الجمهورية الجزائرية الديمقراطية الشعبية
وزارة التعليم العالي و البحث العلمي
Université Abou Bekr Belkaid
Tlemcen Algérie
جامعة أبي بكر بلقايد
تلمسان الجزائر

Faculté des Sciences - Tidjani HADDAM
Département d'Informatique

Mémoire de fin d'études

Pour l'obtention du diplôme de Master en Informatique

Option: *Modèles Intelligents et Décision (M.I.D)*

Thème :

La sélection de services Web À base de l'algorithme d'abeilles

Réalisé par :

M^elle^ BOUROUIS Meriem **Et** **M^elle^ DJERRIRI Sabah**

Soutenu le 01 Juillet 2013 à 12 h
Devant le jury composé de MM.

- *BENAZZOUZ .M* *(Président)*
- *HADJILA .F* *(Encadreur)*
- *MOUAFEK .B* *(Examinateur)*
- *BENMOUNA .Y* *(Examinateur)*

Année universitaire : 2012 - 2013

Remerciements

*Avant tout nous remercions **Allah** tout puissant d'avoir nous aider à réaliser ce travail au bon moment.*

*Nous tenons à remercier très sincèrement monsieur **HADJILA.F,** maitre d'encadrement assistant chargé, pour avoir encadré et dirigé notre travail, C'est grâce a lui qu'on a eu l'occasion de réaliser notre projet de fin d'études, et pour la patience et la disponibilité dont il a fait preuve à notre égard,*
Il peut être assuré de notre sincère respect.

*Un très grand merci à touts les **enseignants** de notre département d'informatique qui ont assuré notre formation durant ces cinq ans d'études.*

*Nous remercions plus particulièrement les membres de **jury** pour avoir accepté de juger ce modeste travail : Mr BENAZZOUZ .M, Mr MOUAFEK .B et Mr BENMOUNA .Y*

Nous ne pouvons enfin clôturer ces remerciements sans remercier du fond du cœur tous ceux qui de près ou de loin ont bien voulu nous encourager pour que ce travail puisse être achevé.

Dédicace :

Je dédie ce modeste travail à :

Mes très chers parents, en témoignage de leur amour, et dont le soutien de tous les instants m'a permis de mener à bien ce travail. J'espère que Dieu tout puissant me donne la force et le courage pour que je puisse rendre leurs sacrifices.

Mes frères : Sofiane, Abdelhamid

Ma sœur : Imane

Mon binôme M^elle BOUROUIS Meriem et sa famille.

Toute la promotion 2^éme année Master MID 2013.

Mes enseignants et à tous ceux qui me sont chers.

Sabah.

Moi aussi je dédie ce modeste travail à :

Toute ma famille, surtout Mes parents qui m'ont soutenu au cours de mes études par ses sacrifices et son confiance,

Que dieu leurs procure santé et longue vie.

Mon frère : Saïd.

Mes sœurs : Sara, Nada et Hiba.

Enfin à tous mes collègues en Master II MID,

Je les souhaite tous bon courage !!

Meriem.

Table des matières

Liste des figures

Liste des tableaux

Liste d'abréviations

- **API :** Application Programming Interface
- **AEs :** Algorithmes Evolutionnaires
- **B2B :** Business to Business
- **B2C :** Business to Consumer
- **CSW :** Composition des Services Web
- **FTP :** File Transfer Protocol
- **HTTP :** Hyper Text Transfer Protocol
- **IHM :** Interface Homme/Machine
- **MIP :** Programmation Entière Mixte
- **MOEA :** Algorithmes Évolutionnaires Multi-Objective
- **OASIS :** Organization for the Advancement of Structured Information Standards
- **OMG :** Object Management Group.
- **POP :** Post Office Protocol
- **QoS :** Qualité de Service
- **RPC :** Remote Procedure Call
- **SW :** Service Web
- **SOA :** Architecture Orientée Services
- **SOAP :** Simple Object Access Protocol
- **SMTP :** Simple Mail Transfer Protocol
- **SPO :** Optimisation par eSsaim Particulaire
- **SIA :** Système Immunitaire Artificiel
- **URI :** Uniform Resource Identifier
- **UML :** Unified Modeling Language
- **UDDI :** Universal Description Discovery and Integration
- **W3C :** World Wide Web consortium
- **WSDL :** Web Service Description Language
- **XML :** eXtensible Markup Language.

Résumé

La composition de services Web est l'une des motivations principales de l'architecture orientée service (SOA). En effet, la sélection de telles compositions est très gourmande en termes de temps, surtout lorsqu'on prend en considération la qualité de service (QoS), les préférences des utilisateurs et les contraintes globales.

Notre travail consiste à développer une application basée sur une approche d'optimisation mono-objective globale pour la sélection des meilleurs services Web en fonction de QoS. Les colonies d'abeilles exploitent les relations de voisinages et les heuristiques de dance pour explorer rapidement un grand espace de recherche.

Les résultats obtenus sont très satisfaisants et confirment l'utilité de cette approche dans la réduction de temps de réponse.

Mots clés : SOA, Services Web, Sélection de compositions de services, optimisation, qualité de service, Meta heuristique, Colonies d'abeilles.

Abstract

The composition of Web services is one of the principal motivations of service oriented architecture (SOA). Indeed, the selection of such compositions is very greedy in terms of time, especially when we take into account the quality of service (QoS), the preferences of users and the global constraints.

Our work consists in developing an artificial bee colony for optimizing the QoS Web service selection. The obtained results are very satisfactory, and confirm the utility of this approach in this domain.

Keywords : SOA, Web services, Web services selection, optimization, quality of service, Meta-heuristic, Bee colony.

ملخص

تعد عملية تركيب خدمات الويب من أهم تحديات الهيكلة الخدماتية. في الواقع انتقاء مثل هاته التركيبة يتطلب الكثير من الوقت خاصة عندما نأخذ بعين الاعتبار نوعية الخدمات المقترحة وأولويات المستعمل كما نسعى لتحقيق بعض الشروط الشاملة التي تخص التركيبة ككل.

إن عملنا هذا يقتضي تطبيق مقاربة لتحسين انتقاء تركيبات خدمات الويب , حيث إن هاته المقاربة مستوحاة من سلوكيات الحيوانات أو الحشرات التي تعيش في مجموعات, حيث أن أكثرها نظاما و ذكاءا جماعيا هي مجموعات النحل و التي تتميز بقدرتها على التواصل فيما بينها, وقد أظهرت نتائج العمل التطبيقي ذلك.

الكلمات المفتاحية : الهيكلة الخدماتية, خدمات الويب, انتقاء تركيبات خدمات الويب, مقاربة تحسينية, نوعية الخدمات, مستعمرات النحل.

Introduction générale :

I. 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 et en particulier les services web. Ces derniers ont l'avantage de faciliter l'intégration d'applications ou de systèmes d'information.

L'importance des standards dans ce contexte a sans doute accentué le phénomène. Le service web traite les données et relie à l'internet et à l'externe d'une application logiciel. Il a maintenant évolué pour englober diverses ressources d'information dans le monde entier accessibles.

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.

II. Problématique :

En quelques années, internet a touché un public de plus en plus nombreux et satisfait des besoins de plus en plus variés. Cependant, Pour certains types d'application, il est nécessaire de combiner un ensemble de services Web (services Web agrégés ou composites) et comme les services Web (SW) avec des fonctionnalités similaires sont censés être fournis par des fournisseurs concurrents, le défi majeur est de concevoir des stratégies d'optimisation qui retournent les "meilleurs" services Web où la composition de ces derniers est à l'égard de l'utilisateur qui s'attend fourni de qualité.

La sélection d'une telle composition de services Web est basé sur la QoS, dans notre cas chaque SW est caractérisé par cinq critères de QoS qui sont : la Latence, la Fiabilité, la Disponibilité, le Coût et la Réputation telle qu'on cherche à maximiser les valeurs des critères avec un sens de préférence croissant et à minimiser les critères avec une préférence décroissante généralement de type coût.

III. Contribution :

On propose dans ce travail une approche de sélection de services Web composites à base Colonies d'abeilles orientée qualité de service « QoS aware Web service selection based on Bee colony ».

Des approches basées sur le comportement de butinage s'inspirer des mécanismes sous-jacents au processus de butinage chez les abeilles. Outre les études expérimentales montre plusieurs support de modèles théoriques et de l'efficacité en contour le processus décisionnel décentralisée de l'abeille lors la recherche de nourriture.
Nous nous inspirons de ces techniques pour décrire une approche d'optimisation de sélection des meilleures compositions de services Web.

Les algorithmes d'abeilles forme une classe des méta-heuristiques récemment proposée pour des problèmes d'optimisation difficile (problèmes NP-Hard et les problèmes dynamiques), la majorité des problèmes qui ont étaient résolus par ces algorithmes, ont donné de très bon résultats concernant la valeur de la fonction objective, et le temps d'exécution qui a été acceptable.

IV. Plan du mémoire :

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

- **Le premier chapitre** est composées de deux parties essentielles, la première met l'accent sur la technologie des services Web, son architecture ainsi que les principaux standards qu'elle supporte, dans la deuxième on a présenté une partie de l'état de l'art qui introduit la problématique, nous montrons aussi un survol sur les approches proposées dans la littérature pour la sélection de services web composites à base de QoS, en citant quelques algorithmes utilisés par ces approches.
- **Le deuxième chapitre** concerne notre propre cas d'application dont le cadre de la sélection des services web en basant sur l'algorithme d'abeilles. Ce chapitre montre la conception de notre système, son implémentation ainsi que l'évaluation de ses performances.
- **La conclusion générale** résume les résultats de notre travail, et présente les perspectives que nous souhaitons réaliser dans le futur.

Chapitre I :

I. Introduction :

De nos jours, les entreprises et les systèmes d'information expriment un grand besoin pour échanger des informations et des services. Ceci nécessite des langages communs de communication. Les efforts de standardisation de ces langages ont donné lieu à un nouveau domaine de recherche connu sous le nom de « protocoles B2B ». Une technologie émergeante dans ce domaine a permis de tracer quelques pistes intéressantes pour la communication entre entreprises. Cette technologie est celle de services Web.

Les services Web sont un paradigme naissant qui vise à la transposition des architectures par composant dans le cadre du Web. Un service Web est un composant logiciel qui offre des services à travers une interface standardisé. La particularité des services Web est l'utilisation de la technologie Internet comme infrastructure pour la communication entre eux.

Les fonctionnalités offertes par les services Web sont relativement simples, alors que les besoins des utilisateurs sont de plus en plus complexes, à tel point qu'un seul service Web ne peut pas les satisfaire. L'implication de la composition des services Web, dans le but de créer un nouveau service Web dit composite ainsi que la sélection automatique de ces dernières offre des nouvelles perspectives qui répondent aux exigences complexes des utilisateurs .

II. Services web :

II.1. Définition service Web :

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, …). Ci-dessous, nous citons quelques définitions généralement acceptées et fournies :

Selon [Justin, 2002] : « Un service web est une agrégation de fonctionnalités publiées pour être utilisées. Il utilise internet comme conduit pour réaliser une tâche. Il est semblable à un processus métier virtuel qui définit des interactions au niveau application ».

Selon IBM [Colan, 2003] : « Les services web sont la nouvelle vague des applications Web. Ce sont des applications modulaires, auto-contenues et auto-descriptives qui peuvent être publiées, localisées et invoquées depuis le web. Les services web effectuent des actions allant de simples requêtes à des processus complexes. Une fois qu'un service Web est déployé, d'autres applications peuvent le découvrir et l'invoquer ».

Le consortium W3C définit un service Web comme étant : « une application ou un composant logiciel identifié par un URI, dont ses interfaces et ses liens peuvent être décrits en XML, sa définition peut être découverte par d'autres services Web et il peut interagir directement avec d'autres services Web à travers le langage XML et en utilisant des protocoles Internet standards ». [W3C-WSA-Group, 2004]

II.2. Architecture des services Web :

L'exposition et l'utilisation des services se fait dans un contexte particulier qui définit clairement les interactions entre le service et ses utilisateurs. Les services Web communiquent via un ensemble de technologies fondamentales qui partagent une architecture commune. Ils ont été conçus pour être réalisés sur de nombreux systèmes développés et déployés de façon indépendante. Les technologies utilisées par les services Web sont HTTP, WSDL, XML-RPC, SOAP et UDDI.

L'architecture standard d'un service Web est organisée en plusieurs couches. Chacune d'elles répond à des préoccupations fonctionnelles différentes telles que la publication, la description, la messagerie et le transport. Comme illustré sur la Figure I.1 Les différentes couches de l'architecture d'un service Web s'interfacent avec des standards, comme suit:

- **La couche Business Processus :** Cette couche supérieure permet l'intégration de services Web, elle établit la représentation d'un « *BusinessProcess* » comme un ensemble de service Web. De plus, la description de l'utilisation de différents services composant ce service est disponible par l'intermédiaire de cette couche.
- **La couche de publication :** repose sur l'annuaire UDDI (Universal Description Discovery and Integration), qui assure le regroupement, le stockage et la diffusion des descriptions des services Web.

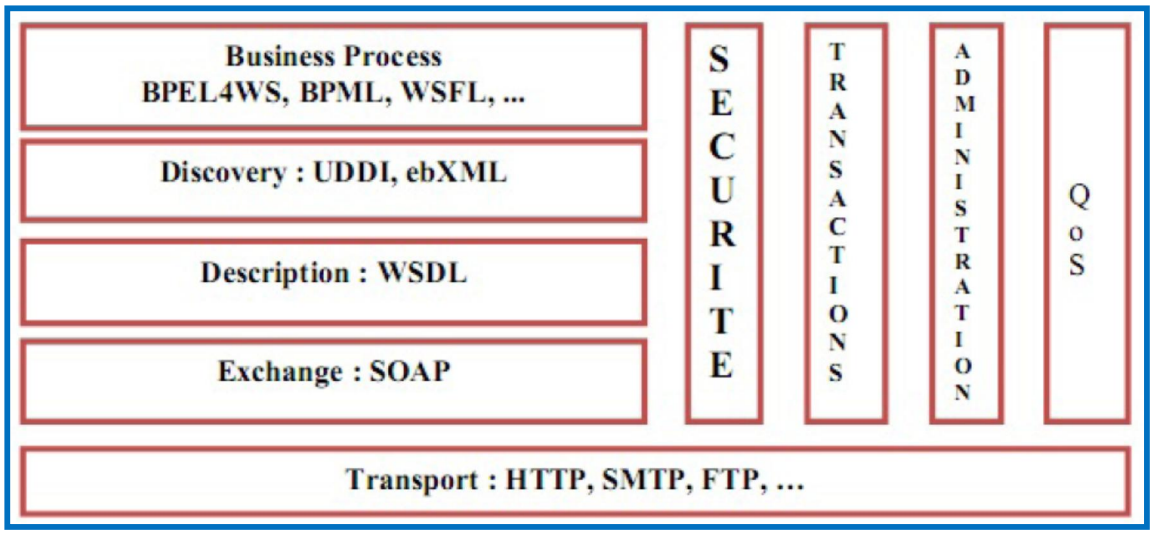


Figure I.1 : Architecture en Pile des services Web [Bertino, 2004].

- **La couche description :** est prise en charge par le langage WSDL (Web Service Description Language) [Christensen et al., 2001], qui décrit les fonctionnalités fournies par le service Web, les messages reçus et envoyés pour chaque fonctionnalité, ainsi que le protocole utilisé pour la communication.
- **La couche communication :** la couche de communication des messages propose différents mécanismes liés à l'acheminement des messages (format de communication des messages, adressage, routage...etc.). Cette couche utilise des protocoles reposants sur le langage XML (eXtensible Markup Language), car sa syntaxe unique résout les conflits syntaxiques lors de l'encodage des données. Actuellement, SOAP (Simple Object Access Protocol) est le protocole le plus utilisé pour cette couche.
- **La couche transport:** le protocole le plus utilisé dans cette couche est l'HTTP (Hyper Text Transfer Protocol). Cependant, d'autres protocoles peuvent être utilisés, tels que le SMTP (Simple Mail Transfer Protocol) ou le FTP (File Transfer Protocol), permettant ainsi aux services Web de rester indépendants du mode de transport utilisé.
- **Les couches transversales :** Security, Transactions, Administration, QoS, Ce sont des couches qui rendent viable l'utilisation effective des services Web dans le monde industriel.

II.3. Standards de services Web :

Une caractéristique qui a permis un grand succès de la technologie des services web est qu'elle est construite sur des technologies standards, nous présentons dans cette section les trois standards de base, à savoir, le protocole SOAP, le langage WSDL et l'annuaire UDDI. Ces trois standards sont basés sur le langage XML.

i. **SOAP :** est un standard du Consortium W3C définissant un protocole de transmission de messages permettant la normalisation des échanges de données. Il présente un ensemble de règles pour structurer des messages, qui peuvent être utilisées dans de simples transmissions unidirectionnelles, mais il est particulièrement utile pour exécuter des dialogues requête-réponse RPC (Remote Procedure Call) en utilisant http comme protocole de communication, et aussi les protocoles SMTP et POP (Post Office Protocol) [Mitra et al, 2003].

   SOAP assure l'interopérabilité entre composants tout en restant indépendant des systèmes d'exploitation et des langages de programmation, donc théoriquement les clients et les serveurs de ces dialogues peuvent fonctionner sur n'importe quelle plate-forme et être écrits dans n'importe quel langage à partir du moment où ils peuvent formuler et comprendre des messages SOAP. Il représente donc un composant de base pour développer des applications distribuées, qui exploitent des fonctionnalités publiées comme services par des intranets ou Internet. Il utilise principalement les deux standards HTTP et XML :

   - HTTP comme protocole de transport des messages SOAP. Il constitue un bon moyen de transport en raison de sa popularité sur le web.
   - XML pour structurer les requêtes et les réponses, indiquer les paramètres des méthodes, les valeurs de retours, et les éventuelles erreurs de traitements,

La structure du message SOAP est illustrée dans la figure I.2 par la suite :

L'en-tête du protocole de transport : qui dépend de protocole de transport utilisé, par exemple si le protocole HTTP est utilisé, l'en-tête contient :

- La version de protocole HTTP utilisée.
- La date de génération de message SOAP.
- Le type d'encodage du contenu (généralement de type XML).

L'enveloppe SOAP : la partie principale d'un message SOAP est l'enveloppe (symbolisée par la balise enveloppe), cette dernière est subdivisée en deux sous-parties: la partie en-tête (Header) et la partie corps du message (Body).

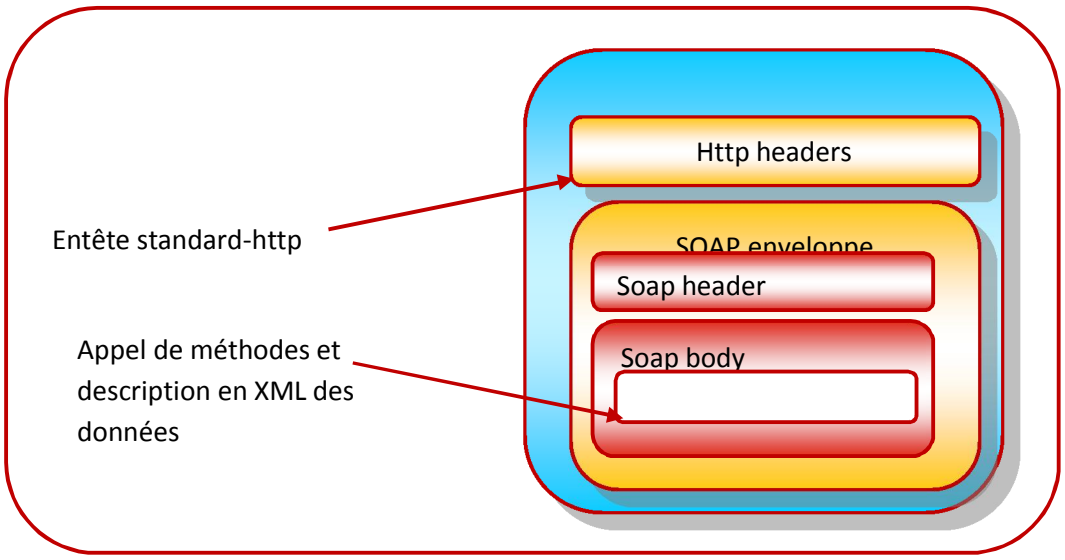


Figure I.2 : Structure de message SOAP.

- *L'en-tête du message SOAP* : est optionnelle, extensible. Les balises XML qui permettent de symboliser cette partie sont: <env:Header> et </env:Header> qui peuvent être complétées par des attributs permettant de définir le domaine de noms du service Web. En fait, l'en-tête permet principalement d'ajouter des informations sur le comportement des différents nœuds intermédiaires, lors de traitement du message.

  Un nœud étant un intermédiaire SOAP, incluant le récepteur et l'émetteur SOAP, désignable depuis un message SOAP. Son rôle est de traiter l'en-tête, ensuite de transférer le résultat (le message SOAP modifié) à un autre intermédiaire (qui peut être le récepteur final).
- *Le corps du message SOAP* : l'élément corps de message SOAP contient des données spécifiques à l'application. Les balises symbolisant cette partie sont: <env:Body> et </env:Body>. Les données doivent donc être sérialisées selon l'encodage XML. En plus, des données de cette partie peuvent transporter un type spécial comme: les messages d'erreurs (SOAP Fault).

ii. **UDDI** (Universel Description, Discovery and Integration) : est un standard défini par OASIS. Il définit la structure d'un annuaire de services Web, et la structure de gestion de services (publication, localisation, découverte) sous forme de répertoire, il permet de stocker les informations nécessaires pour retrouver et accéder à un service, telles que les informations techniques et l'adresse des services Web, le nom de la personne/société qui gère un service donné, la description des fonctionnalités [Clement et al., 2004].

Un service d'annuaire UDDI est un service Web qui gère les méta-données des services, l'information sur les fournisseurs de services et les implémentations des services. Afin de trouver un service Web, il est possible d'utiliser un annuaire UDDI en précisant des exigences concernant le service requis. On cherche le service par son nom et/ou par des mots clés.

L'annuaire UDDI se concentre sur le processus de découverte de l'architecture orientée services (SOA), et utilise des technologies standards telles que XML, SOAP et WSDL qui permettent de simplifier la collaboration entre partenaires dans le cadre des échanges commerciaux. L'accès au référentiel s'effectue de différentes manières.

Les informations sur un service publié dans un annuaire UDDI se présentent sous trois facettes comme illustré sur la figure I.3 [Hubert et al., 2003] :

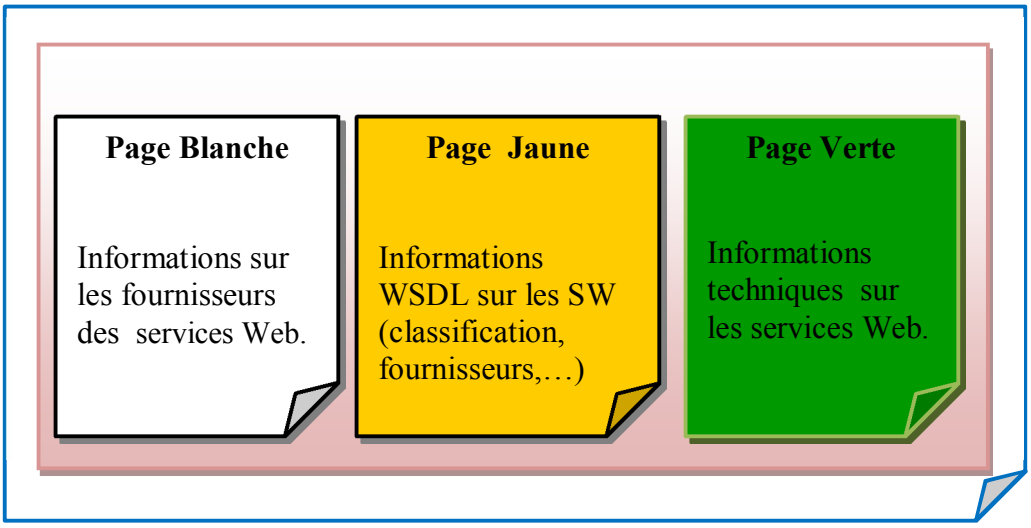


Figure I.3 : les trois facettes de l'annuaire UDDI [Hubert et al., 2003].

- Les pages blanches comprennent la liste des entreprises ainsi que des informations associées à ces dernières (coordonnées, description de l'entreprise, identifiants...).
- Les pages jaunes recensent les services Web de chacune des entreprises sous le standard WSDL.
- Les pages vertes fournissent des informations techniques précises sur les services fournis.

Un annuaire UDDI est conçu pour être interrogé par des messages SOAP, afin de pouvoir stocker et fournir des informations permettant l'accès aux documents WSDL (Web Services Description Language) décrivant les protocoles et les formats de messages nécessaires pour interagir avec les services Web répertoriés dans l'annuaire. Les outils de recherche disponibles sont basés sur des mots-clés et ne prennent pas en

considération les relations entre les services Web et les caractéristiques sémantiques de chaque service Web, forçant l'utilisateur à reformuler sa requête en utilisant de nouveaux termes clés.

Les spécifications UDDI inclue notamment :

- des API SOAP qui permettent l'interrogation et la publication d'informations,
- la représentation XML du modèle de données de l'annuaire et des formats de message SOAP
- des définitions de l'interface WSDL de SOAP,
- des définitions d'APIs de différents modèles techniques qui facilitent le travail des systèmes d'identification et de catégorisation des enregistrements UDDI.

Un registre UDDI se compose de cinq types de structures de données principales décrites sous forme de schémas XML: le Business entity, le Business service, le Binding template, la Tmodel et *Publisher assertion*. Cette répartition par type fournit des partitions simples pour faciliter la localisation rapide et la compréhension des différentes informations qui constituent un enregistrement comme montre la figure I.4 :

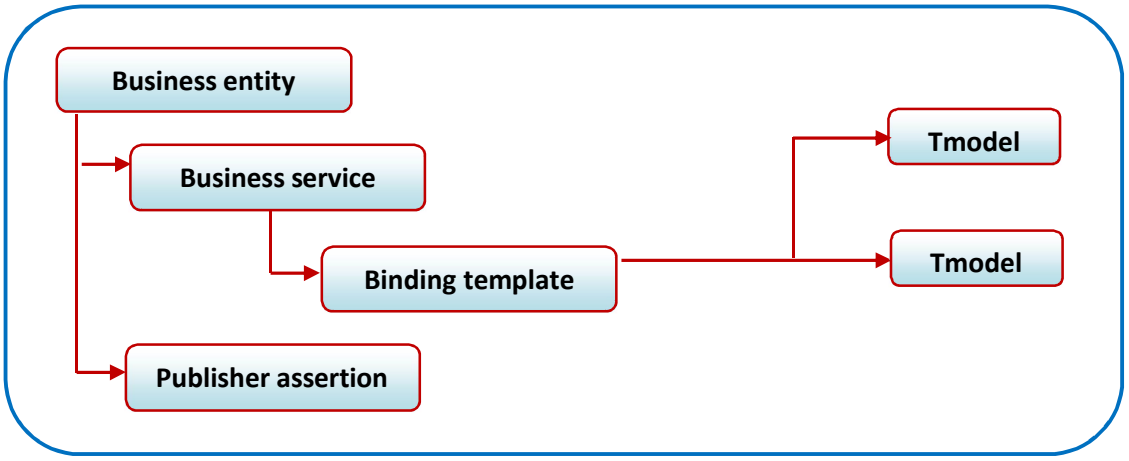


Figure I.4 : Structures de données de l'annuaire UDDI [UDDI 2.0].

- *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 (pages blanche).
- *Business service* : ensemble d'informations sur les services publiés par l'entreprise (nom du service, description textuelle, les catégories…) (pages jaunes).
- *Binding template* : ensemble d'informations sur le lieu d'hébergement du service (c.à.d. l'adresse du point d'accès du service) (pages vertes).

- *Tmodel :* ensemble d'informations sur le mode d'accès au service (définitions WSDL), il peut agir aussi d'une spécification abstraite ou d'une taxonomie.
- *Publisher assertion :* ensemble d'informations contractuelles entre les partenaires.

iii. **WSDL** (Web Service Description Language) **:** Le besoin d'une description claire de la communication entre services Web a abouti à des normalisations aux niveaux des messages échangés et des protocoles. Une grammaire XML bien structurée était proposée pour décrire les services Web et paramétrer les échanges de messages. Un langage de description s'est basé sur cette grammaire et il est devenu le standard des services Web, c'est le WSDL.

WSDL [Christensen et al., 2001], est un standard du W3C, qui permet de définir une syntaxe XML pour décrire les méthodes et les paramètres des services Web invocables par le biais de messages au format SOAP. Il permet de définir qu'est-ce qu'un service Web est capable de faire, son emplacement et comment l'invoquer.

La figure I.5 montre que le standard offre une description à deux parties représentent des concepts du modèle WSDL : une première partie de description abstraite et une deuxième partie des éléments de description concrète.

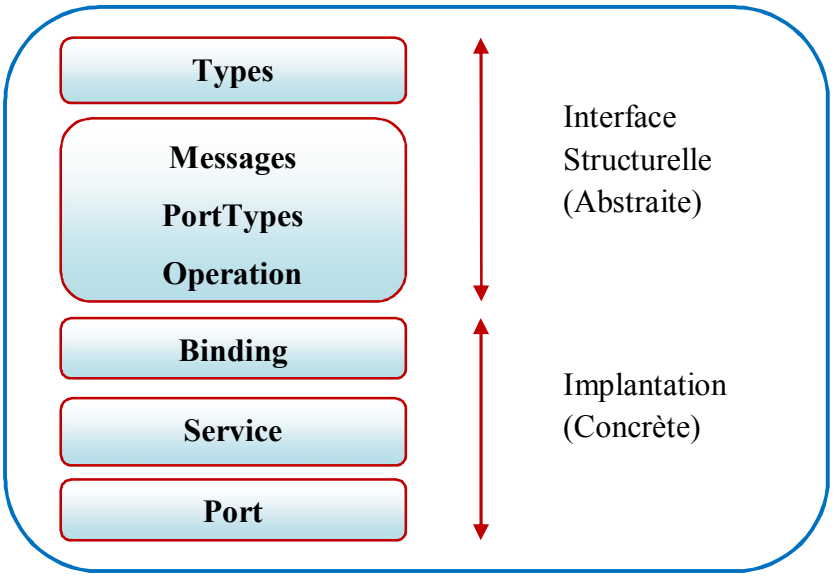


Figure I.5 : Structure d'une description WSDL [Christensen et al., 2001].

Les éléments de la partie abstraite décrivent principalement le service en termes de messages (envoyés et reçus). Ils sont indépendamment décrits par un schéma XML de description concrète.

Au niveau d'une balise XML notée "types", la partie abstraite inclut des éléments notés "operation" dont chacune associe un ou plusieurs messages à un modèle d'échange de messages. Le modèle spécifie le nombre et l'ordre ainsi que la cible et la source abstraites des messages. Les opérations sont groupées dans un élément noté "interface" ou aussi "portType" sans préciser le moyen (le protocole) d'échange de messages.

Techniquement, la structure d'un élément "binding" ressemble à la structure d'un élément "interface/portType" mais avec des détails en plus. La spécification WSDL offre deux types de "binding" pour l'envoi de messages entre client et services Web :

- soit intégrés dans des messages du protocole SOAP transmis par exemple en HTTP;
- soit directement par exemple dans des messages du protocole HTTP.

III. Sélection de services Web :

Les organisations sont de plus en plus nombreuses à se tourner vers des architectures à base de services Web pour le développement et l'intégration d'applications ou de systèmes d'information. Il arrive très fréquemment que de nombreux services répondent à un même ensemble de besoins fonctionnels. Ces services se distinguent les uns des autres par leurs propriétés non fonctionnelles. On intéresse ici à les comparer selon des critères de QoS (cout, réputation, disponibilité, etc.).

Par la suite on va présenter une partie de l'état de l'art qui introduit la problématique formellement, nous montrons aussi un survol sur les approches proposées dans la littérature pour la sélection de services web composites à base QoS en citant quelques algorithmes utilisés par ces approches.

III.1. Exemple de motivation :

La sélection des compositions de services « QoS-aware service composition », est l'une des problématiques les plus importantes de l'architecture orientée service. Pour la présenter on considère la situation suivante :

On suppose qu'il y a un utilisateur qui veut planifier un voyage, pour cela il a besoin de consommer 3 types de services au minimum, une réservation de billet d'avion, une réservation d'hôtel et une location de voiture comme montre la figure I.6, on note aussi qu'on doit sélectionner un seul service (ou entreprise, fournisseur,…) de chaque catégorie (ou classes) en utilisant les critères de QoS (réputation, fiabilité, couts, temps d'exécution…), en plus l'utilisateur exige des contraintes globales sur chaque critère de

QoS tel que une contrainte globale s'applique sur les 3 services sélectionnés, par exemple le cout total des 3 services ne doit pas excéder une certaine limite.

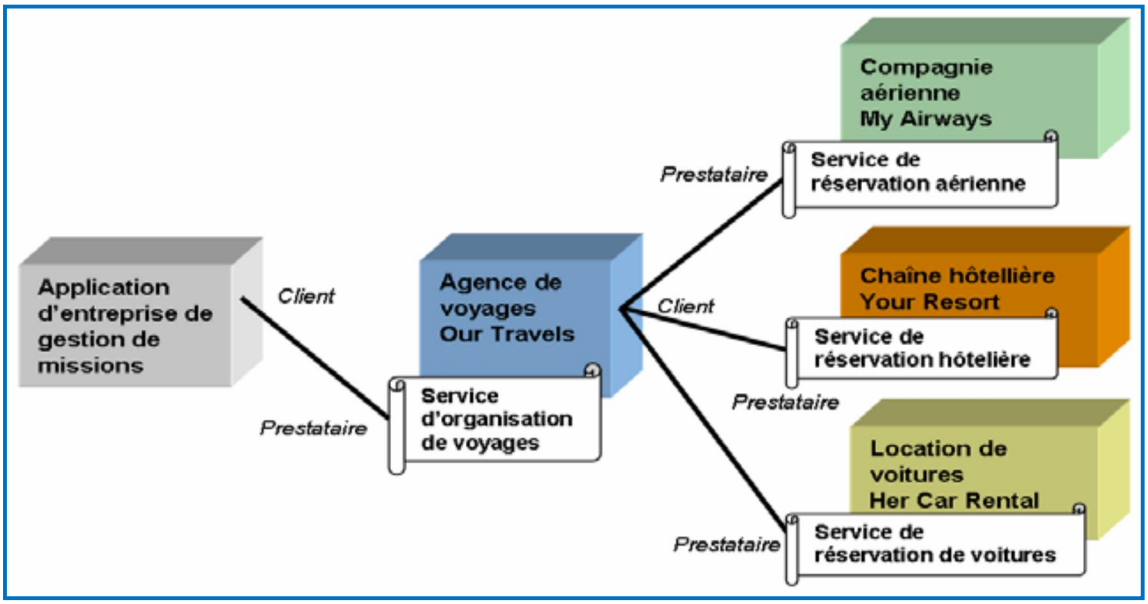


Figure I.6 : Exemple de composition des SW de l'agence de voyage

Ce problème est NP-Hard, les solutions exactes de ce problème ont une complexité exponentielle, et de ce fait nous ne pouvons plus garantir les exigences temps réelles, ou le passage à l'échelle.

III.2. Formalisation du problème :

La problématique de sélection de services web composites à base de QoS constitue un raffinement de la composition de services web. Plus précisément nous considérons, dans ce cas un workflow abstrait, c.-à-d. une composition dont les tâches ne sont pas encore affectées aux services et nous voulons la concrétiser. En d'autres termes nous cherchons des instances concrètes qui peuvent remplacer les nœuds abstraits, de telle sorte que la QoS du résultat est optimisée, et en plus les contraintes globales des utilisateurs sont satisfaites, ces contraintes portent généralement sur les facteurs de QoS, tels que le cout, la disponibilité…

Il est clair que cette sélection est une version multidimensionnelle du problème du sac à dos [Pisinger, 1995], par conséquent toute solution exacte à ce problème nécessite un temps d'exécution exponentielle, et de ce fait nous ne pouvons garantir les exigences temps réelles des utilisateurs. De façon plus formelle, nous modélisons le problème comme suit, soient :

- $C_{abstraite}$ = {S1, . . ., Sn}, une composition abstraite qui représente la requête de l'utilisateur, c.-à-d. les n classes de services à consommer (Si).
- Cons = {c_1, . . ., c_m}, Un ensemble de contraintes globales définies par l'utilisateur.
- $C_{concrète}$ = {s_1, . . ., s_n}, une composition concrète, c.-à-d. nous remplaçons chaque classe « Si » par un service concret (s_i Si).

Nous devons rechercher une composition concrète C telle que :

- La ou les sous fonctions objectives $U_k(C)$ sont optimisées. (U_k représente la sous fonction objective du K^{eme} attribut de QoS). Si on agrège les m attributs de QoS en une seule valeur alors nous aurons besoin d'une seule fonction objective U.
- Les contraintes globales sont vérifiées, c.-à-d. $Q'(k(c)) \geq c_k$, $\forall$ c_k Cons.

Si nous supposons que le nombre de candidats par classe est l, alors le nombre global de compositions possibles est l^n, l'énumération totale de ces compositions ne peut être faite en temps raisonnable, en plus la présence des contraintes globales, impose un temps exponentiel pour avoir une solution exacte.

III.3. Approches de sélection de services web :

La sélection automatique de services web, a fait l'objet de plusieurs travaux, de façon générale on distingue 03 grandes classes : la sélection mono objective, La sélection multi objective, et la sélection hybride (mono et multi objective), ces classes forment une hiérarchie comme montrée sur la figure I.7, par la suite on va détailler chacune des approches en donnant quelques algorithmes utilisés par ces dernières :

III.3.1. La sélection mono-objective :

Cette catégorie suppose que les m valeurs de qualité de service soient agrégés en un seul score, c.-à-d. On considère une seule fonction objective qui associe des poids aux différents attributs de QoS.

Cette dernière est aussi divisée en 03 sous approches (classes) : la sélection locale, la sélection globale et hybride c.-à-d. globale et locale.

a) La sélection globale :

Ces approches explorent un espace de recherche dont les nœuds sont des compositions complètes (c.-à-d. contenant toutes les taches), on distingue deux sous classes l'optimisation globale exacte et approximative ;

i. L'optimisation globale exacte :

Elle se base sur la programmation entière ou la programmation par contrainte, ou les énumérations exhaustives, ces méthodes donnent des résultats optimaux. [Kritikos et Plexousakis, 2009]. Ces approches ont un temps d'exécution exponentiel, les travaux de [Zeng et al. 2003, 2004] utilisent les techniques de programmation entière et mixtes (MIP) [Nemhauser et Wolsey, 1988] pour trouver la composition optimale des services. Dans la même optique [Ardagna et al. 2005, 2007] Étendent le modèle de programmation linéaire afin d'inclure les contraintes locales. Dans ce modèle, les contraintes globales sont exprimées sur la composition entière, et par l'utilisateur final, tandis que les contraintes locales peuvent être spécifiées par le concepteur de la composition au niveau des classes.

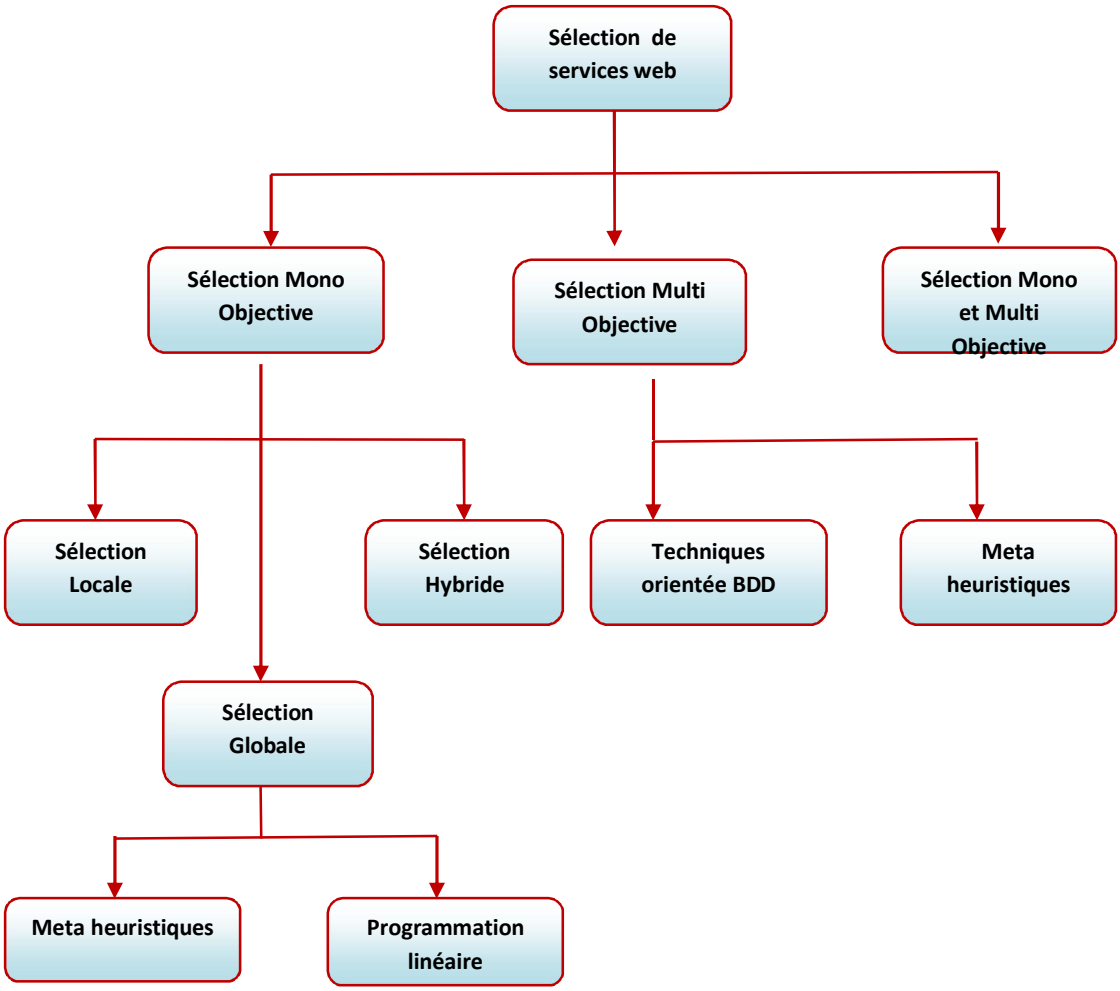


Figure I.7 : Approches de sélection de services web composites à base QoS.

[Lei Xu et al., 2011] Proposent une méthode de sélection de services web à base QoS en considérant une variation de valeurs de QoS en fonction de l'intervalle de temps.

[Yi Xia et al., 2011] Proposent une sélection globale de composition de services en adoptant 04 structures de flux de contrôle : parallélisme, séquence, choix conditionnels, et les boucles, la requête est formalisée en BPEL, elle possède 05 critères de QoS. L'algorithme est nommé « qssac », il peut donner un résultat proche de l'optimal (mais les auteurs n'ont pas d'expérimentations sur le degré d'optimalité).

[Zhai et al. 2009] proposent un algorithme à base de MIP afin de remplacer des services en pannes, la réparation doit garantir la satisfaction des contraintes globales de la composition.

Selon [Maros, 2003] la programmation entière est efficace si le nombre de service l est petit par contre si l dépasse une certaine limite (quelques milliers) alors le temps d'exécution n'est plus raisonnable.

ii. L'optimisation globale approximative (méta-heuristiques) :

Cette catégorie consiste à explorer une partie de l'espace de recherche, en adoptant des recherches heuristiques ou des méta-heuristiques (algorithmes d'optimisation génériques, qui adoptent une recherche locale), elles permettent de donner des résultats proches de l'optimal, tout en ayant un temps d'exécution abordable (polynomial). Par contre elles ne peuvent être généralisées pour le reste des problèmes.
Plusieurs travaux sont inspirés des algorithmes heuristiques :

En [Khan, 1998] qui est l'un des premiers algorithmes proposés pour le problème de sac à dos (version MMKP). L'auteur propose une heuristique UHE pour sa résolution, UHE utilise une mesure appelée la consommation des ressources agrégés pour mettre à jour un élément de chaque groupe à chaque tour de sélection.

En [Akbar et al. 2001] Modifient l'heuristique en créant M-UHE, ils proposent une étape de prétraitement de trouver une solution réalisable et une étape de post-traitement pour améliorer la valeur totale de la solution.

En [Akbar et al. 2006], proposent une autre heuristique, C-UHE, et évaluent sa performance et son optimalité contre plusieurs heuristiques, y compris la M-UHE. Les résultats montrent que la C-UHE est meilleur par rapport à M-UHE en termes de temps d'exécution. Cependant, les expériences montrent également que M-UHE est la meilleure en termes de degré d'optimalité, tandis que l'optimalité de C-HEU diminue à

mesure que le nombre de candidats par classe augmente. Les expériences montrent que l'algorithme C-UHE fonctionne mieux dans le cas où l'objectif à maximiser (par exemple, la valeur d'utilité de la composition de services) n'est pas proportionnel aux besoins en ressources (c.-à-d. les valeurs de QoS). Et de ce fait elle ne peut être appliquée pour la sélection de services composé à base de qualité (puisque l'utilité dépend des qualités de service).

Une version modifiée de l'algorithme M-UHE, appelée WS-UHE, est conçue par [Yu et al., 2007]. La complexité temporelle de WS-UHE est polynomiale.

En dépit de l'amélioration significative de ces algorithmes par rapport aux solutions exactes, les deux algorithmes ne sont pas scalables (c.-à-d. si le nombre de services Web croit de manière dramatique alors le temps d'exécution n'est plus temps réel). De plus, l'algorithme WS-UHE n'est pas adapté au caractère distribué des services web. Ceci est dû au fait que WS-HEU applique des améliorations sur une composition de services dont les valeurs de QoS proviennent de plusieurs facilitateurs. (Chaque facilitateur gère une classe donnée).

L'optimisation par essaim particulaire (SPO) est utilisée dans une approche de sélection de services web pour faire aussi une optimisation mono-objective et globale. C'est une nouvelle classe des méta-heuristiques proposée en 1995 par Kennedy et Eberhart. [Kennedy & Eberhart, 1995], TRIBES est un algorithme SPO créé par M. Clerc dans [Clerc, 2003]. Cet algorithme permet d'avoir un paramétrage automatique de l'optimisation par essaim particulaire.

La sélection clonale un autre algorithme utilisé pour faire une optimisation mono-objective et globale sur l'espace de recherche dans une approche de sélection de services web. Dans le même but de résoudre le problème de sélection de services web, et toujours des idées inspirées des mécanismes naturels qui ont été exploitées pour développer des heuristiques inspirées de la nature.

Le système immunitaire artificiel (SIA) est un paradigme récent qui tente de capturer des caractéristiques intéressantes des systèmes immunitaires naturels, comme la mémorisation, la reconnaissance de formes, l'apprentissage… La théorie de la sélection clonale a été utilisée comme source d'inspiration pour le développement des SIA qui effectuent des taches d'optimisation, Castro et Von Zuben [Von et Castro, 2001] ont développés 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 taches de filtrage et d'optimisation.

b) La sélection locale :

Elle consiste à filtrer un seul service de chaque classe en utilisant une fonction objective et indépendamment des autres classes, ensuite elle compose les n résultats qui correspondent aux n classes. Cette approche possède une complexité linéaire O(l), en plus elle est fortement adaptée aux environnements distribuée, en effet la gestion de la QoS (mesures, mise à jours …) est faite par des facilitateurs (brokers) distribués. [Benatallah et al. 2002] [Li et al. 2007]. Mais nous notons en contrepartie la non prise en charge de cette classe des contraintes globales (ex : le cout global de la composition), ce qui la rend obsolète pour les problèmes réels.

[Alrifai et al, 2008] utilisent une approche locale qui consiste à diviser les contraintes globales, en contraintes locales en se basant sur la distribution statistiques des valeurs de QoS. Les auteurs gèrent uniquement la séquence.

Dans [Benouaret et al. 2011] les auteurs considèrent la sélection de services web en prenant en compte les préférences sur les sorties de services, pour cela ils appliquent un tri local (pour chaque classe) en adoptant la notion de fuzzy dominance, cette dernière permet la comparaison de 02 services. En fin ils retiennent les K premières solutions. Les auteurs ne gèrent pas les contraintes globales.

c) La sélection hybride :

Cette approche est un compromis des deux précédentes approches, en commençant la recherche par une optimisation globale, puis continuant le travail avec une optimisation local, sa complexité temporelle est inférieure à celle de l'optimisation globale, l'approche peut également manipuler des contraintes globales.

Alrifai en [Alrifai et al. 2012] montre que plusieurs chercheurs ont proposé l'utilisation de la programmation de nombre entier mixte [Nemhauser et Wolsey, 1988] pour résoudre le problème de composition de service web à base QoS.
Des variables binaires de décision sont employées dans le modèle pour représenter les candidats de service. Un service candidat sij est choisi dans la composition optimale si son variable xij correspondant est placé à 1 dans la solution du modèle et jeté autrement. Pour inclure les variables de décision, le problème de résoudre le modèle peut être formulé comme problème de maximisation de valeur de service globale, sujet aux

contraintes globales de QoS, tout en satisfaisant les contraintes d'attribution sur les variables de décision.

Pour faire face aux limitations des approches précédentes, nous divisons le problème averti de composition de service web à base de QoS en deux sous-problèmes qui peuvent être résolus plus efficacement dans deux phases suivantes :

Dans la première phase, le compositeur de service décompose chaque contrainte globale de QoS en contraintes locales au niveau de services de composant et envoie ces contraintes aux courtiers de service.
Dans la deuxième phase, chaque courtier de service effectue le choix local pour trouver le meilleur service composant qui satisfait les contraintes locales.
Les résultats indiquent que l'approche hybride est plus performante que l'WS-UHE en termes d'optimalité.

III.3.2. La sélection multi-objective :

Un grand nombre d'approches existent pour résoudre les problèmes d'optimisation multi-objective. Certains d'entre eux utilisent la connaissance qu'ils ont au sujet du problème pour donner des préférences à quelques objectifs, de ce fait déviant l'aspect multi-objective. D'autres donnent à tous les objectifs le même niveau d'importance,… Parmi ces approches, nous devrions distinguer deux catégories : approches non-Pareto et Pareto. Les approches de Non-Pareto ne traitent pas réellement le problème comme problème multi-objective. Elles essayent de le convertir en problème mono-objectif. D'autre part, les approches de Pareto ne transforment pas les objectifs du problème, mais essayent de les optimiser simultanément.

L'optimisation à base de skyline « the skyline based optimization » peut être manipulé en employant les techniques de base de données multiples [Bouguettaya, 2010], par exemple nous pouvons employer l'algorithme de clivage et conquérir, l'algorithme Bitmap, l'algorithme basé sur index (B tree, Hash table), et l'algorithme de plus proche voisin (R tree).

Il existe plusieurs travaux qui tient compte des préférences d'utilisateur pour sélectionner les k top skylines dominants [Alrifai, 2010] 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 parteo-dominance pour ranger les services web.

L'utilisation des algorithmes évolutionnaires pour résoudre des problèmes Multi-objective a été motivée principalement en raison de la nature d'AEs basée sur la population qui permet la génération de plusieurs éléments du Pareto ensemble optimal dans une seule course. Les Algorithmes Évolutionnaires multi-objective (MOEA) être parmi les méthodes de résolution les plus puissantes pour l'optimisation multi-objective [Coello et al., 2002]. MOEA tiennent compte des objectifs contradictoires et laissent trouver un ensemble des solutions non dominées.

III.3.3. La sélection mono et multi-objective :

C'est une approche hybride qui combine les deux sélections mono et multi-objective, telle que elle effectue une recherche (ou filtrage) multi objective pour chaque classe, après elle groupe hiérarchiquement les services de chaque classe en choisissant un représentant de ces groupes, ensuite elle continue avec une recherche mono-objective (par niveau) sur les représentants des groupes (ex. la sélection clonale).

En [Alrifai et al., 2010] utilisent une approche de sélection hybride, telle qu'ils effectuent une recherche multi objective pour chaque classe, en groupant hiérarchiquement les services de chaque classe en choisissant les services skylines car seulement ces services ne sont pas dominés par n'importe quel autre service et sont valides comme candidats pour la composition en utilisant l'algorithme intelligent K-means, ensuite elle continue avec une recherche mono-objective sur les représentants des groupes en utilisant le standard MIP pour décomposer les contraintes de QoS en contraintes locales, qui sont alors employées pour choisir efficacement le meilleur service de chaque classe. Les variables dans le modèle MIP de l'approche hybride représentent les niveaux locaux de QoS de chaque classe de service plutôt que les candidats réels de service, en le rendant scalable au nombre de service candidats que l'approche d'optimisation globale.

Les résultats de l'évaluation expérimentale indiquent une performance significative par rapport aux approches existantes, qui se fondent sur l'optimisation globale. Ces expériences ont prouvé que l'exécution des méthodes à base des skylines est affectée par la difficulté du problème de composition, en termes de nombre de contraintes de QoS.

IV. Conclusion :

Dans ce chapitre on a présenté une formalisation du problème de sélection automatique de services web avec un exemple de motivation, ensuite on a montré une partie d'état de l'art des différentes approches de sélection de services Web composites proposées dans la littérature pour résoudre ce problème.

Donc les services Web peuvent être utiles dans la plupart des scénarios applicatifs lorsque la communication peut être établit sur un modèle bidirectionnel (requête/réponse). L'application des services Web est multiple, autant dans les domaines du B2B (Business to Business) que dans les B2C (Business to Consumer).

Le chapitre suivant est consacré à étudier notre application dans ce domaine c.-à-d. la sélection de services Web à base de colonies d'abeilles.

Chapitre II :

I. Introduction :

Le nombre de services sur internet devient de plus en plus important ce qui nous a invités à concevoir des approches de sélection plus efficaces, surtout pour les requêtes composées. En général, nous pouvons trouver un ensemble de services qui fournissent la même fonctionnalité mais différent en terme de qualité, dans telle situation que nous devons choisir les meilleurs services, on a besoin d'appliquer un certain algorithme d'optimisation. Dans ce travail nous proposons une approche de sélection des services web par une application cliente. Nous utilisons un modèle à base de poids pour un ensemble de paramètres de QoS (Qualité de Service), pour la sélection du meilleur service web. Les paramètres de QoS sont définis selon les besoins des applications clientes. Notre objectif est de simuler l'optimisation à base de colonies d'abeilles sur le problème de sélection de services web.

Dans ce dernier chapitre nous allons voir les différentes étapes suivies durant la réalisation de notre application, nous commencerons d'abord par présenter le scénario de préparation de voyage, qui illustre notre problématique, nous présenterons la base de services, ensuite nous décrivons les outils qui ont permis la construction du prototype, puis nous donnons les différents diagrammes UML représentant la phase conception ainsi qu'une description de l'IHM de notre application, et nous finissons par une discussion des résultats que nous avons obtenus.

II. Scénario :

Nous considérons une agence de voyages [Bouguettaya et al, 2009], nommé « travelagency », fournissant l'organisation de voyages (par exemple, le transport, l'itinéraire, et l'hébergement) pour ses clients (figure II.1). Supposons qu'un professeur d'université, X, veut assister à une conférence internationale à Sydney, en Australie.

Les services typiques nécessaires par X pourraient inclure les compagnies aériennes, au sol de transport (par exemple, de taxi et de location de voitures), les services de divertissement d'hébergement (par exemple, des hôtels), et d'autres (par exemple, un restaurant et de l'opéra). X a besoin à la recherche d'abord pour les services qui fournissent des paquets de voyage. La recherche peut être effectuée dans certains registres de services bien connus.

Cet exemple exprime un scénario typique de la sélection de services Web dans le domaine de tourisme et c'est ce qu'on veut simuler dans notre application avec la sélection de dix services :

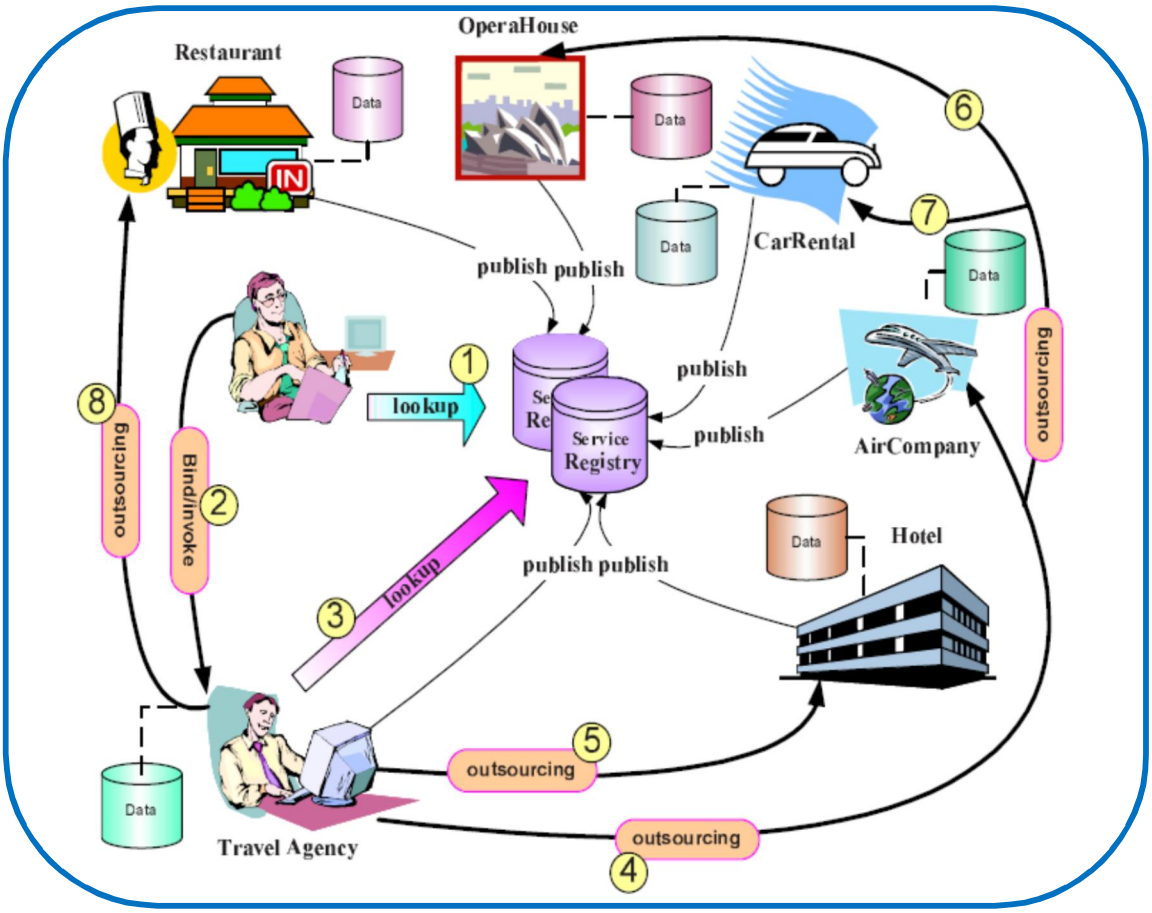


Figure II.1 : Un scénario de préparation de voyage [Bouguettaya et al, 2009]

III. Qualité de Service :

Etant donné qu'il y a plusieurs services Web publiés sur Internet, beaucoup d'entre eux, ne peuvent satisfaire les différents besoins d'un utilisateur. Il sera donc nécessaire, d'ordonner et de classer plusieurs services Web en se basant sur différents critères de qualité. En effet, en disposant d'une bibliothèque de services Web similaires, nous pouvons faire une sélection à la main, en faisant l'écoute sur les services les plus populaire, ou guidée par un outil d'aide à la décision. La sélection des services Web a pour objectif de déterminer le service le plus adéquat.

III.1. Définitions :

Jusqu'aujourd'hui, il n'existe pas de consensus sur la définition de la qualité de service (QoS). Mais on peut la définit comme "un ensemble d'exigences dans le comportement collectif d'un ou plusieurs objets".

Dans le contexte des technologies de l'information et multimédia, la QoS a été définie par [Vogel et al, 1995] comme "l'ensemble des caractéristiques quantitatives et qualitatives d'un système multimédia, nécessaires pour atteindre la fonctionnalité requise par l'application".

Nous pouvons aussi dire que la qualité de service représente l'aptitude d'un service à répondre d'une manière adéquate à des exigences, exprimées ou implicites, qui visent à satisfaire ses usagers. Ces exigences peuvent être liées à plusieurs aspects d'un service, par exemple :

- Le débit : Le nombre de requêtes servies pendant un intervalle de temps.
- Le temps de réponse : Le temps requis pour compléter une requête du service web.
- La fiabilité : La capacité d'un service d'exécuter correctement ses fonctions.
- La scalabilité : La capacité du service de traiter le plus grand nombre d'opérations ou de transactions pendant une période donnée tout en gardant les mêmes performances.
- La robustesse : La probabilité qu'un service peut réagir proprement à des messages d'entrée invalides, incomplètes ou conflictuelles.
- La disponibilité : La probabilité d'accessibilité d'un service.
- Le prix d'exécution : C'est le prix qu'un client du service doit payer pour bénéficier du service.
- La réputation : C'est une mesure de la crédibilité du service. Elle dépend principalement des expériences d'utilisateurs finals.
- La sécurité : C'est un regroupement d'un ensemble de qualités à savoir : la confidentialité, le cryptage des messages et le contrôle d'accès.

III.2. Modèles de QoS existants :

Le groupe de travail « Architecture des Services Web » du W3C travaillant sur les architectures des services web, a identifié et décrit un ensemble de paramètres de QoS pour les services web [Kangchan et al, 2003], à savoir : la performance (qui englobe le débit, le temps de réponse et le temps d'exécution), la fiabilité, la scalabilité ou l'adaptation au facteur d'échelle, la capacité, la robustesse, le traitement d'exception, l'exactitude, l'intégrité, l'accessibilité, la disponibilité, l'interopérabilité et la sécurité.

Il n'y a pas un consensus bien précis au sujet de l'ensemble des QoS importantes pour les services web, mais la plupart des travaux de recherche qui ont essayé d'identifier et de classifier les paramètres de QoS ont pris en considération les paramètres définis par le W3C aux quels sont associés dans certains travaux d'autres paramètres.

III.3. Critères de QoS considérés :

Dans ce qui suit nous nous focalisons sur les cinq critères de QoS suivants pour calculer la qualité de la composition de services Web sélectionnée par notre système :

i. **Coût :** est défini comme le montant (selon une certaine devise) pour exécuter l'opération.

ii. **Le Temps de latence :** c'est le temps nécessaire pour traiter une requête dès l'instant de son envoi jusqu'au moment de la réception de la réponse.
   - Formule latence = Le temps de réception de la réponse par le client – Le temps d'envoi de la requête par le client.
   - Quantification : seconde.

iii. **La Disponibilité :** représente le pourcentage de requêtes réussites par le fournisseur. Les réponses échouées correspondent aux exceptions reçues du côté client.
   - Formule Disponibilité = Nombre de requêtes réussites / Nombre total de requêtes
   - Quantification : Pourcentage.

iv. **La Fiabilité (sûreté) :** la fiabilité des opérations est la capacité que l'opération doit être exécutée dans le délai prévu au maximum.
   - Formule fiabilité = 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.
   - Quantification : Pourcentage.

v. **La Réputation :** de l'opération est une mesure de la fiabilité de l'opération. Elle dépend principalement du rapport à laquelle la fourniture effective du service est conforme à sa promise.

   Les qualités fraîches peuvent être obtenues sur la base sur la publicité des fournisseurs de services dans la description du service alors que la réputation est basée sur le classement des utilisateurs finaux.

III.4. Calcul de la qualité de composition de SW :

III.4.1. Fonctions d'agrégation :

Certains des critères de QoS peuvent être évalués de manière qualitative, mais la plupart d'entre eux sont des attributs quantitatifs.

Due à la présence inévitable des erreurs et des défaillances, les systèmes ne sont jamais totalement disponibles, fiable et sur. Pour cela, ces attributs sont généralement exprimés relativement en probabilité. Pourtant, généralement tous les systèmes nécessitent de définir la disponibilité, l'intégrité et la maintenabilité.

L'ensemble des critères de QoS peut être divisé en deux: attributs négatifs et attributs positifs, telle que les valeurs des attributs négatifs ont besoin d'être minimisées (temps de réponse, prix,…), et les valeurs des attributs positifs ont besoin d'être maximisées (disponibilité, fiabilité, ...). Pour simplifier, nous considérons seulement les attributs positifs et pour les attributs négatifs il suffit de multiplier leurs valeurs par (-1).

Maintenant, nous avons besoin de calculer les paramètres de QoS pour la composition des services Web (CSW), En se basant sur la signification de chaque critère de QoS, nous définissons un ensemble de fonctions d'agrégation permettant de calculer les valeurs agrégées des critères de QoS pour la CSW.

Les fonctions d'agrégation de chaque critère sont données par le tableau II.1 :

Critères de QoS	Fonctions d'agrégation
Sûreté	$\prod_{i=1}^{n} sur(opi)$
Disponibilité	$\prod_{i=1}^{n} dis(opi)$
Réputation	$1/n \sum_{i=1}^{n} rep(opi)$
Latence	$\sum_{i=1}^{n} lat(opi)$
Coût	$\sum_{i=1}^{n} cou(opi)$

Tableau II.1 : Les fonctions d'agrégation des critères de QoS

Alors la qualité du web service composite peut être décrite par un vecteur de QoS comme suit :

$$QoS\ (CSW) = (sur(CSW), dis(CSW), rep(CSW), lat(CSW), cou(CSW)).$$

La latence et le coût prennent des valeurs scalaires dans R+, la disponibilité et la fiabilité (sureté) représentent des valeurs de probabilité (une valeur réelle entre 0 et 1), et la réputation varie sur l'intervalle [1,5], voir le tableau suivant :

Critères de QoS	Valeurs
Sûreté (sur)	0.5 – 1.0
Disponibilité (dis)	0.7 – 1.0
Réputation (rep)	1 – 5
Latence (lat)	1 – 300 (s)
Coût (cou)	1 – 30 ($)

Tableau II.2 : Les intervalles des critères de QoS [Bouguettaya et Qi, 2010]

Tout d'abords, nous avons besoin de définir quelques fonctions d'adaptation qui vont être utilisé pour les critères un peu plus spécifiques. Les deux fonctions d'agrégation concernant la sûreté et la disponibilité présentées dans le tableau II.1, ne combinent pas les paramètres de QoS de multiples opérations des services par une méthode linéaire alors nous proposons deux fonctions linéaires pour adapter les fonctions d'origines :

$$\boldsymbol{Sûreté} = \sum_{i=1}^{n} ln(sur(op_i));$$

$$\boldsymbol{Disponibilité} = \sum_{i=1}^{n} ln(dis(op_i));$$

L'objectif est que nous étendons le model de telle façon à supporter une fonction d'agrégation simple et générique qui regroupe toutes les opérations effectuées sur la composition des services Web :

$$\mathbf{Q'_i(CSW)} = \sum_{j=1}^{n} \mathbf{Q_i}(op_j);$$

III.4.2. Fonction objective :

Pour évaluer la qualité multidimensionnelle du web service composite, une fonction utilitaire est utilisée. Celle-ci permet la comparaison entre les compositions de services Web en termes de QoS.

La description de cette fonction objective nécessite le calcul de deux valeurs : une valeur maximum et un autre minimum par rapport à tous les critères de QoS.
Par exemple la valeur maximum du critère *Coût* (Q'^{max}_i) quelle que soit la composition des services Web généré peut être calculée par la sommation des valeurs des couts des services les plus coûteux (Q^{max}_i) dans chaque classe de services, formellement nous calculons la valeur agrégée maximum et minimum du i^{ème} critère de QoS comme suit :

$$\mathbf{Q'^{max}_i} = \sum_{j=1}^{n} \mathrm{Q}^{\max}_i\left(op_j\right);$$

$$\mathbf{Q'^{min}_i} = \sum_{j=1}^{n} \mathrm{Q}^{\min}_i\left(op_j\right);$$

Le présent processus est ensuite suivi par le processus de weighting pour représenter les priorités et les préférences des utilisateurs, puisqu'ils peuvent avoir des préférences sur les résultats de leurs requêtes, telle qu'ils peuvent spécifier une importance relative pour chaque paramètre de QoS. Nous attribuons des poids compris entre 0 et 1 pour chaque paramètre de QoS pour refléter leur degré d'importance. Dans notre cas les valeurs par défaut des poids sont égaux c.-à-d., 0.2 partout. Nous utilisons le score de la fonction objective «fitness» pour évaluer la qualité de la composition des services Web.

La meilleure Web service composite sera celle avec le score maximal de fitness, initialement la fitness est calculer par la formule suivante :

$$\boldsymbol{fitness\_initiale(CSW)} = \sum_{\boldsymbol{Q'_i}} \boldsymbol{w_i}\ \frac{\boldsymbol{Q'_i(CSW) - Q'^{min}_i}}{\boldsymbol{Q'^{max}_i - Q'^{min}_i}}$$

Tel que $\boldsymbol{w_i} \in \boldsymbol{R_0^+}$ et $\sum_{i=1}^{5} \boldsymbol{w_i} = \mathbf{1}$ ce sont les poids des paramètres de QoS représentant les priorités de l'utilisateur.

L'optimisation orientée QoS est « centrée sur l'utilisateur ». Le rôle de l'utilisateur est de donner le poids des paramètres de QoS. Le but est de trouver la CSW avec la meilleure qualité en fonction des préférences de l'utilisateur sur l'ensemble des CSW et la violation d'une de ces contraintes va baisser le taux d'optimalité de la solution.

Dans ce cas on doit calculer un nouveau score concernant ces contraintes violées notées « $\boldsymbol{C_i}$ » et le soustraire de la fitness_initiale calculé précédemment comme sorte de pénalité :

$$\boldsymbol{penalty} = \sum_{\boldsymbol{i=1}}^{\mathbf{5}} (\boldsymbol{C_i} - \boldsymbol{Q'_i})^{\mathbf{2}}$$

Donc le score finale sera calculé comme suit :

$$fitness = fitness\_initiale - penalty$$

IV. Algorithme d'optimisation :

Dans la nature, plusieurs espèces sont caractérisées par le comportement social. Ce comportement est également un des principaux caractéristiques des insectes sociaux (abeilles, termites, fourmis…). De ces principes-là, les chercheurs se sont inspirés pour développer des méthodes basées sur les comportements de ces animaux, et ont donné naissance à ce que l'on appelle par méta-heuristique, ce mot concerne toutes les méthodes qui modélisent l'interaction des agents qui sont en mesure de s'auto-organiser. Elles représentent des méthodes de résolution de problèmes combinatoires qui consistent à réitérer certains processus jusqu'à obtenir la solution optimale.

Notre approche consiste à proposer un algorithme d'optimisation qui se base sur les colonies d'abeilles [Pham et al., 2006], pour sélectionner efficacement les meilleures compositions de services Web en terme de QoS :

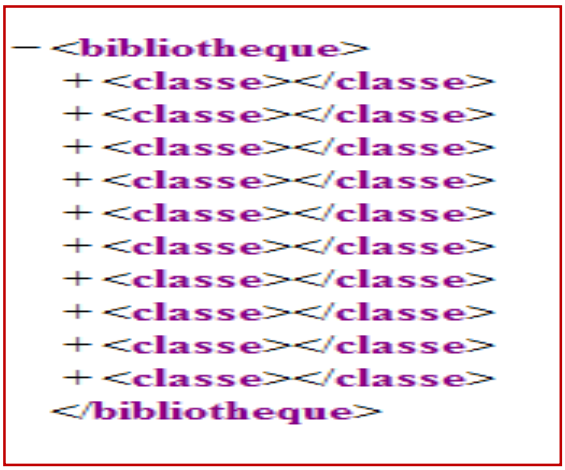
Pseudo code pour "Bees Algorithm"

```
Input : Beesnum, Sitesnum, EliteSitesnum, EliteBeesnum, OtherBeesnum;
Output : Beebest;
1 : Pop ← InitPop (Beesnum);
2 : While (it <= max_ it)
3 :  Evaluate_ pop (Pop);
4 :  Sort_ pop (Pop);
5 :  Next_ generation← Ø;
6 :  For I < Sitesnum
7 :     If I <= EliteSitesnum
8 :     Then RecruitedBeesnum ← EliteBeesnum ;
9 :          Next_ generation← Next_ generation + BestRecruitedBees;
10 :    Else
11 :         RecruitedBeesnum ← OtherBeesnum;
12 :         Next_ generation← Next_ generation + BestRecruitedBees;
13 :     End if;
14 :  End for;
15 :  RemainingBeesnum ← ( Beesnum – Sitesnum);
16 :  For j to RemainingBeesnum do
17 :         Next_ generation← Next_ generation + Random_ Bee ();
18 :  End for;
19 :  Pop ← Next_ generation;
20 : End while;
21 : Return Beebest.
```

L'algorithme d'abeilles ou « BeesAlgo » [Pham et al., 2006], est un nouvel algorithme basé sur la population de recherche. L'algorithme imite le comportement de butinage de nourriture des essaims des abeilles de miel. Dans sa version de base, l'algorithme exécute un genre de recherche de voisinage combiné avec la recherche aléatoire et peut être employé pour les problèmes d'optimisation.

V. Présentation de la base :

Notre base des services Web est un fichier XML « Base_SW.xml », qui contient dix classes de services Web comme montre l'arbre du document ci-dessous :



Et quarante provider par classe telle que chaque provider ou instance « inst » de classe est caractérisé par cinq critères de QoS, voilà un exemple des premières instances de la première classe :

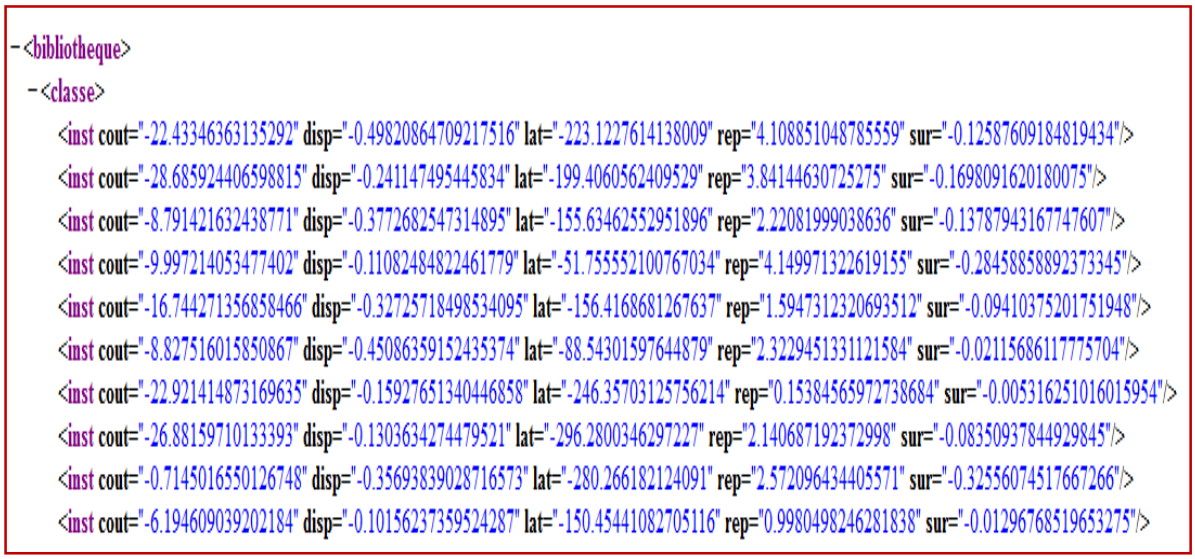


Figure II.2 : Exemple de données de la base de services Web

Les données du fichier sont générées aléatoirement d'une façon où la latence est prise dans l'intervalle comprit entre 1 et 300 seconde, la fiabilité (sureté) entre 0.5 et 1, la disponibilité entre 0.7 et 1, le coût entre 1 et 30$, et la réputation entre 1 et 5.

Notre objectif est de trouver une composition de services Web (CSW) qui maximise la fiabilité, la réputation et la disponibilité ainsi qu'elle minimise le coût et le temps de latence, en plus elle doit satisfaire les contraintes de l'utilisateur si elles existent.

VI. Description de la requête :

La requête de l'utilisateur consiste à vérifier la qualité de services de la composition sélectionnée par le system, en considérant les valeurs minimales ou maximales des cinq critères de QoS suivantes :

- La borne minimale de la fiabilité
- La borne minimale de la disponibilité
- La borne minimale de la réputation
- La borne maximale du cout
- La borne maximale de temps latence,

Ses bornes sont fixées par défaut par 100 pour le cout, 1000 pour latence, 2 pour la disponibilité et la sureté, 1 comme minimum de la réputation. Nous considérons aussi des poids égaux pour tous ces critères de QoS fixés par « 0.2 », sachant que ces valeurs peuvent être changées selon les besoins des utilisateurs.

VII. Conception :

VII.1. Processus de développement logiciel :

Un processus définit une séquence d'étapes, en parties ordonnées, qui concourent à l'obtention d'un système logiciel ou à l'évolution d'un système existant.
En d'autre terme, c'est les différentes opérations réalisées à fin d'élaborer le produit logiciel. Dans notre cas nous avons opté pour un processus unifié, ce choix est justifié par les principes sur lesquelles se base ce processus.

Un processus unifié est un processus de développement logiciel construit sur UML, il est itératif et incrémentale centré sur l'architecture, conduit par les cas d'utilisations et piloté par les risques [Rocques, 2007].

VII.2. Modélisation avec UML :

UML (*Unified Modeling Language*) : se définit comme un langage de modélisation graphique et textuel destiné à comprendre et décrire des besoins, spécifier, concevoir des solutions et communiquer des points de vue.

UML unifie à la fois les notations est les concepts orientés objet. Il ne s'agit pas d'une simple notation, mais les concepts transmis par un diagramme ont une sémantique précise et sont porteurs de sens au même titre que les mots d'un langage. UML permet de modéliser de manière claire et précise la structure et le comportement d'un système indépendamment de toute méthode où de tout langage de programmation. UML est à présent un standard défini par l'Object Management Group (OMG).

i. Diagramme de cas d'utilisation :

Le diagramme de cas d'utilisation permet de structurer les besoins des utilisateurs et les objectifs correspondants de notre système d'abeilles. Il permet aussi d'identifier les possibilités d'interactions entre le système et ces utilisateurs (intervenants extérieurs au système). Il part du principe que les objectifs du système sont tous motivés.

La figure II.2 représente le diagramme de cas d'utilisation de notre système d'abeilles :

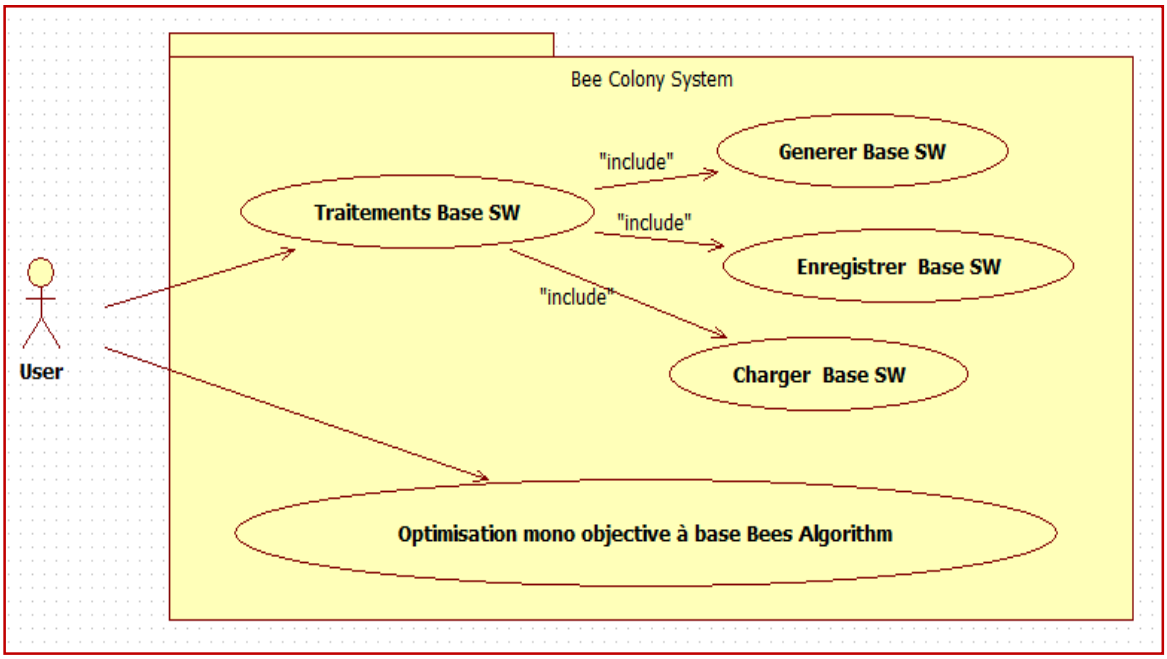


Figure II.3: Use case diagram

ii. Diagramme de classes :

Le 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 les classes qui sont les composantes de notre système final, donc nous avons l'interface qui fait appelle à la classe Lecture qui concerne tous ce qui est traitements de la base des SW et la classe de l'algorithme d'abeilles qui permette la sélection des meilleures abeilles (CSW), cette dernière a besoin d'être évaluer a l'aide de notre fonction objective (la classe Evaluation), en fin la classe Statistiques permette de représentés les résultats obtenus graphiquement (histogrammes). Donc il permet de structurer le travail de développement de manière très efficace.

La figure II.3 représente le diagramme de classes de notre application :

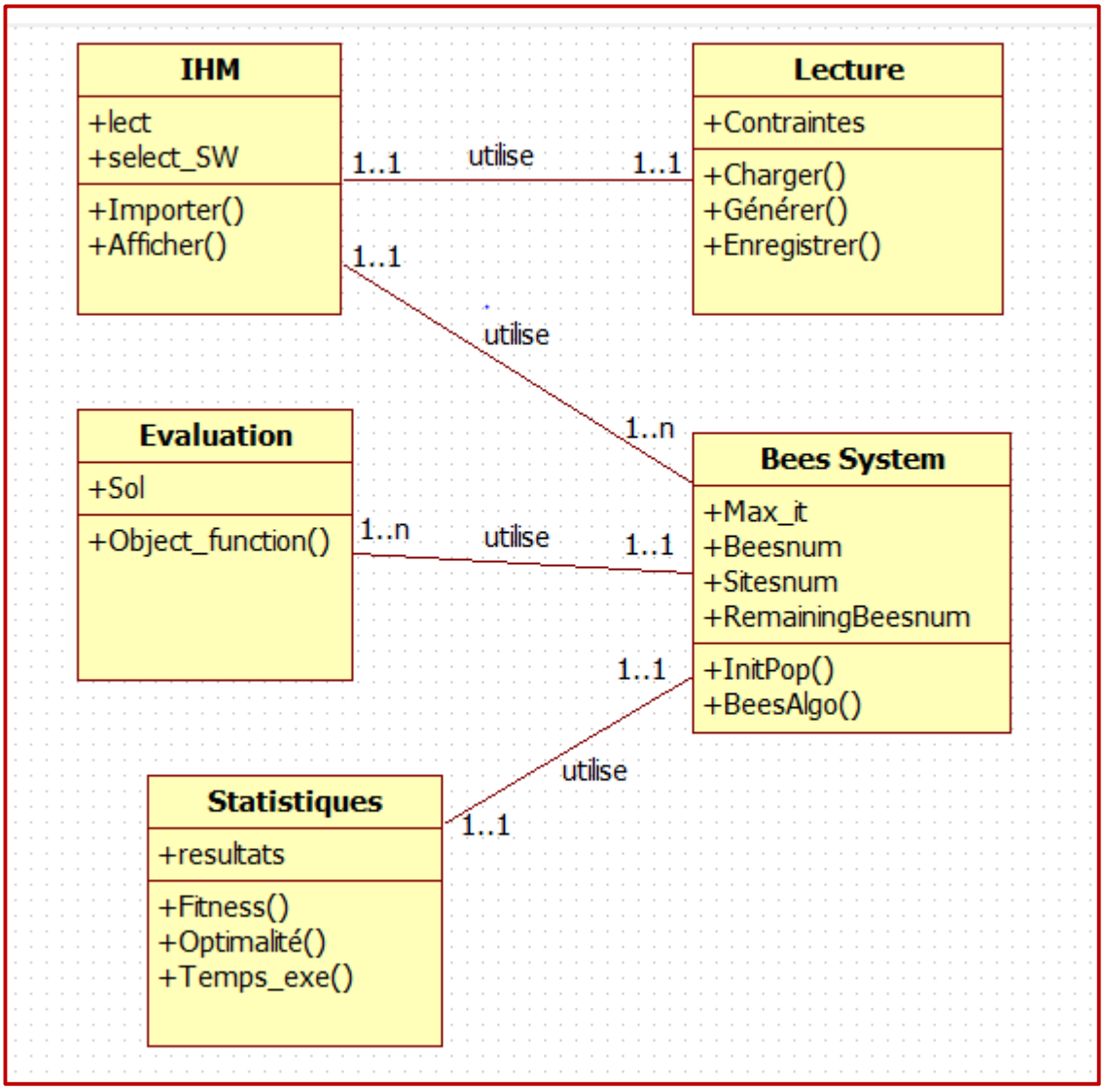


Figure II.4 : Diagramme de classes

VIII. Outils et environnement de développement :

Nous avons développé notre application sur une machine ayant un processeur Intel Core i5, avec une vitesse de 2.50 GHz, doté d'une capacité mémoire de 4GB de RAM sous Windows 7.

Avant de commencer l'implémentation de notre application, nous allons tout d'abord spécifier les outils utilisés qui nous ont semblé être un bon choix vu les avantages qu'ils offrent :

- **Langage Java :**

  Notre choix du langage java a été guidé par les avantages qu'offre la programmation orientée objet. Java est un langage de programmation à usage général, évolué et orienté objet. Ses caractéristiques ainsi que la richesse de son écosystème et de sa communauté comme l'utilisation des processus qui augmentent les performances des entrées/sorties, facilitent l'internationalisation. Il examine le programme au fil de l'exécution et libère automatiquement la mémoire. Cette fonctionnalité diminue les risques de panne du programme et ne laisse pas la possibilité de mal utiliser la mémoire et lui ont permis d'être le choix préféré pour le développement de notre application.

- **L'IDE NetBeans :**

  Nous avons développé notre application sous l'environnement de développement intégré (IDE) NetBeans version 6.1 Beta pour Java, placé en open source par Sun en 2007 sous licence CDDL (common development and Distribution license).

  Le choix de NetBeans était fondamental puisqu'il est un logiciel permettant principalement le développement en java et il fournit un environnement convivial en termes d'utilisation.

- **L'API JDOM :**

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

- **JFreeChart :**

  Est une bibliothèque open source qui permettent d'afficher des données statistiques sous une forme graphique. Elle possède plusieurs formats dont le camembert, les barres ou les lignes et propose de nombreuses options de configuration pour personnaliser le rendu des graphiques.

  Elle peut s'utiliser dans des applications standalones ou des applications web et permet également d'exporter le graphique sous la forme d'une image.

  Les données utilisées dans le graphique sont encapsulées dans un objet de type Dataset. Il existe plusieurs sous-types de cette classe en fonction du type de graphique souhaité.

IX. Implémentation et prototype :

IX.1. Présentation de l'IHM :

L'interface homme/machine (IHM) représente l'élément clé dans l'utilisation de tout système informatique. Les interfaces de notre système sont conçues de manière à être simples, compréhensible et faciles à utilisées.

Par la suite nous donnons quelques captures d'écran qui montrent les différentes parties, composantes et fenêtres de notre application :

- Commençant par la fenêtre principale de l'interface, cette première fenêtre nous donne une idée générale sur le système, c'est une sorte de présentation de l'objectif du système conçu pour une bonne compréhension des opérations et traitements qui suivent :

Figure II.5 : Fenêtre principale du système de sélection de SW.

- La deuxième fenêtre concerne notre base des services Web, elle permette d'effectuer plusieurs traitements sur cette base, par exemple :
  i. On peut charger la base en utilisant le boutant « Charger », il permet de configurer le fichier XML « Base_SW.xml » qui est constitué de dix classes de services et quarante provider par classe.
  ii. On peut aussi générer une nouvelle base d'une façon aléatoire, en précisant des intervalles des valeurs de critères de QoS convenablement à leurs sens.
  iii. Une fois on termine l'exécution, on perdre cette nouvelle base générer, mais à l'aide du boutant « Enregistrer » on peut enregistrer ce qui a été généré :

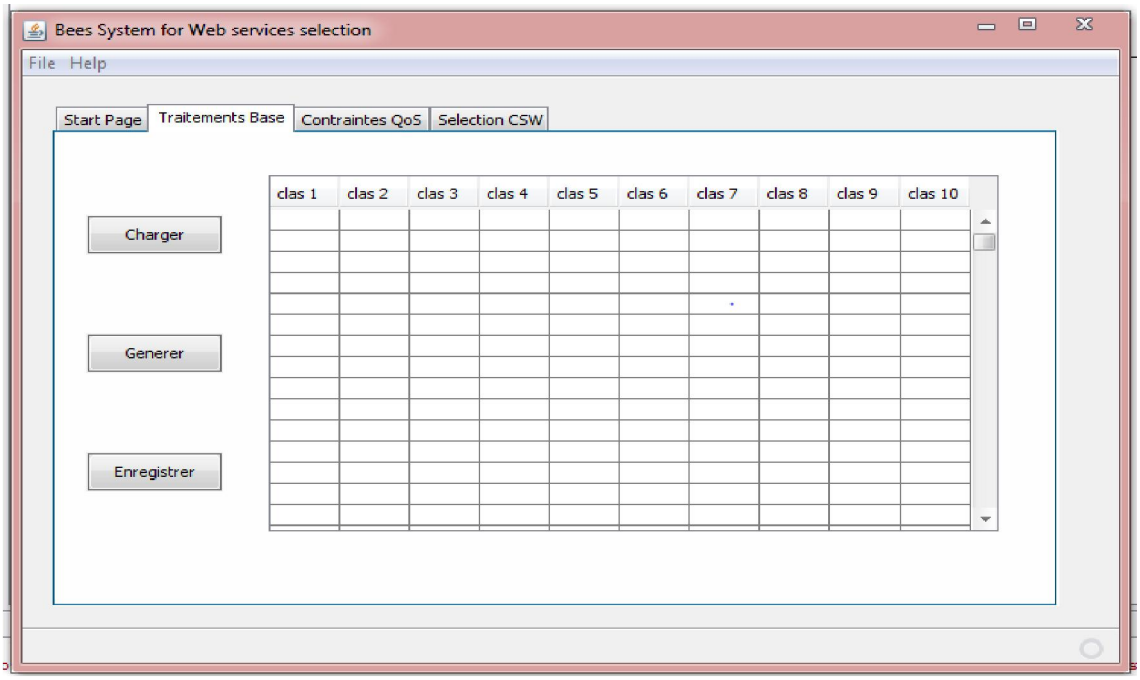


Figure II.6 : Traitements de la base des services Web

- Tout d'abord nous avons commencé par charger les données de la base des services Web, alors chaque service est caractérisé par cinq critères de QoS : le cout, le temps de latence, la disponibilité, la sureté et sa réputation c.-à-d., cinq valeurs, ces valeurs vont être rempli par colonnes, par exemple les cinq premières valeurs de la première colonne « class 1 » représentent le premier provider de cette classe et ainsi de suite, en fait on aura 200 lignes en total.

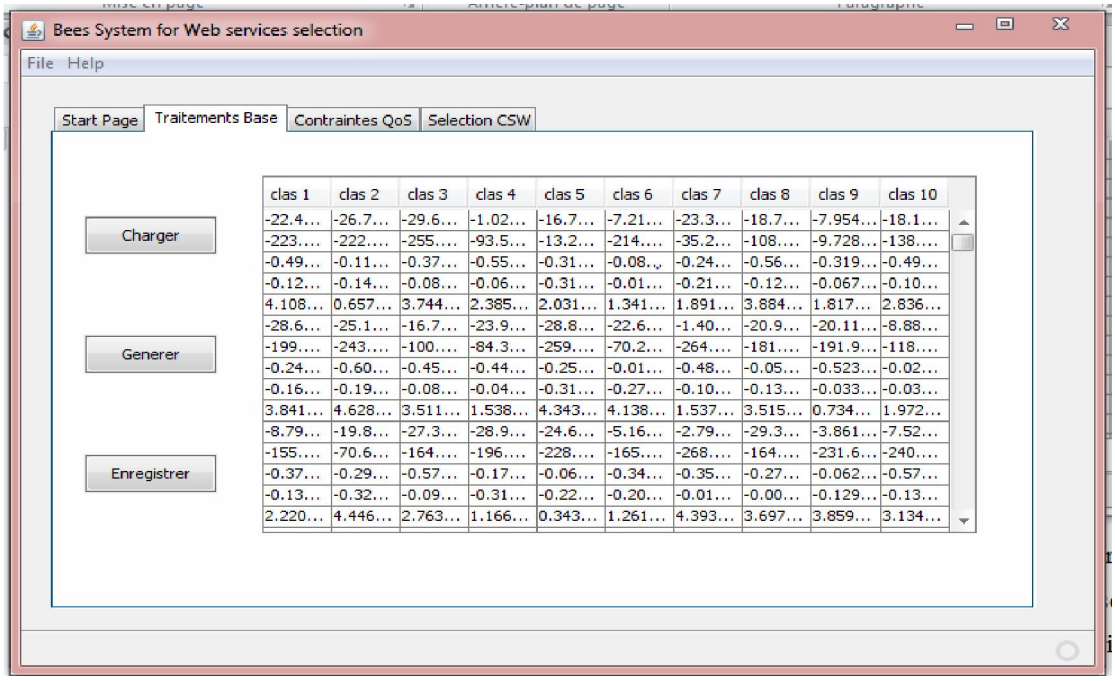


Figure II.7 : Chargement de la base de services Web.

- Ensuite nous précisons des contraintes de QoS, telle que ces contraintes doivent être vérifier durant la sélection des compositions de services Web, en respectant ces contraintes nous devons pas dépasser une certaine valeur de cout pour la totalité des services de la composition, même chose pour le temps de latence, et nous garantissons un minimum de sureté, réputation et de disponibilité de ces services sélectionnés.

Une contrainte violée ça veut dire une mauvaise composition est générée, ce qui baisse sa fitness ainsi que le taux d'optimalité :

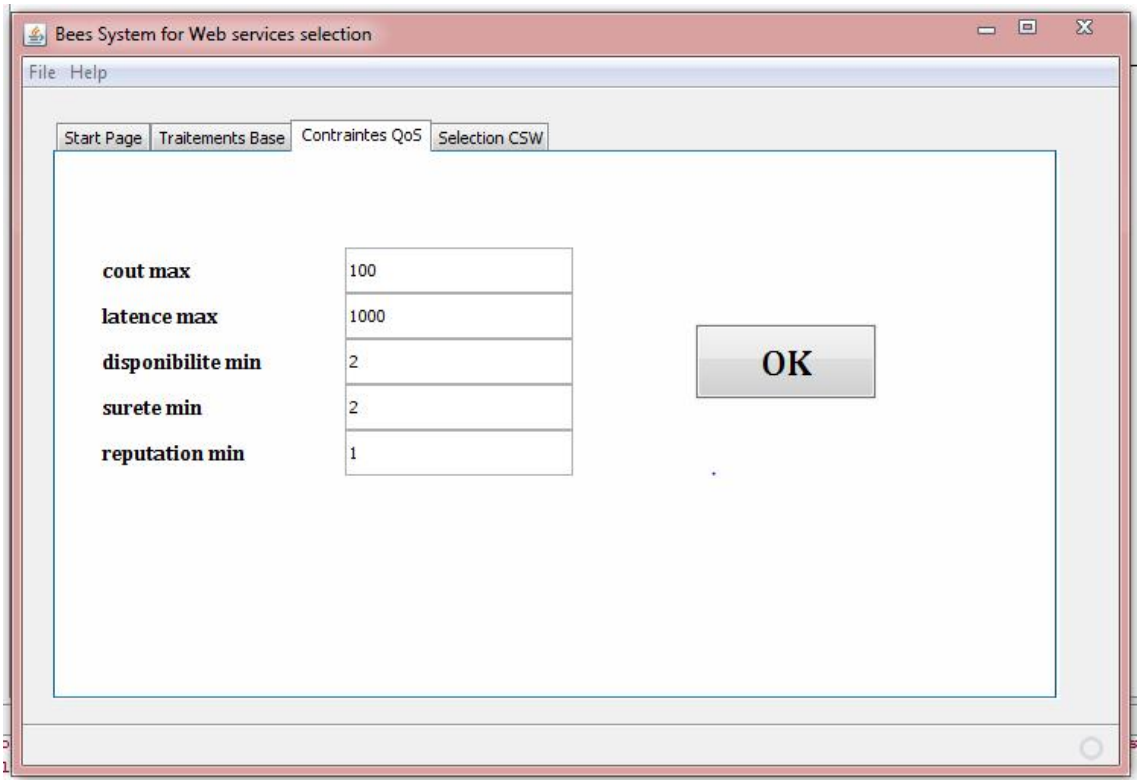


Figure II.8 : Contraintes globales de qualités de services

- Maintenant nous pouvons lancer la sélection de compositions de services Web, en précisant le nombre de simulations ainsi que le nombre maximum des itérations par simulation.

  L'algorithme d'optimisation « BeesAlgo » permet de faire une optimisation mono objective qui améliore les résultats obtenus en chaque itération, jusqu'à l'obtention d'une solution proche de l'optimum c.-à-d., une composition de services (Bee) qui satisfait au mieux les besoins de l'utilisateur :

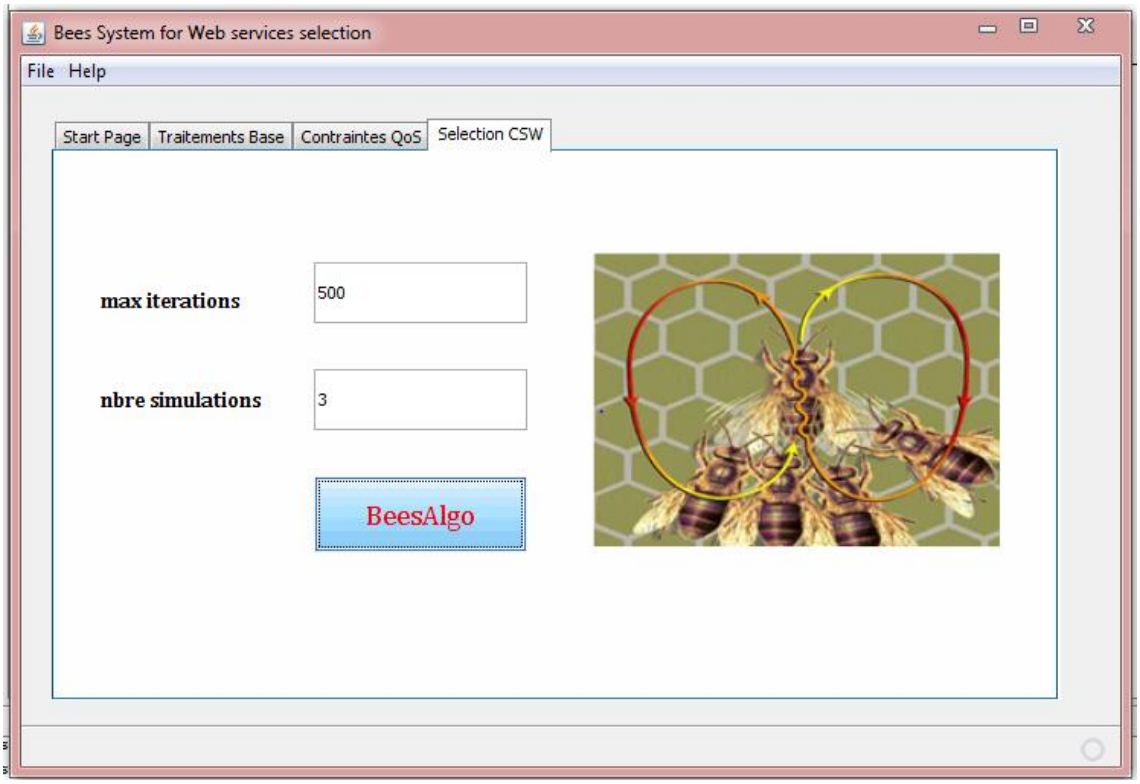


Figure II.9 : Bees système pour la sélection de SW

X. Expérimentations :

Nous présentons maintenant un échantillon des résultats obtenus au cours de nos expérimentations en considérant les configurations des paramètres système suivants :

X.1. Première configuration :

Nous proposons une configuration qui se base sur trois simulations et 500 itérations par simulation, sachant que nous gardons les mêmes paramètres d'entrées pour les différentes simulations comme montre le tableau II.3 au-dessous :

Paramètres	Valeurs	Description
Beesnum	60 Bees	taille de population initial = 60 abeilles, compositions,…
Sitesnum	50 Bees	meilleures compositions en terme de QoS
EliteSitesnum	30 Bees	sous group des abeilles Sitesnum
EliteBeesnum	7 RecruitedBees	les abeilles élues → 7 voisins
OtherBeesnum	2 RecruitedBees	les autres abeilles → 2 voisins
RemainingBeesnum	10 RandomBees	le reste de la nouvelle pop est généré aléatoirement.

Tableau II.3 : Les valeurs et la description des paramètres système.

Les résultats sont représentées par les histogrammes suivants, telle que le temps d'exécution est donnés par minute et le taux d'optimalités représente un pourcentage :

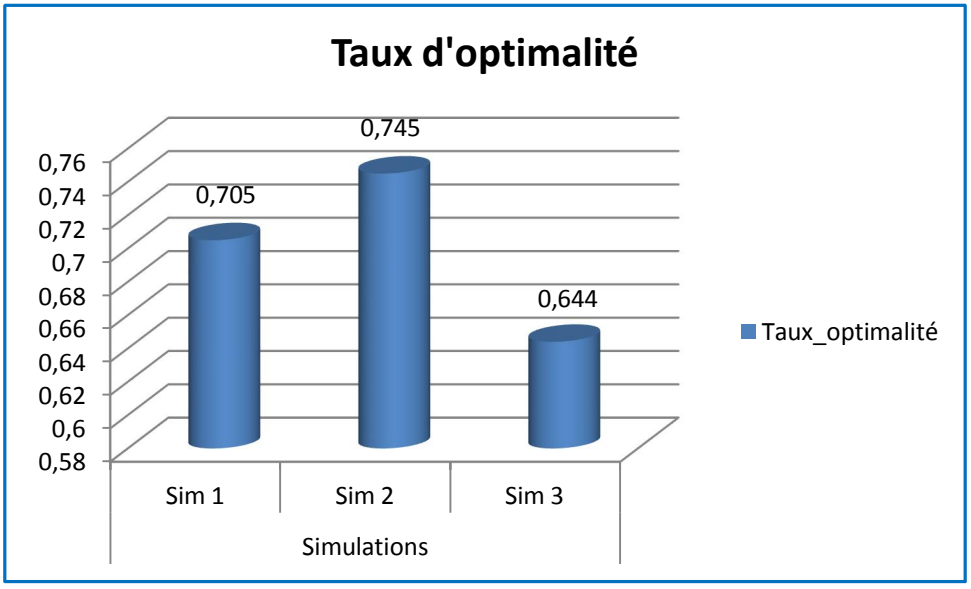


Figure II.10 : taux d'optimalité des 3 simulations.

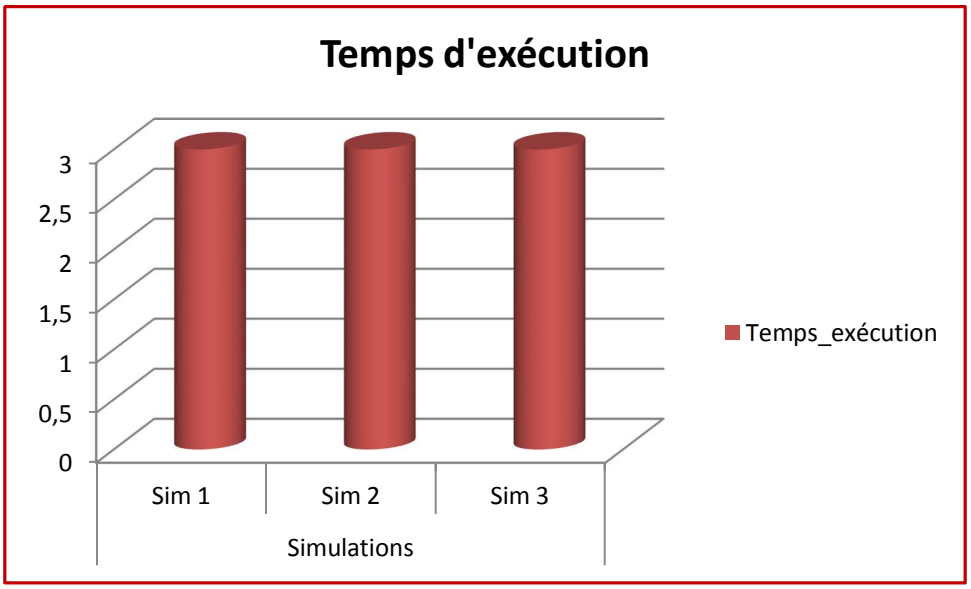


Figure II.11 : temps d'exécution des 3 simulations.

On remarque que le temps d'exécution ne dépasse pas les trois minutes pour toutes les simulations et on peut atteindre jusqu'à 75% comme taux d'optimalité.

	Taux_optimalité	Temps_exécution (min)
Sim 1	0,705	3
Sim 2	0,745	3
Sim 3	0,644	3

Tableau II.4 : résultats de la configuration une.

X.2. Deuxième configuration :

Nous proposons maintenant une autre configuration qui se base sur le nombre d'abeilles, nous avons fait 500 itérations pour chaque simulation en modifiant les paramètres d'entrées pour pouvoir comparer les résultats, voilà un tableau qui montre le paramétrage expérimenté :

	Sim 1	Sim 2	Sim 3
Beesnum	30	60	120
Sitesnum	25	50	100
EliteSitesnum	15	30	60
RemainingBeesnum	5	10	20

Tableau II.5 : paramètres de la configuration deux.

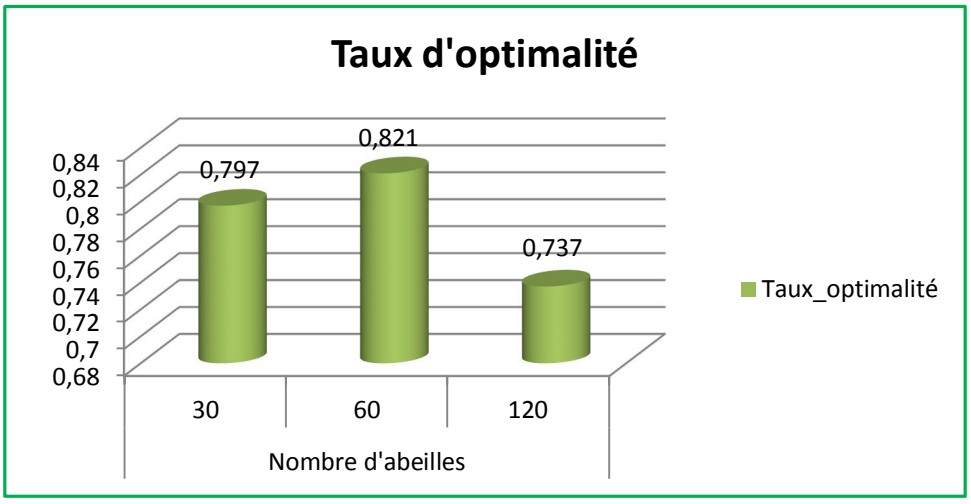


Figure II.12 : taux d'optimalité en fonction du nombre d'abeilles

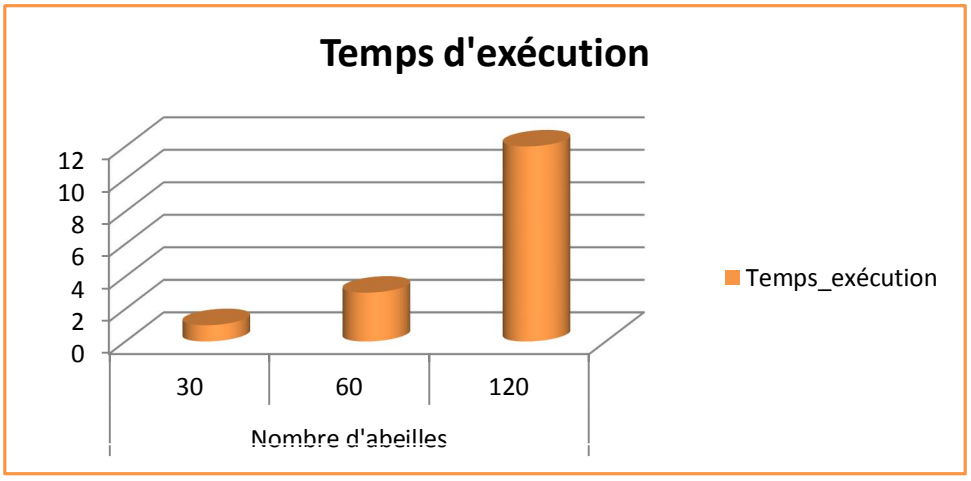


Figure II.13 : temps d'exécution en fonction du nombre d'abeilles

En changeant les paramètres du système d'abeilles on remarque une amélioration des résultats obtenus en termes d'optimalité.
Le temps d'exécution augmente lorsque le nombre d'abeilles de la population initiale est plus grand (120 Bees), et on peut atteindre jusqu'à 80% comme taux d'optimalité.
Voilà un tableau qui montre les résultats exacts obtenus :

	Taux_optimalité	Temps_exécution (min)
Sim 1	0,797	1
Sim 2	0,821	3
Sim 3	0,737	12

Tableau II.6 : résultats de la configuration deux.

XI. Discussion :

En discutant un petit peu les résultats obtenus par notre système d'abeilles pour la sélection de services Web :

Premièrement nous avons conclu que les résultats obtenus en chaque exécution sont relatives et on ne peut pas les devinés même si on garde les mêmes paramètres d'entrée on peut avoir des résultats différentes, qui peuvent s'améliorés comme peuvent décroitre, parce qu'une partie du déroulement de l'algorithme se base en quelque sorte sur l'aléatoire, en commençant par une population initiale aléatoire on tente de l'améliorée en passant par plusieurs traitements.

Mais tous dépend de ce point de départ, si c'est un mauvais point on peut jamais atteindre l'objectif, il n'y a pas un chemin entre notre point de départ et le point disant optimum. Par contre si on commence avec une bonne population initiale on peut avoir des bons résultats en un bon temps d'exécution.

Mais quand-même on peut contrôler les paramètres d'entrées du système, par exemple en augmentant la taille de la population on peut atteindre un taux d'optimalité qui dépasse les 80% mais ceci peut coûter en terme du temps, d'après les expérimentations qu'on a fait on se contente de « 60 Bees » pour un taux de 80% et un temps d'exécution raisonnable.

Aussi en maximisant les abeilles « Sitesnum » on minimise la notion de l'aléatoire dans le processus c.-à-d., les « RemainingBeesnum », par conséquent le taux

d'optimalité augmente mais ça va causer un traitement en plus et donc un temps d'exécution plus grand, et donc on essaie de garder l'équilibre entre ces deux paramètres.

XII. Conclusion :

Dans ce dernier chapitre nous avons proposé un algorithme d'optimisation basé sur les colonies d'abeilles « BeesAlgo » pour résoudre le problème de sélection de services Web, cet exemple d'application nous a permis de bien illustrer le travail, de présenter les résultats de la validation de notre approche, afin d'évaluer l'efficacité de cette dernière.

D'après nos résultats, nous avons confirmé l'efficacité de l'algorithme d'abeilles [Pham et al., 2006], dans le domaine de sélection de services Web, en effet les résultats obtenus sont acceptables et montrent leur supériorité par rapport à d'autres approches de l'état de l'art par exemple, recuit simulé et la recherche taboue.

Conclusion générale :

Les services Web sont des technologies émergentes et prometteuses pour le développement, le déploiement et l'intégration d'applications sur l'Internet. Ils constituent la technologie de base pour le développement d'architectures orientées services. Ces architectures sont de plus en plus répandues sur le Web. Le principe essentiel de l'approche service Web est de transformer le Web en un dispositif distribué d'échange et de calcul, où les services Web peuvent interagir d'une manière intelligente. Actuellement, de nombreux services Web, avec des fonctionnalités similaires sont fournis par des fournisseurs concurrents, et de ce fait les utilisateurs finaux ont besoins d'approches efficaces pour la sélection des services.

Dans ce mémoire, nous avons présenté un travail qui a pour but de faire une optimisation mono objective basée sur l'algorithme d'abeilles pour la sélection de ces compositions de services Web. Cette méthode exploite les relations de voisinage et des heuristiques à base de dance pour explorer efficacement les grands espaces de recherche.

D'après notre propre cas d'application et les résultats obtenus par d'autres études, les algorithmes à base colonies d'abeilles en général et le « BeesAlgo » en particulier fournissent de bons résultats pour les problèmes d'optimisations difficiles, on peut dire que cette méthode est mieux adaptée aux problèmes d'optimisation combinatoire.

Tout travail est amené à être amélioré, en ce sens, notre application peut encore évoluer et se voir améliorer. Pour une continuation de notre travail, plusieurs perspectives peuvent être envisagées :

- Il est possible de faire une optimisation multi objective ou hybride à base de colonies d'abeilles pour résoudre le même problème proposé dont le but d'améliorer et de varier les résultats finaux
- On peut aussi proposer des algorithmes qui calculent automatiquement les priorités en fonction de la répartition statistique des qualités de services.
- Comme on peut résoudre ce même problème par une autre méthode carrément, telle que les colonies de Fourmies, et l'optimisation par essaimes particulaires, …ce qui permet de comparer les performances des différentes approches.

[Alrifai et al, 2009] : M. Alrifai and T. Risse. Combining global optimization with local selection for efficient qos-aware service composition. In WWW April 20–24, 2009, Madrid, Spain. ACM 978-1-60558-487-4/09/04., pages 881–890, 2009.

[Alrifai, 2010] : 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.

[Alrifai et al., 2012] : Alrifai, M., Risse, T., and Nejdl, W. 2012. A hybrid approach for efficient Web service composition with endto- end QoS constraints. ACM Trans. Web 6, 2, Article 7 (May 2012), 31 pages. DOI = 10.1145/2180861.2180864 http://doi.acm.org/10.1145/2180861.2180864

[Arnaud, 2005] : Arnaud VEZAIN, Les service web - présentation générale, rapport technique, Association HERMES, Février 2005.

[Ardagna et al., 2005] : 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.

[Ardagna et al., 2007] : D. Ardagna and B. Pernici. Adaptive service composition in flexible processes. IEEE Transactions on Software Engineering, 33(6):369–384, 2007.

[Akbar et al., 2001] : 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.

[Akbar et al. 2006] : AKBAR, M. M., RAHMAN, M. S., KAYKOBAD, M., MANNING, E. G., AND SHOJA, G. C. 2006. Solving the multidimensional multiple-choice knapsack problem by constructing convex hulls. *Comput. Oper. Res. 33,* 5,1259–1273.

[Benatallah et al. 2002] : BENATALLAH, B., SHENG, Q. Z., NGU, A. H. H., AND DUMAS, M. 2002.Declarative composition and peer-topeerprovisioning of dynamic web services.In *Proceedings of the International Conference on DataEngineering*. IEEE, Los Alamitos, CA, 297–308.

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

[Bertino, 2004] : A. Bertino, M. Keidl, et E. Kemper, A Framework for Contextaware Adaptable Webservices, EDBT, LNCS 2992, Springer- Verlag Berlin Heidelberg, 2004.

[Bouguettaya et al, 2009] : Athman Bouguettaya, Qi Yu, Foundations for Efficient Web Service Selection, Springer Science and Business Media, LLC 2009.

[Bouguettaya et Qi, 2010] : Qi Yu, A Bouguettaya. Foundations for Efficient Web Service Selection Springer Science+Business Media, 2010.

[Benouaret et al, 2011 a] : Karim Benouaret, Djamal Benslimane, Allel HadjAli, Mahmoud Barhamgi: FuDoCS: A Web Service Composition System Based on Fuzzy Dominance for Preference Query Answering. PVLDB 4(12): 1430-1433 (2011).

[Benouaret et al, 2011 b] : Karim Benouaret, Djamal Benslimane, Allel HadjAli, Mahmoud Barhamgi: Top-k Web Service Compositions Using Fuzzy Dominance Relationship. IEEE SCC 2011: 144-151.

[Christensen et al., 2001] : E. Christensen, F. Curbera, G. Meredith, and S. Weerawarana, Web Services Description Language (WSDL) 1.1, rapport téchnique,W3C, http://www.w3.org/TR/wsdl , 2001.

[Coello et al., 2002] : Coello C.C.A., Van Veldhuizen D.A, Lamont G.B. Evolutionary Algorithms for Solving Multi-objective Problems. Kluwer Academic Publishers, New York (2002).

[Colan, 2003] : IBM.BPEL4WS (version1.1), http://www.ibm.com /developerworks/library/ws-bpel/, 2003.

[Clerc, 2003] : Clerc M. , TRIBES - Un exemple d'optimisation par essaim particulaire sans paramètres de contrôle, Conférence OEP'03, Paris, France, 2 Octobre, 2003.

[Clement et al., 2004] : Clement L., Hately A., von Riegen C. et Rogers T., UDDI v.3.0.2, OASIS Specification, http://uddi.org/pubs/uddi_v3, 2004.

[Eric et al., 2005] : Newcomer, Eric; Lomow, Greg, Understanding SOA with Web Services, Addison Wesley, 2005.

[Erl, 2005] : Thomas Erl, Service-oriented Architecture : Concepts, Technology, and Design, Prentice Hall, 2005.

[Hubert et al., 2003] : K. Hubert , M. Valérie , LES WEB SERVICES, Edition DUNOD, 2003.

[Justin, 2002] : O'Sullivan Justin, Edmond David, Ter Hofstede Arthur, What's in a service Distrib. Parallel Databases ?, 12(2-3): 117–133, 2002.

[Kennedy et Eberhart, 1995] : Kennedy J. , Eberhart R.C. ,Particle Swarm Optimization, Proceedings of the IEEE International Conference On Neural Networks, pages 1942-1948, IEEE Press, 1995.

[Khan, 1998] : KHAN, M. S. 1998. Quality adaptation in a multisession multimedia system: Model, algorithms, and architecture. Ph.D. dissertation.

[Kangchan Lee et al, 2003] : Kangchan Lee, Jonghong Jeon, Wonseok Lee, Seong-Ho Jeong, and Sang-Won Park. Qos for web services : Requirements and possible approaches. Technical report, W3C, Web Services Architecture Working Group, November 2003.

[Kritikos et Plexousakis, 2009] : KRITIKOS, K. AND PLEXOUSAKIS, D. 2009. Mixed-integer programming for QoS-based web service matchmaking. *IEEE Trans. Services Comput. 2,* 2, 122–139.

[Li et al. 2007] : LI, F.,YANG, F.,SHUANG, K., AND SU, S. 2007. Q-peer: A decentralized QoS registry architecture for web services. In *Proceedings of the International Conference on Services Computing*.145–156.

[Mitra et al., 2003] : Mitra N. et Lafon Y, SOAP Version 1.2 Part 0: Primer (Second Edition), W3C, http://www.w3.org/TR/soap12-part0/, 2003.

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

[Nemhauser et Wolsey, 1988] : NEMHAUSER, G. L. AND WOLSEY, L. A. 1988. *Integer and Combinatorial Optimization*. Wiley, New York.

[OASIS, 2006] : OASIS Reference Model for Service Oriented Architecture, Committee Specification, 2006.

[Pham et al., 2006] : D. T. Pham, Ghanbarzadeh A., Koc E., Otri S., Rahim S., and M.Zaidi. The bees algorithm - a novel tool for complex optimisation problems. In *Proceedings of IPROMS 2006 Conference*, pages 454–461, 2006.

[Pisinger, 1995] : D. Pisinger. Algorithms for Knapsack Problems. PhD thesis, University of Copenhagen, Dept. of Computer Science, February 1995.

[Rocques, 2007] : Rocques P. et Vallées F., « UML 2 en action, de l'analyse des besoins à la conception », Eyrolles, 2007.

[Vogel et al, 1995] : Andreas Vogel, Brigitte Kerhervé, Gregor von Bochmann, and Jan Gecsei. Distributed multimedia and qos : A survey. IEEE MultiMedia, 2(2) :10–19, 1995.

[Von et Castro, 2001] : 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.

[W3C-WSA-Group, 2004] : W3C-WSA-Group, W3C Web Service Architecture Group, Web Services Architecture. http://www.w3.org/TR/ws-arch, 2004.

[Yu et al., 2007] : T. Yu, Y. Zhang, and K.-J. Lin. Efficient algorithms for web services selection with end-to-end qos constraints. ACM Trans. on the Web, 1(1), 2007.

[Zeng et al. 2003] : L. Zeng, B. Benatallah, M. Dumas, J. Kalagnanam, and Q. Z. Sheng. Quality driven web services composition. In WWW, pages 411–421, 2003.

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

[Zhai et al. 2009] : ZHAI, Y., ZHANG, J., AND LIN, K.-J. 2009. Soa middleware support for service process reconfiguration with endto- end QoS constraints. In *Proceedings of the IEEE International Conference on Web Services (ICWS'09)*. IEEE, Los Alamitos, CA, 815–822.