous avons essayé de présenter le système immunitaire humain, Cette partie sera consacrée à une récapitulation des propriétés intéressantes du système immunitaire qui constituent, d'un point de vue informatique, une source d'inspiration très riche, qui sont [34]:

a) **Multicouche :** Le système immunitaire possède une architecture multicouche qui consiste en deux sous-systèmes inter-liés qui sont le système immunitaire inné et le système immunitaire adaptatif. Ces deux systèmes combinent leurs tâches et responsabilités pour assurer la protection et la sécurité globale.

b) **Unicité :** Chaque élément dans le système immunitaire assume des responsabilités particulières.

c) **Autonomie :** Le système immunitaire humain ne possède aucun contrôle central ou un gestionnaire particulier. Il possède une autonomie globale dans la détection et l'élimination des intrus.

d) **Distribution :** Les cellules immunitaires et les molécules sont distribuées dans le corps humain pour assurer sa protection. Il n'existe pas un point de contrôle centralisé.

e) **Parallélisme :** Le système immunitaire est capable de produire plusieurs réponses immunitaires en même temps à des endroits dispersés.

f) **Tolérance au soi :** Le système immunitaire humain peut différencier entre les cellules de soi et les cellules de non soi.

g) **Apprentissage :** Le système immunitaire augmente la capacité d'identification d'anticorps à un antigène sélectif (les réponses primaire et secondaire). Il apprend continuellement les structures de pathogènes.

h) **Adaptabilité :** Le système immunitaire humain permet la production des cellules de plus en plus spécialisées pour l'identification des antigènes. Cela est garanti par la théorie de la sélection clonale suivie par le mécanisme de l'hyper-mutation somatique.

i) **Dynamique :** Le système immunitaire change continuellement par la création de nouvelles cellules et molécules, l'élimination des cellules vieilles ou endommagées. Un bon exemple de la dynamique du système immunitaire est la théorie du réseau idiotypique.

j) **Diversité :** Le système immunitaire naturel est capable d'identifier n'importe quels intrus qui envahissent le corps. Le répertoire cellulaire est complet. Cette particularité est due à plusieurs mécanismes qui sont par exemple : l'hyper-mutation somatique, la reproduction de récepteur, l'identification approximative des intrus, etc.

k) **Mémorisation :** Après une réponse immunitaire à un antigène donné, un ensemble de cellules constituent l'ensemble de cellules mémoires qui seront dotées par une durée de vie longue afin de fournir des réponses immunitaires plus rapides et plus puissantes aux rencontres suivantes d'un même antigène.

l) **Coopération :** Les cellules immunitaires coopèrent leurs capacités pour assurer une meilleure détection et également une protection puissante par exemple les cellules T d'aide, les molécules MHC, etc.

m) **Détection :** Le système immunitaire est capable d'identifier et détecter les intrus dans le corps sans aucune connaissance antérieure de la structure de ces intrus.

4. Éléments du système immunitaire

Le système immunitaire est un système complexe composé de plusieurs éléments. Il est possible de classer ses éléments en deux classes : les organes, les cellules.

4.1. Les organes du système immunitaire

Les organes du système immunitaire sont nombreux et peuvent être divisé en deux classes:

- *Les organes lymphoïdes primaires :* ils constituent le site de développement et de maturation des cellules : [34]

  - *Moelle osseuse* : Lieu de développement des cellules pro-génitrice lymphoïde qui après leur croissance se diviseront pour former les

cellules précurseurs des lymphocytes B et T. Les cellules précurseurs des lymphocytes T quittent la moelle osseuse pour le thymus mais celles des lymphocytes B restent dans la moelle osseuse où elles continuent leur maturation.

- *Thymus* : Dans le bas du cou, constitue le site de maturation des lymphocytes T, Mais la plupart d'entre eux meurent sur place, seul 5 % quittent le thymus.

- *Vaisseaux lymphatiques* : Transportent la lymphe, les vaisseaux lymphatiques sont situés dans tout le corps.

➢ *Les organes lymphoïdes secondaires:* essentiellement les ganglions et la rate qui sont le lieu de l'interaction entre l'antigène (Ag) et le lymphocyte. [35]

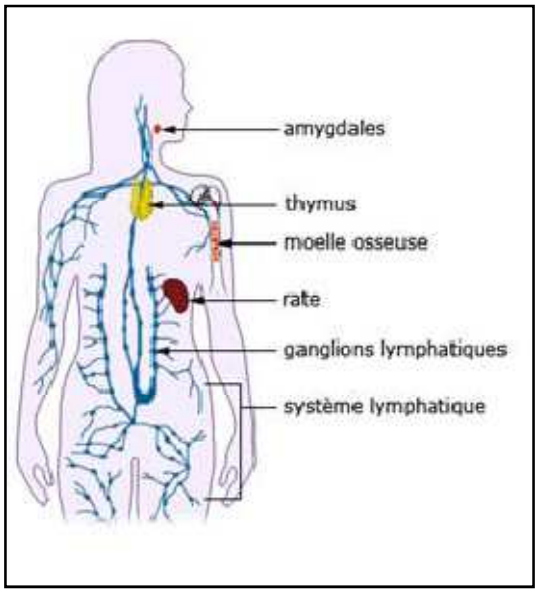


Figure III.1 Les organes du système immunitaire

4.2. Les cellules du système immunitaire

Les globules blancs aussi appelées leucocytes qui se trouvent dans le sang, la lymphe et les organes lymphoïdes sont les cellules responsable de protéger le corps des

organismes étrangers. Il existe différents types de leucocytes dont les lymphocytes B et T.

- Les **lymphocytes B** : la fonction principale des lymphocytes B est de produire des protéines appelé *anticorps* qui se fixent sur les protéines étrangères des antigènes. La partie de l'anticorps responsable de la reconnaissance de l'antigène est appelée *paratope*. Le paratope se lie à une partie spécifique de l'antigène appelée *épitope*. La liaison entre un paratope et un épitope est d'autant plus forte que leurs formes sont complémentaires. La force de cette liaison est appelée *affinité*. Les cellules B qui vont mieux reconnaître l'antigène vont proliférer en se clonant. Les lymphocytes B ont également la capacité de se comporter en cellule présentant le corps étranger (cellule mémoire).

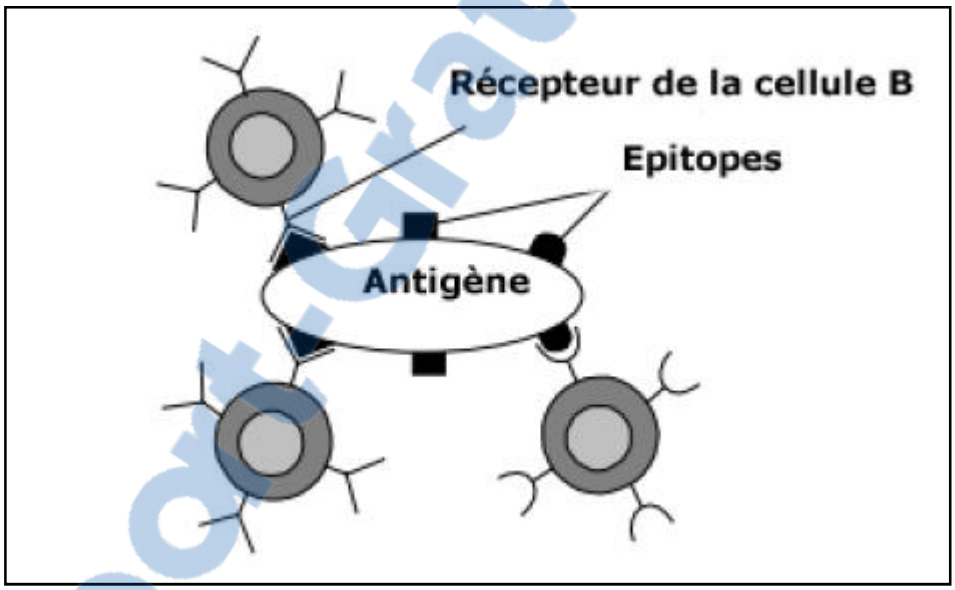


Figure III.2 Structure d'un antigène avec ses épitopes

- Les **lymphocytes T :** Il existe plusieurs types de lymphocytes T : [36]

  - Les lymphocytes *T cytotoxiques* qui, lorsqu'elles sont activées, détruisent directement les cellules infectées par des virus et les cellules tumorales.

  - Les lymphocytes *T auxiliaires*, qui se chargent de la coordination d'autres aspects de la réponse immunitaire c'est-à-dire elles déclenchent l'expansion clonale et stimulent ou suppriment la formation d'anticorps.

- Les lymphocytes T *suppresseurs* sont des régulateurs de l'immunité et luttent contre les réactions auto-immunes.

5. Théorie de la sélection clonal

Quand le corps est exposé à un antigène, les lymphocytes B produisent les anticorps. Chaque lymphocyte secrète un seule type d'anticorps, qui est relativement spécifique à l'antigène. Quand l'anticorps s'associe à l'antigène à l'aide des récepteurs (épitopes-paratopes) et reçois un signal des cellules telles que les T auxiliaires, les cellules B sont stimulées à proliférer (division cellulaire) et murir vers des cellules terminales (non divisibles) sécrétrices d'anticorps appelés ***cellules plasma***. Le processus de division cellulaire (mitose) génère des clones c'est-à-dire une ou un ensemble de cellules qui sont la progéniture d'une seule cellule. D'autre part, les cellules T jouent un rôle central dans la régulation de la réponse des lymphocytes B et sont par excellence dans les réponses immunitaires à médiation cellulaire. [37]

En plus de la prolifération, les lymphocytes subissent un second processus qui permettra de sélectionner parmi les nouvelles cellules celles présentant une grande affinité afin d'en faire des cellules B mémoires. Les cellules mémoires circulent dans le sang, la lymphe et les tissus, et lorsqu'elles sont exposées à un stimulus antigénique pour la seconde fois elles commencent à se différencier en lymphocytes capables de produire des anticorps de haute affinité, présélectionnées pour l'antigène spécifique qui a stimulé la réponse primaire. [37] Les principales caractéristiques de la théorie de la sélection clonale sont les suivantes :

- La prolifération et différenciation des cellules après avoir été stimulé par des antigènes ;
- La création de nouveaux changements génétiques aléatoires, représentant des profils d'anticorps différents, par une forme de mutation somatique accélérée (ce processus est appelé la maturation d'affinité) ;
- L'élimination des nouveaux lymphocytes qui portent des récepteurs de faible affinité antigénique.

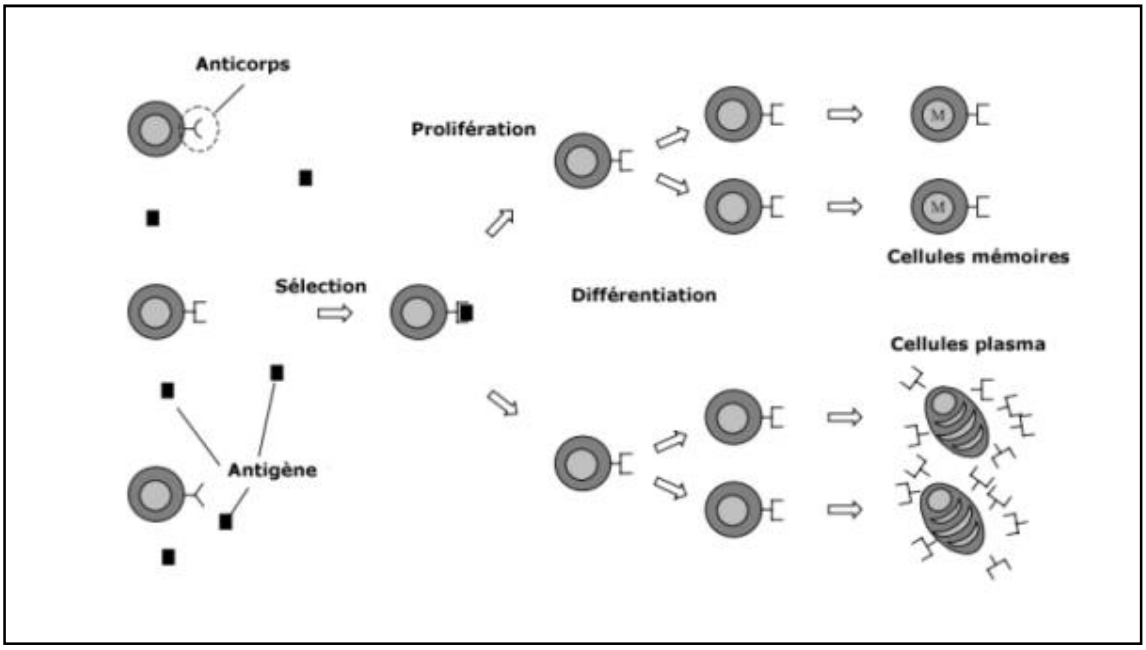


Figure III.3 Processus de sélection clonale

III. Le système immunitaire artificiel (SIA)

Le système immunitaire biologique possède la capacité pour protéger le corps humain contre une variété énorme de pathogènes étrangers. Dans les dernières années, un nombre de chercheurs ont étudié le succès et la compétence de ce système naturel et ont proposé le modèle immunitaire artificiel pour la résolution de divers problèmes. Des approches diverses ont été proposées pour mettre en œuvre les mécanismes de base du système immunitaire humain.

Le travail de De Castro, Von Zuben, Nicosia et de Cutello sur la sélection clonal est devenu notable en 2002. Le premier livre sur les systèmes immunitaires artificiels a été édité par Dasgupta en 1999.

Cette section sera consacrée à introduire le système immunitaire artificiel avec une présentation du modèle qui a été mis en œuvre.

1. Définitions

Définition 1 : « Un système immunitaire artificiel est un système informatique basé sur les métaphores du système immunitaire naturel. » [38]

Définition 2 : « Le système immunitaire artificiel est la composition de méthodologies intelligentes inspirées par le système immunitaire naturel afin de résoudre des problèmes du monde réel. » [38]

Les algorithmes des SIA peuvent être divisés en trois grandes catégories :

2. La sélection clonale

La théorie de la sélection clonale a été utilisée comme source d'inspiration pour le développement des SIA qui effectuent des tâches d'optimisation et de reconnaissance de formes. En particulier, l'inspiration a été prise du processus de maturation d'affinité des cellules B et de son mécanisme d'hyper-mutation. Ces SIA font souvent appel à l'idée de cellules mémoires afin de conserver les bonnes solutions du problème à résoudre. Dans leur livre, Castro et Timmis mettent en évidence deux aspects importants de la maturation d'affinité dans les cellules B qui peuvent être exploitées à partir du point de vue informatique:

- Le premier est que la prolifération des cellules B est proportionnelle à l'affinité de l'antigène auquel elles se sont liées, donc les cellules avec le plus grand taux d'affinité, sont celles qui produisent le plus de clones.
- En second lieu, les mutations subies par les anticorps d'une cellule-B sont inversement proportionnelles à l'affinité de l'antigène auquel ils se sont liés, donc plus l'affinité est grande, plus le taux de mutation est petit.

Utilisant ces deux caractéristiques, de Castro et Von Zuben ont développé l'un des algorithmes de SIA inspiré de la sélection clonale les plus populaires et largement utilisés appelé CLONALG, qui a été utilisé pour effectuer les tâches de filtrage et d'optimisation multimodale. [39]

Entrée : ***S*** = ensemble d'antigènes.

Sortie : ***M*** = ensemble de détecteurs de mémoire capable de classer de nouveaux modèles.

Début:

Créer un ensemble aléatoire d'anticorps, ***A***.

Pour chaque élément de ***S*** **faire**:

1. Déterminer l'affinité avec chaque anticorps de l'ensemble ***A***.
2. Créer un sous-ensemble ***K*** des anticorps avec le plus grand taux d'affinité.
3. Générer des clones des anticorps du sous-ensemble ***K***. Le nombre de clones pour un anticorps est proportionnel à son affinité
4. Appliquer le processus de mutation sur les clones pour augmenter leur degré de correspondance avec l'antigène.
5. Exposer les clones de nouveau à l'antigène et recalculer leurs affinités.
6. Les meilleurs clones seront placés dans l'ensemble ***M***.
7. Remplacer les **n** anticorps de plus faible affinité dans ***A*** par de nouveaux anticorps générés aléatoirement.

Fin

Bien que performant le CLONALG présente certains inconvénients, par exemple il ne permet pas de capitaliser les informations générées par chaque population de clone, car dès qu'une cellule mémoire est sélectionnée le reste des cellules mutées seront éliminées, alors que cette population pourrait contenir un certain nombre de cellules de haute affinité. En préservant une grande partie de la population de cellules mature, l'algorithme pourrait construire à partir d'une base solide de liens de haute affinité et devrait théoriquement aboutir à une solution optimale en moins de générations. Toutefois, ceci introduit le risque de convergence vers un minimum local. Ceci pourrait être évité en générant de façon aléatoire de nouveaux anticorps et les ajouter à la population.

3. La sélection négative

Les algorithmes de la sélection négative sont inspirés par le principal mécanisme dans le thymus qui produit un ensemble de cellules T matures capables de se lier seulement aux antigènes du non-soi.

Le premier algorithme de la sélection négative a été proposé par Forrest en 1994 pour détecter la manipulation de données causée par un virus dans un système informatique. Le point de départ de cet algorithme est de produire un ensemble de chaînes de soi, S, qui définissent l'état normal du système. La tâche est alors de générer un ensemble de détecteurs, D, qui ne se lient (reconnaissent) que le complément de S. Ces détecteurs peuvent ensuite être appliqués à de nouvelles données afin de les classer comme étant soi ou non-soi, ou dans le cas de l'œuvre originale de Forrest, détecter si les données ont été manipulées.

L'algorithme de Forrest produit l'ensemble de détecteurs via le processus suivant : [39]

Entrée : ***S*** = ensemble d'éléments du soi.

Sortie : ***D*** = ensemble de détecteurs généré.

Début

Répéter jusqu'à ce que les critères d'arrêt aient été atteints:

1. Générer aléatoirement des détecteurs potentiels et les placer dans un ensemble ***P***.
2. Déterminer l'affinité de chaque membre de ***P*** avec chaque membre de l'ensemble ***S***.
3. **Si** un élément de ***S*** est reconnu par un détecteur de ***P*** selon un seuil de reconnaissance **r**:

   Alors le détecteur est rejeté,

   Sinon il est ajouté à l'ensemble de détecteurs disponibles **D**.

Fin

4. Les réseaux immunitaires (ou idiotypiques)

La théorie du réseau immunitaire a suggéré un système immunitaire avec un comportement dynamique même en absence d'un antigène de non soi. Il existe plusieurs modèles du réseau immunitaire dans cette section nous nous pencherons sur le modèle nommé aiNet « artificial immune NETwork » qui a été proposé par De Castro & Von Zuben [40] [41]. Le modèle aiNet sera composé d'un ensemble d'anticorps, reliés entre eux par des liens avec leurs forces de connexion associé. Les anticorps aiNet sont censés représenter les images du réseau interne des agents pathogènes contenues dans l'environnement auquel ils sont exposés. Les connexions entre les anticorps déterminera leurs interrelations, fournissant un degré de similitude entre eux: plus les anticorps sont proche, plus ils sont similaires.

L'algorithme de base est le suivant : [39]

Entrée : ***S*** = ensemble d'antigène à reconnaitre, **nt** seuil d'affinité réseau, **ct** seuil piscine clonale, **h** le nombre de clones avec la plus grande affinité, **a** le nombre de nouveaux anticorps à introduire.

Sortie : ***N*** = ensemble de détecteurs mémoire capable de classer les modèles non vue.

Début

1. Créer un premier ensemble aléatoire d'anticorps, N.
2. **Répéter** jusqu'à ce qu'un des critères d'arrêt soit atteint

   Pour chaque individu de ***S*** **faire**

   a) Déterminer l'affinité avec chaque anticorps en **N**.

   b) Générer des clones des anticorps avec la plus haute affinité.

   c) Muter les attributs de ces clones, Sélectionner quelques clones de plus haute affinité pour constituer l'ensemble mémoire, **C** ;

   d) Éliminez tous les éléments de **C** dont l'affinité avec l'antigène est inférieur à un seuil prédéfini **ct ;**

   e) Déterminer l'affinité entre tous les anticorps en **C** et éliminer les anticorps dont l'affinité avec les autres est inférieur au seuil **ct** ;

   f) Incorporer les clones restants de **C** dans **N** ;

   Fin

3. Déterminer l'affinité entre chaque paire d'anticorps dans **N** et éliminer tous les anticorps dont l'affinité est inférieur au seuil **nt** ;
4. Introduire un nombre **a** d'anticorps générés aléatoirement et les placer dans **N** ;

Fin

Fin

5. Domaines d'application des SIAs

Le système immunitaire artificiel s'inspire du système immunitaire naturel et chaque processus du ce système sert de base pour un modèle différent. Cette diversité de modèle permet que les SIAs soient utilisés pour résoudre plusieurs problèmes différents tel que : [42]

5.1. Robotique

L'une des tâches les plus difficiles de la robotique est le problème de *navigation autonome*, où un robot (un ensemble de robots) doit pouvoir accomplir certaines tâches sans aucune indication extérieure. Les articles *"A Robot with a Decentralized Consensus-Making Mechanism Based on the Immune System"* (1997) et *"Immunoid: A Robot with a Decentralized Behavior Arbitration Mechanisms Based on the Immune System"* (1996) de Ishiguro et al. portaient sur l'élaboration d'un mécanisme dynamique consensus de décision décentralisé, fondé sur la théorie des réseaux immunitaires où *l'intelligence* devait émerger des interactions réciproques entre les agents (appelé modules de compétence) ou entre un robot et son environnement. Leur but était la construction d'un mécanisme qui peut prendre les décisions appropriées entre plusieurs modules et la façon de préparer ces modules. Cette méthode a été évaluée sur un problème de collecte des ordures en tenant compte de l'autonomie, c'est à dire un robot devait recueillir un ensemble d'ordures et le mettre dans une poubelle, sans manquer d'énergie (niveau de batterie).

5.2. Optimisation

Les problèmes d'optimisation sont présents dans plusieurs domaines d'applications. Le but dans ce type de problème est de trouver l'ensemble des meilleures conditions admissibles pour atteindre un certain objectif.

En 2000, De Castro & Von Zuben ont présenté un algorithme de sélection clonale, qui prend en compte la maturation de l'affinité de la réponse immunitaire, afin de résoudre des problèmes complexes, comme l'apprentissage et l'optimisation multimodal. Leur algorithme constitue une mise en oeuvre des processus biologiques et ne tient pas compte de toute sophistication mathématique pour améliorer ses performances dans des tâches particulières.

5.3. Sécurité des ordinateurs

La protection des ordinateurs contre les virus, les utilisateurs non autorisés, etc., constitue un champ riche de la recherche pour les systèmes de détection d'anomalie. En 1994, Forrest et al. dans leur article *"Self Non-self Discrimination in a Computer"* comparent le problème de la protection des systèmes informatiques à celui de l'apprentissage de la distinction entre soi et le non-soi des systèmes immunitaires. Ils ont décrit une stratégie de détection basée sur la *sélection négative* intrinsèque à notre système immunitaire.

Et il y on a plusieurs autres domaines d’applications comme :

- ✓ La détection et l'élimination des virus informatiques,
- ✓ Reconnaissance de formes,
- ✓ Diagnostic médical,
- ✓ Segmentation d'images,
- ✓ Apprentissage,
- ✓ Ordonnancement,
- ✓ Data mining,
- ✓ Système de classification,
- ✓ …

IV. Conclusion

Dans ce chapitre nous avons vu que les systèmes immunitaires artificiels sont des algorithmes qui s'inspirent du domaine de la biologie plus précisément les systèmes immunitaires des vertébrés. Ce chapitre était destiné à présenter les différentes théories et concepts nécessaires au développement d'un SIA. Il faut noter que les SIA sont particulièrement intéressantes pour résoudre les problèmes dans des environnements distribués et dynamiques.

Dans le chapitre suivant nous allons utiliser l’algorithme à base de sélection clonale pour la résolution du problème de sélection de service.

CHAPITRE IV :

CONCEPTION ET IMPLÉMENTATION DU PROTOTYPE

I. 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 (QoS). [24]

Nous avons choisis l’algorithme de sélection clonale comme algorithme d’optimisation. Dans la suite de ce chapitre, nous détaillons la configuration et le paramétrage utilisé.

II. Description de la base

1. Description de la base

Nous avons créé un schéma de service qui contient dix (10) classes de services web. Le nombre d'instances dans chaque classe est quarante (40) fournisseurs de services web. Chaque fournisseur se caractérise par des qualités de service. Nous utilisons cinq (5) paramètres pour évaluer les opérations des services : latence, fiabilité, disponibilité, coût et réputation. Les valeurs de ces paramètres sont générées en se basant sur une distribution uniforme. Un échantillon de notre base est illustré dans la table (IV.1) qui représente le fournisseur i des cinq premières classes.

Attributs	C1	C2	C3	C4	C5
Coût	-22.4	-26.7	-29.6	-.02	-6.7
Latence	-223.1	-222.7	-255.7	-3.5	-3.2
Disponibilité	-0.4	-0.1	-0.5	-0.3	-0.08
Fiabilité	-0.1	-0.08	-0.6	-0.3	-0.01
Réputation	0.6	3.7	2.3	2.03	1.3

Table IV.1 Un exemple de base de données

2. Description de la requête

La requête de l'utilisateur comporte cinq (05) éléments :

1. La borne minimale pour la fiabilité.
2. La borne minimale pour la disponibilité.
3. La borne minimale pour la réputation.
4. La borne maximale pour le cout.
5. La borne maximale pour latence.

III. Conception

1. Diagramme de cas d'utilisation

Un diagramme de cas d'utilisation permet de structurer les besoins des utilisateurs et les objectifs correspondants d'un système. Il permet aussi d'identifier les possibilités d'interactions entre le système et les acteurs (intervenants extérieurs au système). Il part du principe que les objectifs du système sont tous motivés. La figure (IV.1) représente le diagramme de cas d'utilisation de notre application.

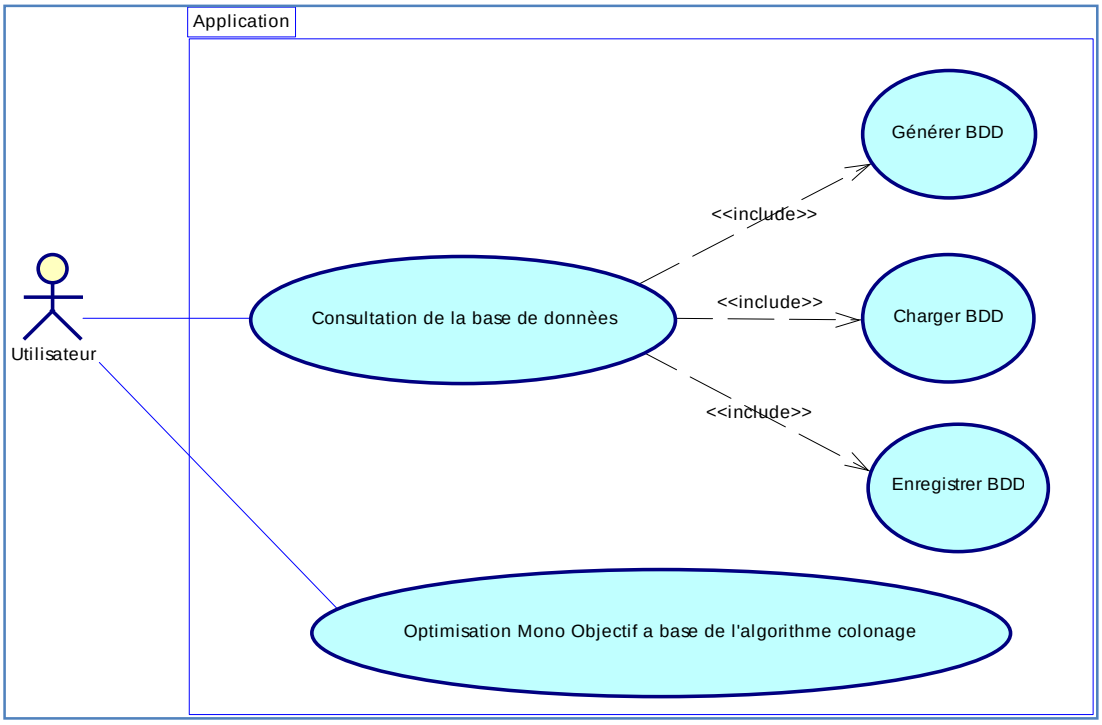


Figure IV.1 Diagramme de cas d'utilisation

2. Diagramme de séquence

Un diagramme de séquence permet de représenter des collaborations entre objets selon un point de vue temporel. Les objets communiquent en échangeant des messages représentés sous forme de flèches. Il peut servir à illustrer un cas d'utilisation. Un exemple de diagramme de séquence de notre application est représenté dans la figure IV.2.

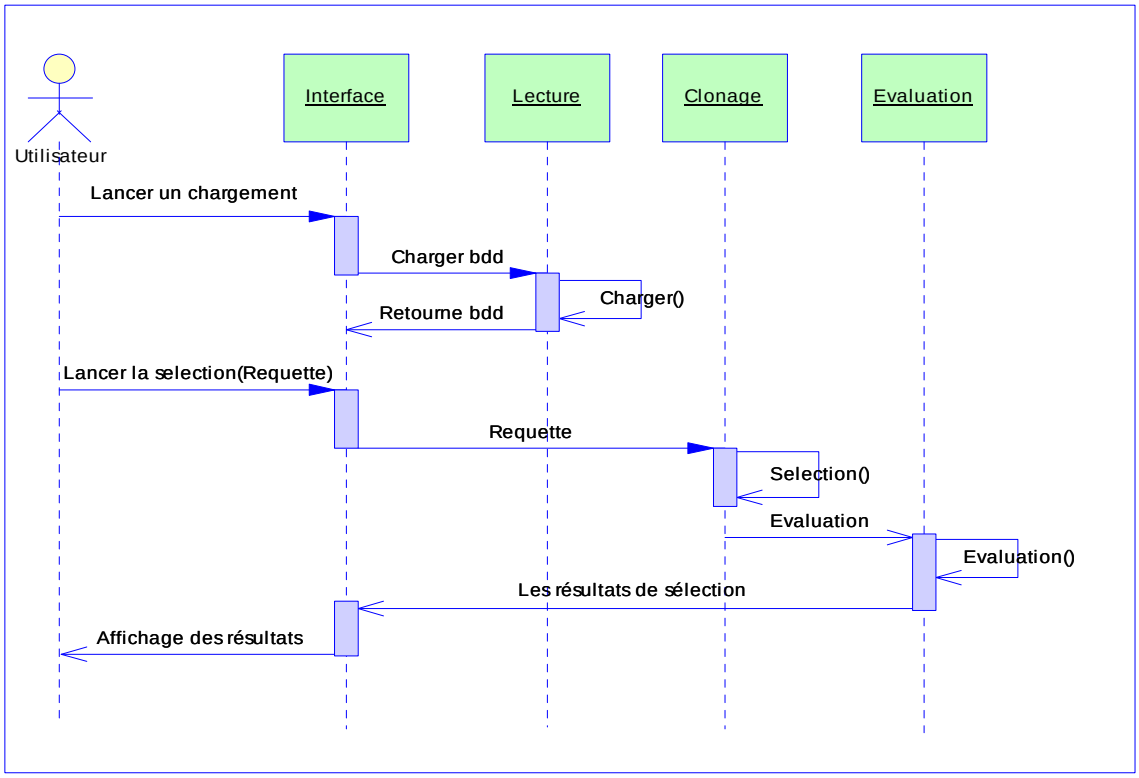


Figure IV.2 Diagramme de séquence

3. Diagramme de classe

Un diagramme de classes permet de représenter les classes intervenant dans le système. Il constitue un élément très important de la modélisation et permettre de définir quelles seront les composantes du système final. Il permet de structurer le travail de développement de manière très efficace. La figure IV.3 représente le diagramme de classes de notre application.

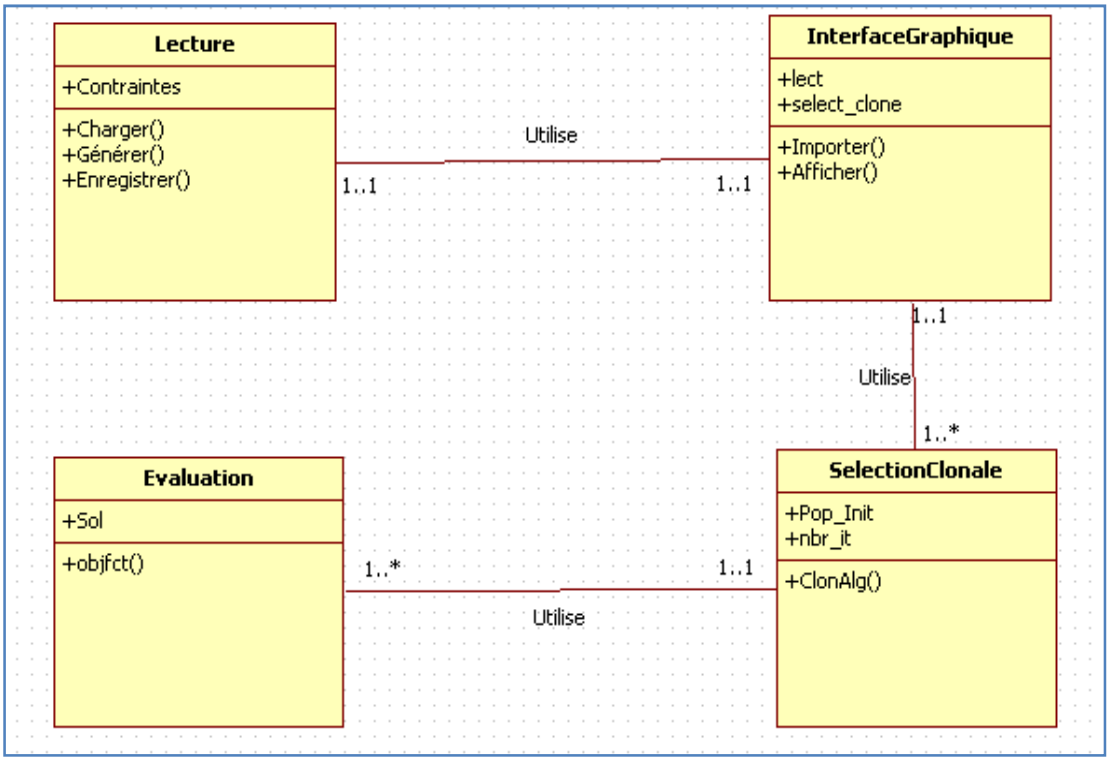


Figure IV.3 Diagramme de classe

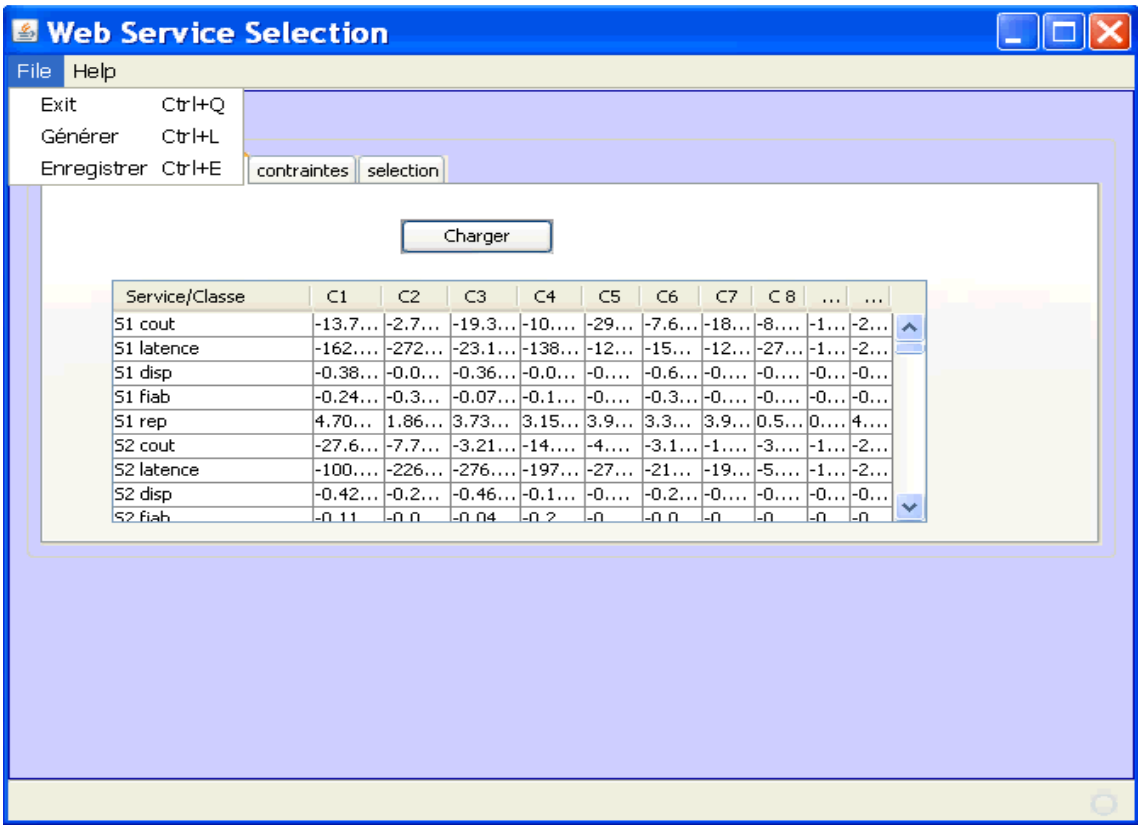
IV. Interface Humain/Machine (IHM)

L'IHM représente l'élément clé dans l'utilisation de tout système informatique. Les interfaces de notre système de recherche sont conçues de manière à être simples, naturelles, compréhensible et d'utilisation faciles.

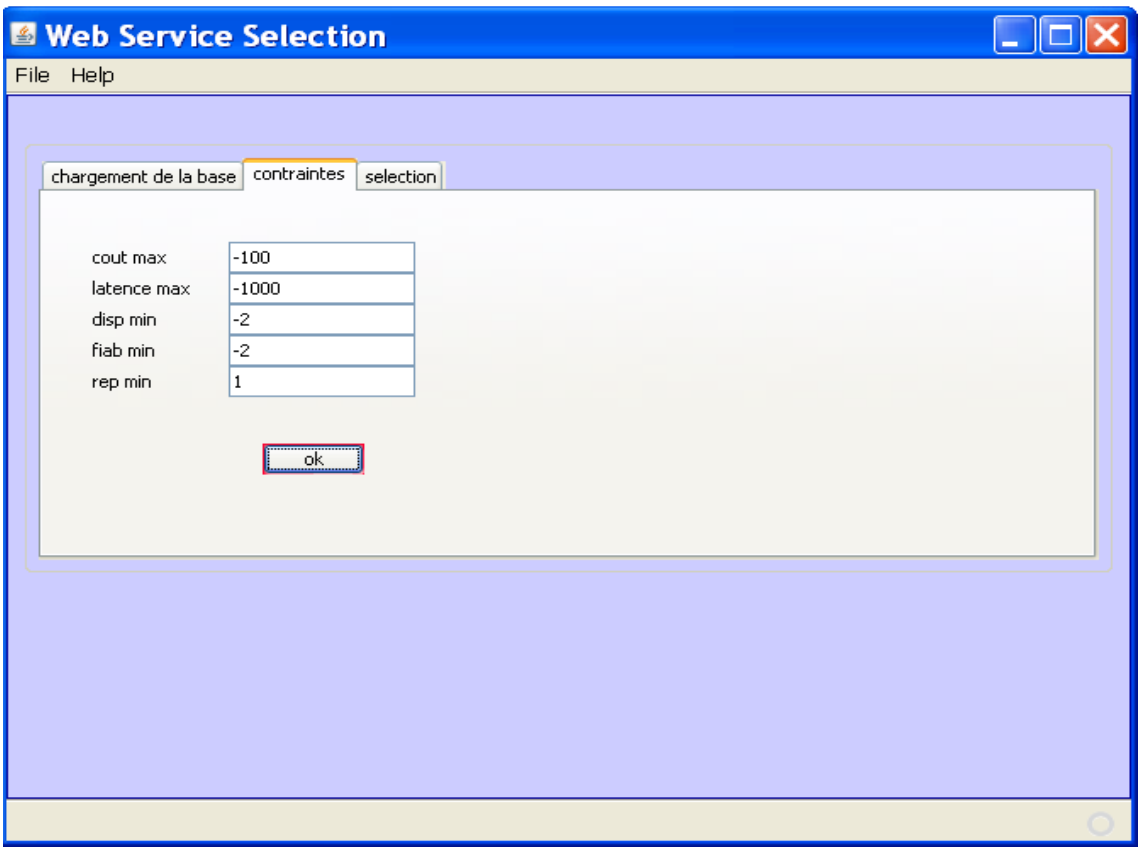
1. Fenêtre principale

La fenètre principale se compose de trois onglets ;

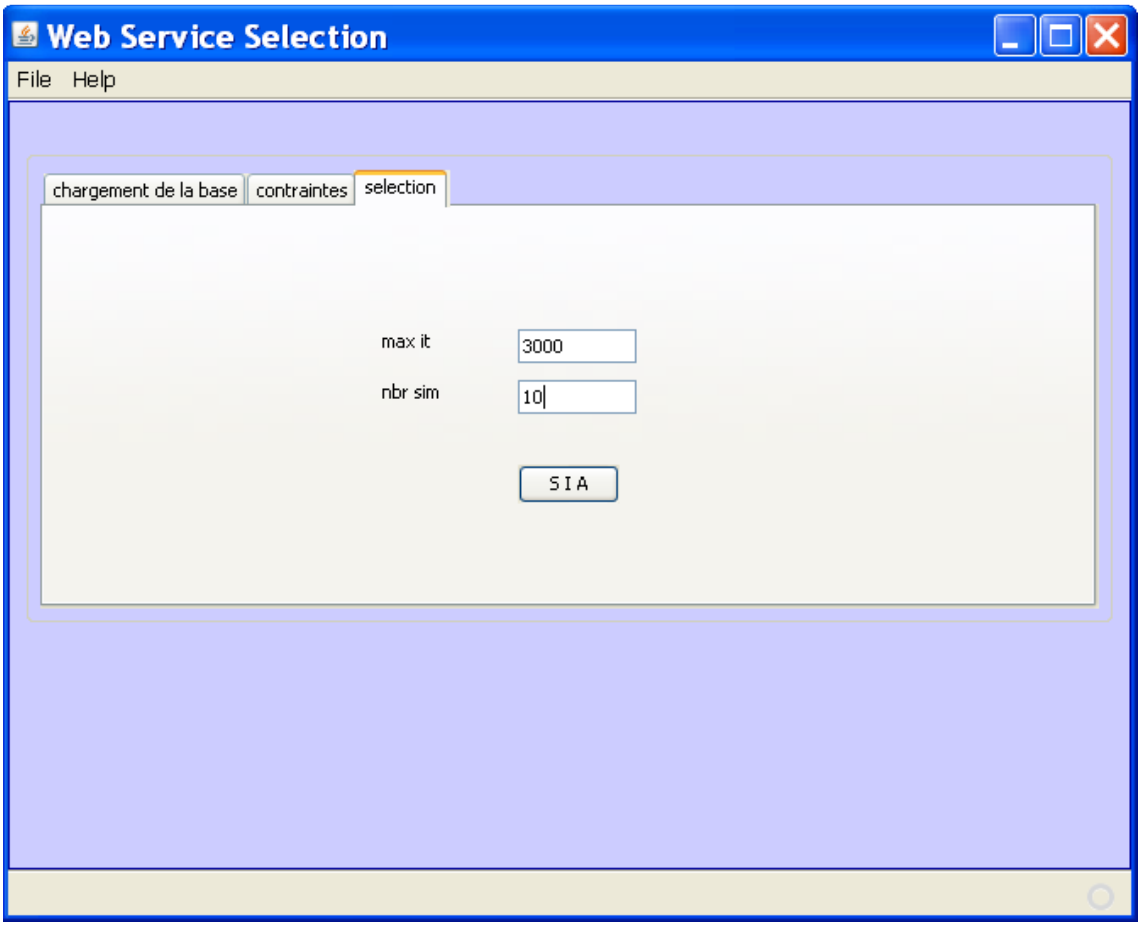
- Le premier onglet : pour le chargement, génération, enregistrement et affichage de la base de donnée.



- Le deuxième onglet : destinée pour la saisie et validation des contraintes.



- Le troisième onglet : ce dernier onglet est destiné pour la sélection à base de SIA, dont l'utilisateur a le choix de saisir le nombre d'itération et simulation qui lui convient.



V. Expérimentation

Nous avons mené une expérience pour évaluer la performance de l'approche proposée. Nous utilisons le scénario précédant (exemple de motivation du chapitre II) comme notre environnement de test pour configurer les paramètres de l'expérience. Le but est de démontrer comment notre approche peut aider le client à sélectionner la meilleure offre. Nous courons notre expérience sous NetBeans IDE 6.9.1 de Sun Microsystems sous le système d'exploitation Windows, Processeur Dual-Core, 2 Giga de RAM.

La figure (IV.8) montre 10 simulations de l'algorithme à base de sélection clonale.

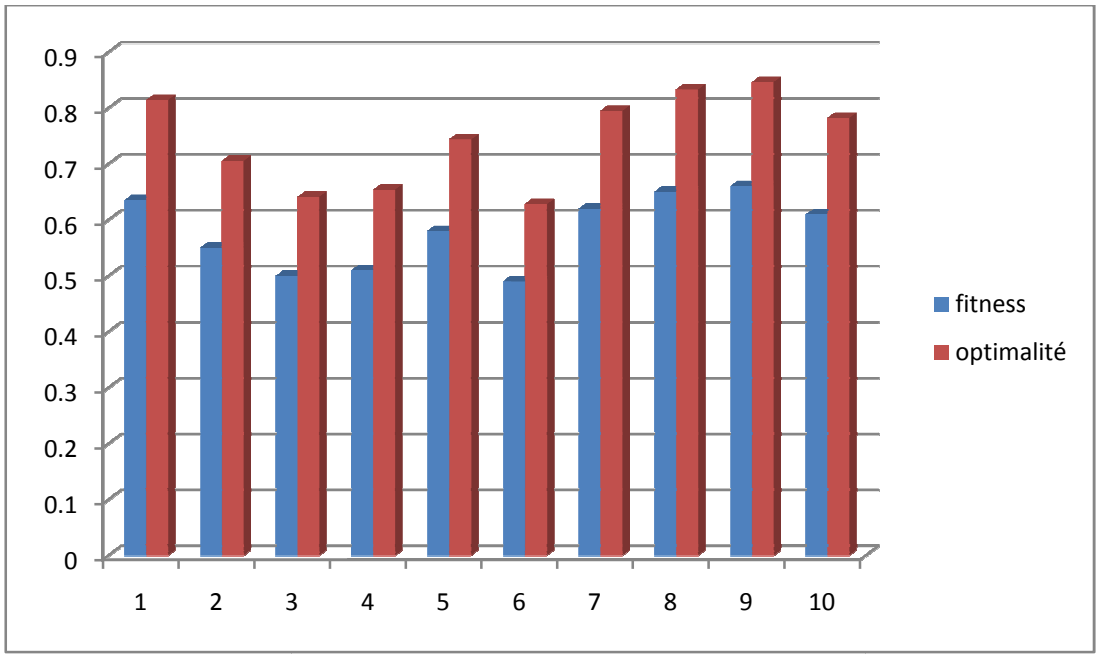


Figure IV.4 Histogramme de fitness et optimalité des 10 simulations

On remarque que les résultats sont proches de l'optimale et on peut atteindre un taux égale à 85%.

Fitness	Optimalité
0.635	0.81410256
0.55	0.70512821
0.5	0.64102564
0.51	0.65384615
0.58	0.74358974
0.49	0.62820513
0.62	0.79487179
0.65	0.83333333
0.66	0.84615385
0.61	0.78205128

Table IV.2 Les fitness et leurs optimalité

VI. Conclusion

Dans ce chapitre on a proposé un algorithme à base de sélection clonale pour le problème de sélection de service web. Les résultats obtenus sont très acceptables et montrent leur supériorité par rapport à d'autres approches.

Conclusion Générale

Ce projet de fin d'étude présente le fruit de notre travail, il nous a permis de connaitre beaucoup de choses qui ont amélioré nos connaissances.

A travers ce travail nous a essayés d'atteindre les besoins de sélection des services web. Nous avons présenté aussi une application qui utilise l'algorithme à base de sélection clonale pour instancier une composition qui satisfait les besoins de l'utilisateur. La composition concrète recherchée doit maximiser un ensemble de critères de Qos (Quality of service) positifs et minimiser un ensemble de critères négatifs, en plus elle doit satisfaire un groupe de contraintes globales.

En ce qui concerne les perspectives de notre travail nous prévoyons les points suivants :

- L'utilisation de l'aiNet (Artificial Immune Network) pour le problème de sélection
- L'utilisation de la programmation par contrainte (et en particulier l'algorithme Branch and Bound).
- L'utilisation des aiNet et clonalg dans une version multi-objectifs.

Références Bibliographiques

[1] Rosanna Bova, Salima Hassas, Salima Benbernou, Réutilisation de services web composites par la métaphore du système immunitaire, Atelier Systèmes d'Information et Services Web, INFORSID 2006.

[2] Thomas Erl, SOA Principles of Service Design (The Prentice Hall Service-Oriented Computing Series from Thomas Erl). Prentice Hall PTR, Upper Saddle River, NJ, USA, 2007.

[3] Hubert Kadima et Valérie Monfort, Web Service : techniques, démarches et outils XML, WSDL, SOAP, UDDI, Rosetta Net, UML, Livre, ISBN : 2-1000-6558-0, 2003.

[4] http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/

[5] N. TEMGLIT, H.ALIANE, M.AHMED NACER, Un modèle de composition des services web sémantiques, Centre de Recherche sur l'Information Scientifique et Technique (CERIST), ALGERIE, 2008.

[6] http://www.softeam.fr/technologies_web_services.php/

[7] World Wide Web Consortium; Extensible Markup Language (XML); http://www.w3.org/XML/

[8] W3C World Wide Web Consortium; "XQuery 1.0: An XML Query Language"; W3C Working Draft; http://www.w3.org/TR/xquery/; 12 November 2003.

[9] W3C World Wide Web Consortium; "The Extensible Stylesheet Language Family (XSL)"; Disponible à: http://www.w3.org/Style/XSL/

[10] W3C World Wide Web Consortium; "XML Path Language (XPath) Version 1.0"; W3C Recommendation 16 November 1999; http://www.w3.org/TR/xpath/

[11] M. Gudgin, M. Hadley, N. Mendelsohn, J. J. Moreau, H. F. Nielsen, A. Karmarkar, and Y. Lafon. Soap version 1.2 part 1: Messaging framework. http://www.w3.org/TR/soap12-part1/, June 2003.

[12] DALI YAHIA. M, Sélection des services web à base de QoWS, Mémoire, UABT, 2011.

[13] Hartwig Gunzer,"Introduction to web services", Sales Engineer, Borland, March 2002.

[14] Copyright Springer Verlag Berlin Heidelberg, 2004.

[15] LearnXmlws, http://www.learnxmlws.com/tutors/wsdl/wsdl.aspx.

[16] W3C (2003), “Universal description, discovery, and integration (UDDI)”. http://www.uddi.org.

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

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

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

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

[21] Ghazi GHARBI, Algorithme de Sélection dans les Applications à Services : Une Approche Basée sur la Méthode d’Analyse de Concepts Formels, Rapport de Master 2 Recherche Système d’information et Ingénierie Avancée des Logiciels, Université Joseph Fourier (Science, Technologie, Médecine), 23 juin 2010.

[22] E.Alrifai, T. Risse Combining Global Optimization with Local selection for Efficient QoS-aware Service Composition In WWW09, April 20–24, 2009, Madrid, Spain.

[23] E.Alrifai, T. Risse Selecting Skyline Services for QoS-based Web Service Composition In Proceedings of the WWW 2010, April 26–30, 2010, Raleigh, North Carolina, USA.

[24] Q Yu, A Bouguettaya. Foundations for Efficient Web Service Selection Springer Science+Business Media, 2010.

[25] M. M. Akbar, E. G. Manning, G. C. Shoja, and S. Khan. Heuristic solutions for the multiple-choice multi-dimension knapsack problem. In Proceedings of the International Conference on Computational Science-Part II, pages 659–668, London, UK, 2001. Springer-Verlag.

[26] D. Ardagna and B. Pernici. Global and local qos constraints guarantee in web service selection. In Proceedings of the IEEE International Conference on Web Services, pages 805–806, Washington, DC, USA, 2005. IEEE Computer Society.

[27] D. Ardagna and B. Pernici. Adaptive service composition in flexible processes. IEEE Transactions on Software Engineering, 33(6):369–384, 2007. Dustdar, S. and Schreiner, W. ‘A survey on web services composition’, Int. J. Web and Grid Services, Vol. 1, No. 1, pp.1–30. (2005).

[28] F. Li, F. Yang, K. Shuang, and S. Su. Q-peer: A decentralized qos registry architecture for web services. In Proceedings of the International Conference on Services Computing, pages 145–156, 2007.

[29] F. Hadjila, Chikh A, M. Dali Yahiya QoS-aware Service Selection Based on Genetic Algorithm In Proceedings of CIIA'11 Saida Algeria 2011.

[30] L. Zeng, B. Benatallah, M. Dumas, J. Kalagnanam, and Q. Z. Sheng. Quality driven web services composition. In Proceedings of the International World Wide Web Conference, pages 411–421, 2003.

[31] L. Zeng, B. Benatallah, A. H. H. Ngu, M. Dumas, J. Kalagnanam, and H. Chang. Qos-aware middleware for web services composition. IEEE Transactions on Software Engineering, 30(5):311–327, 2004.

[32] I. Maros. Computational Techniques of the Simplex Method. Springer, 2003.

[33] B. Sam et K. Hadji, Les Index de Jointures Binaires : De nouveaux algorithmes et études empiriques, Mémoire de master, USTHB, 2011.

[34] Mokhtar GHARBI, Optimisation grâce aux Systèmes Immunitaires Artificiels, CERV. Centre Européen de Réalité Virtuelle, EBV Eco-systémique et Biologie Virtuelles, 2006.

[35] P. Emilie, "Organisation du system immunitaire felin", thèse de doctorat à l'école national de Lyon, France, 2006.

[36] L.N. De Castro and F.J. Von Zuben, "Artificial Immune Systems: Part I, Basic Theory and Application," Technical Report 1999.

[37] L. N. De Castro and F.J. Von Zuben, "Learning and Optimization Using the Clonal Selection Principle," *IEEE Transactions on Evolutionary Computation; Special Issue on Artificial Immune Systems*, 2001.

[38] Leandro N. de Castro and Jonathan Timmis. *Artificial Immune Systems: A New Computational Intelligence Approach*. Springer-Verlag, September 2002.

[39] AISWeb; the Online Home of Artificial Immune Systems. [Online]. http://www.artificial-immune-systems.org/algorithms.shtml.

[40] J. Timmis, T. Knight, L.N. de De Castro, and E. Hart., "An Overview of Artificial Immune Systems," in *Computation in Cells and Tissues: Perspectives and Tools for Thought, Natural Computation Series.*, 2004, pp. 51-86.

[41] F.J Von Zuben and L.N. De Castro, "aiNet: An artificial Immune Network for Data," in *Data Mining: A Heuristic Approach.*: Idea Group Publishing, USA, 2001.

[42] A. Watkins, L. Boggess, and J. Timmis, "Artificial Immune Recognition System (AIRS): An Immune-Inspired Supervised Learning Algorithm," *Genetic Programming and Evolvable Machines*, pp. 291–317, 2004.

Résumé

L'augmentation continuelle des services web sur la toile mondiale crée de nouveaux défis pour le monde académique et industriel. En effet, la présence des services web similaires d'un point de vue fonctionnel mais différents d'un point de vue Qos, nous obligent à mettre en œuvre des techniques d'optimisation à fin de retenir les meilleures compositions de services. Dans ce travail, nous proposons une technique d'optimisation mono-objective basée sur la sélection clonale. Les résultats obtenus confirment l'efficacité des systèmes immunitaires artificiels dans le domaine de sélection de services web.

Mots clés: optimisation combinatoire, affectation sous contraintes, méta-heuristiques, système immunitaire artificiel, algorithmes évolutifs, sélection des services web.

Abstract

The quality of service satisfaction is a primordial issue in service oriented architecture. In this work, we propose a mono-objective meta-heuristic based on clonal selection, in order to select a near optimal combination of services that satisfies the global constraints required by the user, and optimize the user's Qos. This approach uses several heuristics to reduce the space search, such as the mutation, the clone. This work also reports experiments that evaluate the optimality rates of the approach.

Keywords: service oriented architecture, service composition, combinatory optimization; artificial immune system;qualiy of service; web service selection.

ملخص

إن التطوّر الكميّ لخدمات الواب على الشَّبكة العنكبوتية يزيد من صعوبة انتقائها، خاصةً عندما نأخذ بعين الاعتبار مقاييس جودة الخدمة والقيود الشاملة. لمعالجة هذه المشكلة نقتَرح في هذا العمل استعمال أنظمة المناعة الاصطناعية وبتعبير أدق خوارزمية الاستنساخ المسماة (CLONALG).

كلمات مفتاحية: إنتقاء خدمات الواب، تركيب خدمة، جودة الخدمة، تحقيق أمثلية الحاصل.