



REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Mohamed Khider – BISKRA
Faculté des Sciences Exactes, des Sciences de la Nature et de la Vie
Département d'informatique

N° d'ordre : IA22/M2/2019

Mémoire

Présenté pour obtenir le diplôme de master académique en

Informatique

Parcours : Intelligence Artificielle (IA)

Une Approche Effective Pour La Sélection Des Services Web Composites

Par :

DERRADJI HAZAR

Soutenu le 06 juillet 2019, devant le jury composé de :

Zerarka	Mohamed Faouzi	MCA	Président
Ben Seghier	Nadia	MCB	Rapporteur
Chami	Djazia	MAA	Examineur

Remerciements

En premier lieu, je remercie ALLAH le Tous Puissant de m'avoir donné la force, le courage et la patience nécessaire afin d'achever ce travail.

Je tiens à remercier mon encadreur Mme Ben Seghier Nadia pour ses précieux conseils, sa disponibilité et son soutien tout au long de mon travail.

Ma gratitude s'adresse aux membres de jury qui nous font honneur en acceptant d'évaluer notre travail.

Comme je remercie énormément le destin de m'avoir donnée la chance et l'honneur d'être juré encore une deuxième fois par mon enseignant et mon président de juré M.ZERARKA MOHAMED FAOUZI après 20 ans de retour, tous mon respect Monsieur.

Mes profonds remerciements à mon Cher Mari pour sa patience et son soutien moral incessant surtout pendant les moments difficile tout au long des années de mes études.

Un grand remerciement pour ma famille ; mes collègues de travail, ainsi mes plus sincères remerciements vont également aux très chères à mon cœur Ikram , Latrach Imene et Sana qui mon aidées a achevé ce travail, sans oublie Adel pour son encouragement, ainsi tous mes collègues de promotion 2019.

Un très grand merci à l'ensemble de tous les enseignants du département d'informatique de l'université de Biskra pour leurs efforts et leurs compréhensions.

Nombreux sont ceux qui m'ont aidé, encouragé, réconforté, soulagé, tout au long de ces années et je ne saurais leur exprimer mes remerciements autant que je le souhaiterais ; leurs discussions, leurs conseils, leur présence, me furent éminemment précieux.

Enfin, un salut particulier à tous ceux qui m'ont soutenu de près ou de loin pour réaliser ce travail.

Merci

Dédicace

Je dédié ce travail

A toute ma famille

A tous mes amis

A tous ceux qui m'aime

Et tous ceux que j'aime.

Hazar

Résumé

Le marché de commercialisation des services Web sur internet ne cesse d'augmenter, ce qui résulte en un nombre de plus en plus croissant de services offrant des fonctionnalités équivalentes. De ce fait, la sélection d'un service web approprié pour une tâche particulière est devenue un défi difficile pour l'utilisateur. Cependant, ces services diffèrent dans leurs qualités de services (QoS). Ces dernières deviennent alors cruciales pour l'utilisateur, car elles permettent de l'aider à choisir parmi les services Web qui sont fonctionnellement équivalents. De plus, il devient difficile de satisfaire l'exigence complexe d'un utilisateur par juste un seul service web individuel ou atomique, ce qui a conduit les concepteurs à composer les services Web existants. Par conséquent, la sélection n'est plus le simple choix d'un seul service atomique parmi tant d'autres pour une seule tâche, mais la sélection devient une opération complexe qui consiste à combiner un ensemble de services choisis pour chaque tâche, afin de former la composition qui satisfait au mieux les critères QoS de l'utilisateur.

Nous nous intéressons, dans ce mémoire, à la problématique de sélection des services Web dans une composition sur la base des besoins non fonctionnels (QoS). Notre travail consiste à développer une application basée sur une approche d'optimisation multi-objective pour la sélection des meilleurs services Web en fonction de QoS. Nous avons utilisé trois méthodes pour la conception de notre approche : Top-k, Skyline, et Top-k Skyline. Ces trois méthodes sont appliquées en utilisant la dominance floue.

Mots clés : Sélection des services Web, Compositions de services, Top-k, Skyline, Optimisation combinatoire, Dominance floue, Qualité de service.

Abstract :

The marketing market for web-based services on the internet continues to grow, resulting in an increasing number of services offering equivalent functionality. As a result, selecting a suitable web service for a particular task has become a difficult challenge for the user. However, these services differ in their quality of service (QoS). These then become crucial for the user, as they help him choose among web services that are functionally equivalent. Moreover, it becomes difficult to satisfy the complex requirement of a user by just one individual or atomic web service, which has led the designers to dial the existing web services. Therefore, selection is no longer the simple choice of one atomic service out of many for a single task, but selection becomes a complex operation of combining a set of services chosen for each task, in order to form the composition that best meets the QoS criteria of the user.

In this thesis, we are interested in the problem of selecting Web services in a composition based on non-functional needs (QoS). Our job is to develop an application based on a multi-objective optimization approach for the selection of the best web services based on QoS. We used three methods for designing our approach: Top-k, Skyline, and Top-k Skyline. These three methods are applied using fuzzy dominance.

Keywords: Selection of Web Services, Service Compositions, Top-k, Skyline, Combinatorial Optimization, Fuzzy Dominance, Quality of Service.

الملخص :

يستمر سوق التسويق للخدمات القائمة على شبكة الإنترنت في النمو ، مما أدى إلى زيادة عدد الخدمات التي توفر وظائف مكافئة. نتيجة لذلك ، أصبح اختيار خدمة ويب مناسبة لمهمة معينة تحديًا صعبًا للمستخدم. ومع ذلك ، تختلف هذه الخدمات في جودة الخدمة (جودة الخدمة). ثم تصبح هذه الأمور مهمة للمستخدم ، لأنها تساعد في الاختيار بين خدمات الويب المتكافئة وظيفيًا. علاوة على ذلك ، يصبح من الصعب تلبية المتطلبات المعقدة للمستخدم من خلال خدمة ويب فردية أو ذرية واحدة فقط ، مما دفع المصممين إلى طلب خدمات الويب الحالية. لذلك ، لم يعد الاختيار هو الاختيار البسيط لخدمة ذرية واحدة من بين العديد لمهمة واحدة ، ولكن يصبح الاختيار عملية معقدة تجمع بين مجموعة من الخدمات المختارة لكل مهمة ، من أجل تشكيل التكوين الذي يلبي معايير جودة الخدمة للمستخدم.

في هذه الأطروحة ، نحن مهتمون بمشكلة اختيار خدمات الويب في تكوين يعتمد على الاحتياجات غير الوظيفية (QoS). مهمتنا هي تطوير تطبيق يستند إلى نهج تحسين متعدد الأهداف لاختيار أفضل خدمات الويب القائمة على جودة الخدمة. استخدمنا ثلاث طرق لتصميم منهجنا: Top-k و Skyline و Top-k Skyline. يتم تطبيق هذه الطرق الثلاث باستخدام هيمنة غامضة.

الكلمات الرئيسية: اختيار خدمات الويب ، تكوينات الخدمة ، Top-k ، Skyline ، التحسين التوافقي ، الهيمنة المشوشة ، جودة الخدمة.

Table de Matière

REMERCIEMENT.....	I.
DEDICACE.....	II
RESUME	III
TABLE DE MATIERE.....	IV
TABLE DES FIGURES.....	V
LISTE DES TABLEAUX.....	VI
LISTE DES ALGORITHMES.....	VII
INTRODUCTION GÉNÉRALE.....	3
<i>C HAPITRE 1 . LES SERVICES WEB.....</i>	<i>5</i>
1.1 Introduction.....	6
1.2 Définition de Service Web.....	6
1.3 Intérêt des services web.....	6
1.4 Architecture Orientée Services.....	7
1.5 Fonctionnement et Architecture des Services Web.....	8
1.5 .1 Les Technologies des Services Web.....	8
1.5.2 Fonctionnement des services Web.....	9
1.5.3 Architecture en Couches des Services Web.....	10
1.6 Avantages des Services Web.....	12
1.7 Inconvénients des Services Web	13
1.8 Quelques domaines d'Application de Services Web.....	13
1.9 Les Propriétés d'un Service Web	14

«Table de Matière »

1.9. 1 Les Propriétés Fonctionnelles	14
1.9. 2 Les Propriétés non- Fonctionnelles	14
1.10 Qualité de Service (QoS).....	14
1.11 Conclusion.....	15
<i>C H A P I T R E 2. LA SÉLECTION DES SERVICES WEB À BASE QOS.....</i>	<i>16</i>
2.1 Introduction	17
2.2 Technique de Sélection de Service Web.....	17
2.2.1 Approches Exactes.....	18
2.2.2 Approche Heuristique (Approximative).....	18
2.2.3 Approche Méta-Heuristique.....	19
2.2.4 Les Méta-Heuristiques à Base de Solution Unique.....	19
2.2.5 Méta-Heuristiques à Base de Population de Solution.....	19
2.2.6 Approches basées sur les techniques de bases de données.....	19
2.2.6.1 La Requête Skyline	19
2.2.6.2 La Requête Top-K.....	21
2.2.6.3 La Méthode Hybride	21
2.3 Travaux Connexes.....	22
2.3 .1 Approches Exactes.....	22
2.3.2 Approche Heuristique (Approximative).....	22
2.3.3 Méta-Heuristiques à Base de Population de Solution.....	23
2.3.4 Approches basées sur les Techniques de Bases de Données.....	23
2.4 Conclusion.....	25

«Table de Matière »

C H A P I T R E 3 . C O N C E P T I O N D U S Y S T È M E.....	26.
3.1 Introduction	27
3.2 Architecture Générale du Système.....	27
3.3 Exemple Illustratif	28
3.4 La Sélection de Services Composites.....	29
3.4.1 La Sélection des Top-k Services basée sur la Dominance Floue.....	30
3.4.2 La Sélection des Services Skyline basée sur la Dominance Floue.....	35
3.4.3 La Sélection Hybride des Services basée sur la Dominance Floue.....	38
3.5 Conclusion	39
C H A P I T R E 4 . I M P L É M E N T A T I O N D U S Y S T È M E (É T U D E D E C A S).....	40
4.1 Introduction	41
4.2 Environnement de Développement.....	41
4.2.1 . Le langage JAVA.....	41
4.2.2 . Eclipse.....	43
4.2.3 . Gestion de Base de Données (phpMyAdmin)	44
4.3 Présentation des Interfaces Graphiques.....	45
4.3.1 Interface d'accueil.....	45
4.3.2 Interface Principale.....	46
4.3.3 Chargement de la Base de Données.....	46
4.4. Etude de Cas 1.....	49
4.4.1 Algorithme FTop-KS « Top-K en utilisant la Dominance Floue ».....	49
4.4.2 Algorithme α -DSkyS « Skyline en utilisant la Dominance Floue ».....	50
4.4.3 Algorithme Hybride en utilisant la Dominance Floue.....	50

«Table de Matière »

4.5. Etude de Cas 2.....	50
4.6 Conclusion	54
CONCLUSION ET PERSPECTIVES.....	56
BIBLIOGRAPHIES.....	58

Table des Figures

Figure 1.1 : Modèle Fonctionnel de l'architecture SOA	7
Figure 1.2 : Architecture des composants des services web.....	8
Figure 1.3 : Fonctionnement d'un Service Web.....	10
Figure 1.4 : La pile des couches des Services Web	12
Figure 2.1 : Approches de sélection de services web à base QoS.....	17
Figure 2.2 : Approches d'Optimisation du Problème QOSWSC.....	18
Figure 2.3 : Exemple de Fuzzification.....	22
Figure 3.1 : Architecture Générale du Système.....	28
Figure 3.2 : Sélection des Services Composites.....	30
Figure 4.1 : Environnement logiciel utilisé.....	41
Figure 4.2 : Interface principale de Eclipse.....	44
Figure 4.3 : Interface Serveur phpMyAdmin.....	45
Figure 4.4 : Interface d'Accueil.....	45
Figure 4.5 : Interface Principale.....	46.
Figure 4.6 : Chargement des Bases Données.....	46
Figure 4.7 : la Requête Demandée par l'Utilisateur.....	47
Figure 4.8 : Liste des Services candidats de la Classe reserverhotel.....	47
Figure 4.9 : Le Cas si aucune Base correspond à la requête lancer.....	48

Table des Figures

Figure 4.10 : Les Services Candidats de la classe reserverhotel avec leurs QOS.....	49
Figure 4.11 :Algorithme FTop-KS « Top-K en utilisant la Dominance Floue ».....	49
Figure 4.12 : Algorithme α -DSkyS-SFS« Skyline en utilisant la Dominance Floue ».....	50
Figure 4.13 :Algorithme Hybride 'FTop-KSky ' en utilisant la Dominance Floue.....	50
Figure 4.14 :Exemple 1.....	51
Figure 4.15 :Exemple 2.....	52
Figure 4. 16 : Exemple 3.....	52.
Figure 4.17 :Résultat de l'Algorithme Hybride ' <i>FTop-KSky</i> ' en utilisant la Dominance Floue.....	54

«Liste des Tableaux »

Liste des Tableaux :

Tableau 1.1 : Description en couche des services Web.....	11
Tableau 2.1 : Exemple l'Algorithme de Skyline.....	20
Tableau 3.1 : Les Services Web de la Classe reserverhotel.....	29
Tableau 3.2 : Les Valeurs des QoS des Services Web après Normalisation.....	31
Tableau 3.3 : Le Classement des Services selon FDedS().....	35

«Liste des Algorithmes »

Liste des Algorithmes :

Algorithme 3.1 : 'FTop-KS' Algorithme pour la Sélection des K meilleurs Service basé sur la Dominance Floue.....	34
Algorithme 3.2 : 'α-DSkyS-SFS' Algorithme de calcul des Skylines basé sur FDed() et SFS	37
Algorithme 3.3 : 'FTop-KSky' Algorithme pour la sélection des Top-K Skyline services basé sur la Dominance Floue.....	38

Introduction

générale

Introduction générale

Avec l'essor du web qui a eu lieu dans les dernières années, le besoin de permettre qu'une application client invoque un service d'une application serveur en utilisant Internet, a surgi. Ce même besoin a été l'origine de ce qu'on appelle communément les services web. En tenant en compte que les services web permettent de connecter des applications différentes, l'utilité de cette technologie devient évidente et importante. Pour cette raison les activités de la recherche et développement autour du sujet services Web, ont un dynamisme très haut. La présence d'un ensemble de services web similaires d'un point de vue fonctionnel, poussent les utilisateurs à faire des choix à l'aide des critères de QoS, ces critères auront une grande importance lorsque le besoin des clients est complexe.

La sélection est une étape cruciale de laquelle dépend le résultat de la composition. Cette sélection n'est pas évidente, car il s'agit de choisir des services Web parmi un nombre important d'alternatives de services proposés par la phase de découverte.

Nous nous intéressons dans ce mémoire plus particulièrement à la phase de sélection des services qui permet de fournir une composition de services Web. La sélection à base de Qualités de Services consiste à choisir parmi les services Web découverts de chaque tâche, ceux qui répondent au mieux aux exigences de l'utilisateur sur la base des besoins non fonctionnels de QoS. La sélection de services Web basée sur la QoS dépend de la spécification adoptée lors de la définition des critères QoS et du profil QoS du service Web.

Des nouveaux travaux sur la résolution du problème de sélection de services Web ont adopté des approches qui reposent sur les techniques avancées des bases de données pour le traitement de requêtes complexes. Effectivement, deux grandes techniques ont reçu une importance croissante de la part des chercheurs et ont fait l'objet de plusieurs travaux : les requêtes Top-k et les requêtes Skyline. Ces derniers représentent des outils performants pour analyser des données multicritères, ce qui facilite la prise de décisions intelligentes face à des données de grande dimension, où l'utilisateur a des difficultés pour formuler ses préférences. Elles permettent d'extraire l'ensemble des points les plus pertinents quand différents critères, souvent conflictuels, sont pris en compte. Elles s'appuient sur le principe de dominance de Pareto.

«Introduction générale

Pour une sélection optimale des services en fonction de leurs (QoS), nous avons proposé une méthode de sélection des services web composites basé sur la technique orientée base de données en utilisant la dominance floue. On a utilisé trois algorithmes : l'algorithme 'FTop-KS' pour sélectionner les Top-k services Web basée sur la dominance floue, l'Algorithme ' α -DSkyS' qui utilise la requête Skyline basée sur la dominance floue. Ainsi un algorithme hybride de deux techniques Top-k et Skyline qui repose sur la solution du premier algorithme ' α _DSKyS' après on applique l'algorithme 'FTop-KS'.

Nous avons entrepris notre étude selon les quatre chapitres suivants :

Le premier chapitre est consacré aux notions fondamentales des Services Web où nous avons défini l'architecture SOA, l'intérêt de Service web, leur technologies, leur standards de base, ainsi leur fonctionnements et architecture. leurs avantages et inconvénients, nous présentons aussi quelque domaine d'application, et ensuite nous donnons notions sur qualité de service (QdS ou QoS en anglais : Qualité of Service).

Dans le **deuxième chapitre**, nous allons présenter les différentes classes des approches d'optimisation de la sélection des services web dans une composition ainsi les concepts de base de méthode proposée, enfin ; nous citons les travaux connexes.

Le troisième chapitre représente la conception de notre système, son architecture et ces fonctionnalités principales. Et la méthode fuzzification afin de sélectionner le service web le plus fiable à la requête de l'utilisateur.

Le quatrième chapitre traite la mise en œuvre du système, son environnement de développement (les outils utilisés) et l'implémentation de notre système.

Nous terminerons notre mémoire par **une conclusion générale** et nous énoncerons quelques perspectives de recherche.

Chapitre 1
Les Services
Web

1.1 Introduction

Dans ce chapitre, nous décrivons les Services Web en mettant en évidence la définition, l'intérêt et l'architecture SOA(Service-Oriented-Architecture), ainsi les technologies des Services Web. Nous commencerons par étudier le concept des services web ainsi que son fonctionnement, Architecture SOA en couches, et les Principaux standards utilisé dans les services Web, enfin les avantages et les inconvénients des services web. Nous mettons aussi l'accent sur l'aspect non fonctionnel des services Web en présentant quelques paramètres de QoS. Et nous terminant notre chapitre par une conclusion.

1.2 Définition de Service Web

- Le groupe de W3C [1] qui travaille sur les services Web a défini le service Web comme *« un système logiciel destiné à supporter l'interaction ordinateur-ordinateur sur le réseau. Il a une interface décrite en un format traitable par l'ordinateur (e.g. WSDL). Autres systèmes réagissent réciproquement avec le service web d'une façon prescrite par sa description en utilisant des messages SOAP, typiquement transmis avec le protocole http et une sérialisation XML, en conjonction avec d'autres standards relatifs au web »*.
- Selon [2] : *« 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 [3] : *« 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 »*.

1.3 Intérêt des Services Web

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

- Ils sont indépendamment de la plate-forme (UNIX, Windows, . . .)
- l'implémentation (VB, C#, Java, . . .)
- l'architecture sous-jacente (.NET, J2EE, Axis. . .)
- Les Web services sont basés sur les protocoles et les langages du Web, en particulier HTTP et XML.
- Le cadre des Web services contient en lui-même toutes les informations nécessaires à l'utilisation des applications, sous la forme de trois fonctions : trouver, décrire et exécuter.

1.4 Architecture Orientée Services

L'architecture orientée services (SOA) est une architecture logicielle s'appuyant sur un ensemble de composants simples appelés **Services Web**. Son objectif est de décomposer une fonctionnalité complexe en un ensemble de fonctionnalités basiques, fournies par des services Web et de décrire finement le schéma d'interaction entre ces services Web. L'architecture SOA présentée dans **Figure 1.1** implique trois entités d'interaction : les fournisseurs de services, les annuaires de service et les demandeurs des services auxquels nous rajoutons les attributs [4].

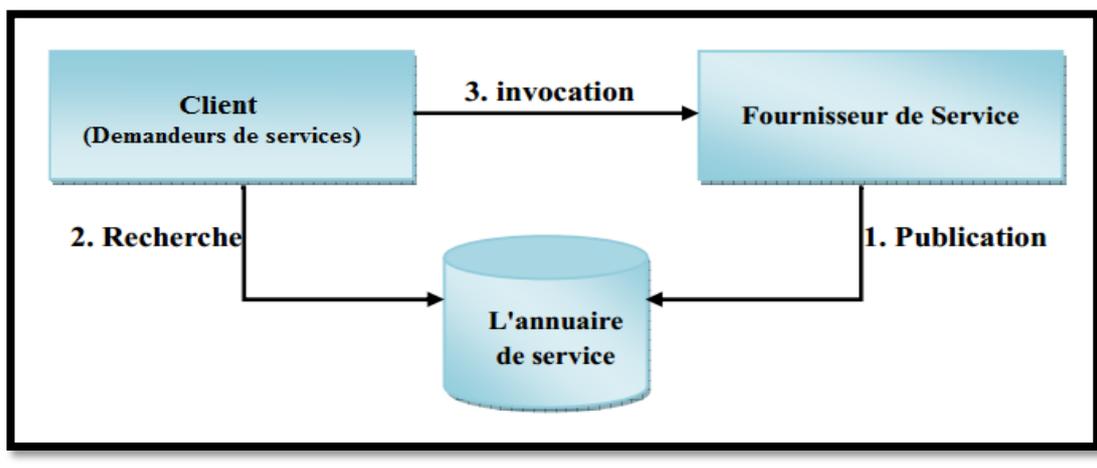


Figure 1.1 : Modèle Fonctionnel de l'architecture SOA [4].

L'architecture SOA comprend trois acteurs [5] :

- **Le fournisseur du service** : il désigne l'entité propriétaire du service. D'un point de vue technique, un fournisseur est constitué par la plate-forme d'accueil du service.
- **Le client ou demandeur du service** : C'est le consommateur de service. D'un point de vue technique, le service client est constitué de l'application qui va rechercher et invoquer un service.
- **L'annuaire de service** : Correspond à un registre de descriptions de services offrant des facilités de publication de services à l'intention des fournisseurs ; et des facilités de recherche de services à l'intention des clients. En d'autres termes, l'annuaire joue le rôle d'intermédiaire entre les clients et les fournisseurs de services.

1.5 Fonctionnement et Architecture des Services Web

Les services Web reprennent la plupart des idées et des principes du Web (HTTP, XML), et les appliquent à des interactions entre machines. Comme pour le World Wide Web, 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.

1.5.1 Les Technologies des Services Web

Les services Web emploient un ensemble de technologies qui ont été conçues afin de respecter une structure en couches sans être dépendante de façon excessive de la pile des protocoles.

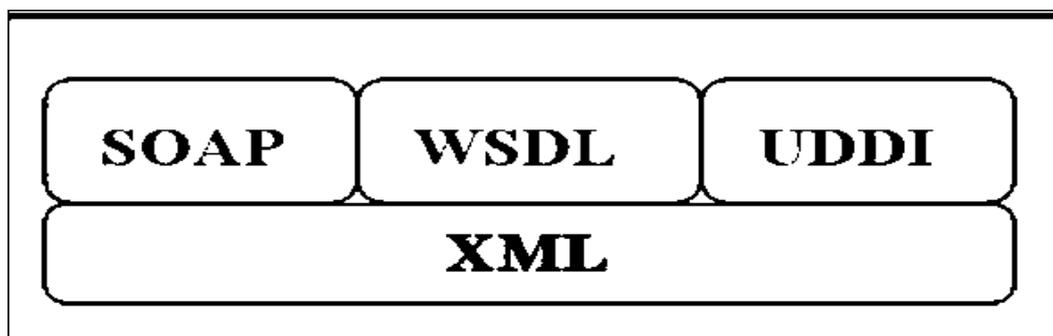


Figure 1.2: Architecture des composants des services web.

On présente les technologies utilisées pour les services web illustré dans la *Figure 1.2* comme suit :

XML-RPC : XML-RPC est un protocole simple utilisant XML pour effectuer des messages RPC. Les requêtes sont écrites en XML et envoyées via HTTP POST. Les requêtes sont intégrées dans le corps de la réponse HTTP. XML-RPC est indépendant de la plate-forme, ce qui lui permet de communiquer avec diverses applications.

SOAP : SOAP (Simple Object Access Protocol) est un protocole standard de communication. C'est l'épine dorsale du système d'interopérabilité. SOAP est un protocole décrit en XML et standardisé par le W3C. Il se présente comme une enveloppe pouvant être signée et pouvant contenir des données ou des pièces jointes. Il circule sur le protocole HTTP et permet d'effectuer des appels de méthodes à distance.

WSDL : WSDL (Web Services Description Language) est un langage de description standard. C'est l'interface présentée aux utilisateurs. Il indique comment utiliser le service Web et comment interagir avec lui. WSDL est basé sur XML et permet de décrire de façon précise les détails concernant le service Web tels que les protocoles, les ports utilisés, les opérations pouvant être effectuées, les formats des messages d'entrée et de sortie et les exceptions pouvant être envoyées.

UDDI : UDDI (Universal Description, Discovery and Integration) est un annuaire de services. Il fournit l'infrastructure de base pour la publication et la découverte des services Web. UDDI permet aux fournisseurs de présenter leurs services Web aux clients. Les informations qu'il contient peuvent être séparées en trois types :

- les pages blanches qui incluent l'adresse, le contact et les identifiants relatifs au service Web.
- les pages jaunes qui identifient les secteurs d'affaires relatifs au service Web .
- les pages vertes qui donnent les informations techniques.

1.5.2 Fonctionnement des Services Web

Le fonctionnement des services Web s'articule autour de trois acteurs principaux illustrés dans la *Figure 1.3* [6] :

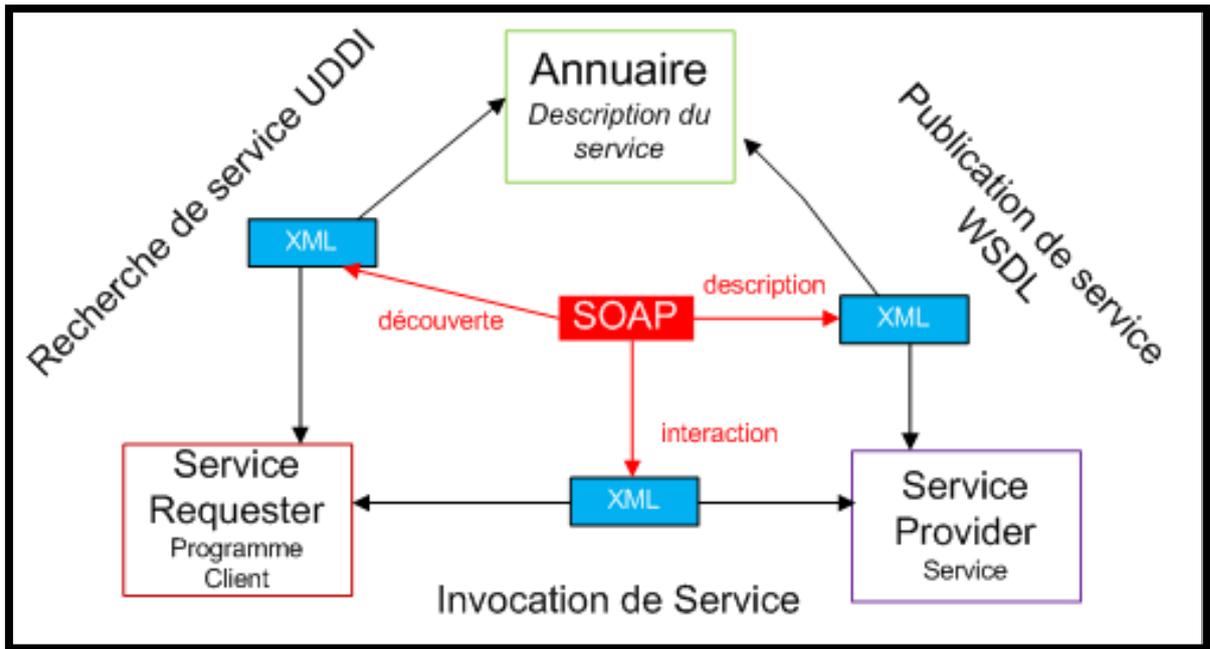


Figure 1.3: Fonctionnement d'un Service Web.

1. **Service provider service** : Le fournisseur de service met en application le service Web et le rend disponible sur Internet.
2. **Service requester programme client** : C'est n'importe quel consommateur du service Web. Le demandeur utilise un service Web existant en ouvrant une connexion réseau et en envoyant une demande en XML (REST, XML-RPC, SOAP).
3. **Annuaire service registry** : Le registre de service est un annuaire de services. Le registre fournit un endroit central où les programmeurs peuvent publier de nouveaux services ou en trouver. Les interactions entre ces trois acteurs suivent plusieurs étapes [6] :
 - **La publication du service** : le fournisseur diffuse les descriptions de ses services Web dans l'annuaire.
 - **La recherche du service** : le client cherche un service particulier, il s'adresse à un annuaire qui va lui fournir les descriptions et les URL des services demandés afin de lui permettre de les invoquer.
 - **L'invoication du service** : une fois que le client récupère l'URL et la description du service, il les l'utilise pour l'invoquer auprès du fournisseur de services.

1.5.3 Architecture en Couches des Services Web

Différentes extensions de l'architecture de base ont été proposées dans la littérature. Le groupe architecture du W3C travaille activement à l'élaboration d'une architecture

étendue standard. Une architecture étendue est constituée de plusieurs couches se superposant les unes sur les autres, d'où le nom de pile des Web services. La **Figure 1.4** décrit un exemple d'une telle pile. La pile est constituée de plusieurs couches, chaque couche s'appuyant sur un standard particulier. On retrouve, au-dessus de la couche de transport, les trois couches formant l'infrastructure de base décrite précédemment. Ces couches s'appuient sur les standards émergents SOAP, WSDL et UDDI comme illustrer dans le **Tableau 1.1**

Couche	Technologie	Rôle
Découverte de services	UDDI	La publication et la découverte des services web sont assurées par le biais du référentiel UDDI, un référentiel UDDI est un catalogue de services Web.
Description de services	WSDL	La description d'un service Web se fait en utilisant le langage WSDL. Il expose l'interface du service.
Communication	SOAP	SOAP prévoit la couche de communication basée sur XML pour accéder à des services Web.
Transport	HTTP	Le transport de messages SOAP est assuré par le standard HTTP.

Tableau 1.1 : Description en couche des services Web.

- 1. Couche transport** : Cette couche est responsable du transport des messages XML échangés entre les applications. Actuellement, cette couche inclut HTTP, SMTP, FTP, et de nouveaux protocoles tels que BEEP.
- 2. Couche communication** : Cette couche est responsable du formatage des données échangées de sorte que les messages peuvent être compris à chaque extrémité. Actuellement, deux styles architecturaux totalement différents sont utilisés pour ces échanges de données. Nous avons d'un côté l'architecture orientée opérations distribuées (protocoles RPC) basée sur XML et qui comprend XML-RPC et SOAP et de l'autre côté une architecture orientée ressources Web, REST (Représentationnel State Transfer) qui se base uniquement sur le bon usage des principes du Web (en particulier, le protocole HTTP).
- 3. Couche description de service** : Cette couche est responsable de la description de l'interface publique du service Web. Le langage utilisé pour décrire un service Web

est le WSDL qui est la notation standard basée sur XML pour construire la description de l'interface d'un service. Cette spécification définit une grammaire XML pour décrire les services Web comme des ensembles de points finaux de communication (ports) à travers lesquels on effectue l'échange de messages.

4. **Couche publication de service** : Cette couche est chargée de centraliser les services dans un registre commun, et de simplifier les fonctionnalités de recherche et de publication des services Web. Actuellement, la découverte des services est assurée par un annuaire UDDI (Universal Description, Discovery, and Integration).

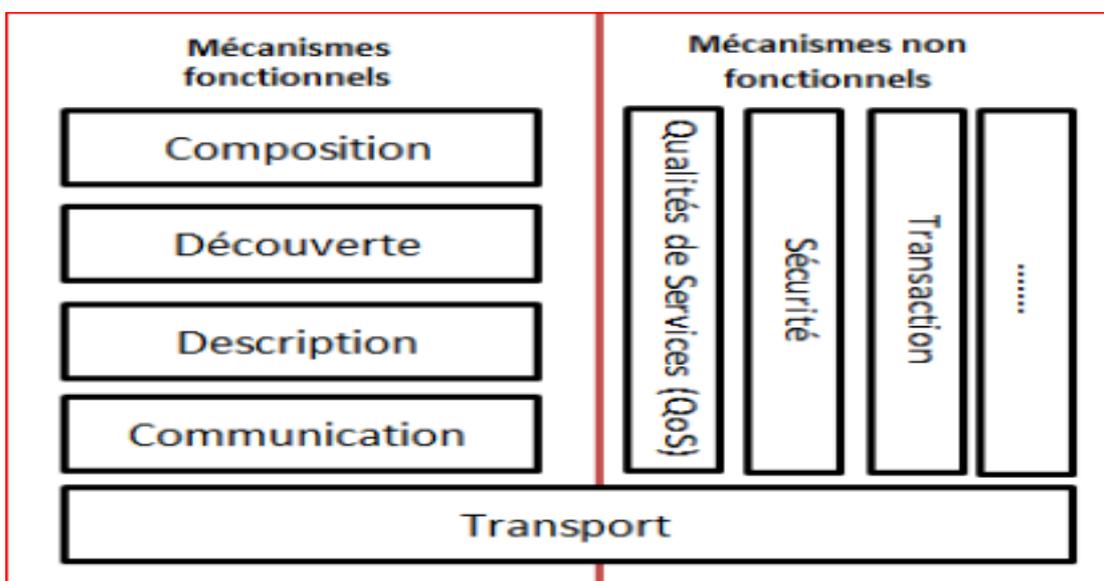


Figure 1.4 : La pile des couches des Services Web [7].

1.6 Avantages des Services Web

La technologie des services web est populaire et couramment utilisée car elle offre des avantages intéressants pour les utilisateurs des systèmes distribués :

- Les services web réduisent le temps de mise en marché des services offerts par les diverses entreprises.
- Les services web permettent à des programmes écrits en des langages différents et sur des plateformes différentes de communiquer entre eux par le biais de certaines normes. En d'autres termes, les services web permettent une meilleure interopérabilité entre les logiciels.
- Les services web utilisent des normes et protocoles ouverts.
- Grâce au protocole HTTP, les services web peuvent fonctionner malgré les pare-feu sans pour autant nécessiter des changements sur les critères de filtrage.

- Les protocoles et les formats de données sont offerts, le plus possible, en format texte pour que la compréhension du fonctionnement des échanges soit plus intuitive.
- Grâce aux services web, les coûts sont réduits par l'automatisation interne et externe des processus commerciaux [8].

1.7 Inconvénients des Services Web

La technologie des services web comporte plusieurs inconvénients dont :

- **Problèmes de sécurité :** Il est facile de contourner les mesures de sécurité mises en place par les pare-feu -l'utilisation du protocole HTTP (tel que mentionné ci-haut) n'a pas que des avantages -car les normes de sécurité des services web laissent encore à désirer. CORBA, par exemple, qui est une technologie plus mûre, est plus sécuritaire.
- **Problèmes de performance :** Les services web sont encore relativement faibles par rapport à d'autres approches de l'informatique répartie telles que CORBA ou RMI.
- **Confiance :** Les relations de confiance entre différentes composantes d'un service web sont difficiles à bâtir, puisque parfois ces mêmes composantes ne se connaissent même pas.
- **Syntaxe et sémantique :** On se concentre beaucoup sur comment invoquer des services (syntaxe) et pas assez sur ce que les services web offrent (sémantique).
- **Fiabilité :** Il est difficile de s'assurer de la fiabilité d'un service car on ne peut garantir que ses fournisseurs ainsi que les personnes qui l'invoquent travaillent d'une façon fiable.
- **Disponibilité :** Les services web peuvent bien satisfaire un ou plusieurs besoins du client. Seront-ils pour autant toujours disponibles et utilisables? Ça reste un défi pour les services web [8].

1.8 Quelques Domaines d'Application de Services Web

Les services Web peuvent être utiles dans la plupart des scénarios applicatifs lorsque la communication peut être établie sur un modèle bidirectionnel (requête/réponse).

- L'application des services Web est multiple, autant dans les domaines du B2B, B2C que pour les domaines de gestion de stock, etc.

- B2C (Business to Consumer) : qualifie une application, un site internet destiné au grand public.
- B2B (Business to Business) : qualifie une application, un site internet destiné au commerce professionnel à professionnel [9].

1.9 Les Propriétés d'un Service Web

L'identité d'un service est définie par ses propriétés fonctionnelles et non fonctionnelles. En effet, les services se distinguent par les fonctionnalités qu'ils peuvent fournir. Deux services équivalents, c'est à dire offrant sémantiquement les mêmes fonctionnalités (exemple un service "Moisture " et un service "Humidité " qui offrent des fonctions de mesure d'humidité), peuvent être différenciés par leurs propriétés non fonctionnelles. Par exemple le service " Humidité " a un temps d'exécution inférieur au service " Moisture " alors que le service " Moisture " est caractérisé par un taux de consommation d'énergie inférieur au premier. Les propriétés non fonctionnelles sont la base de définition de la qualité de service. [10].

1.9.1 Les Propriétés Fonctionnelles : Les propriétés fonctionnelles d'un service désignent les fonctionnalités que ce service peut fournir. Les propriétés fonctionnelles sont décrites dans la description de service en termes d'opérations, et revêtent le fonctionnement du service, comme par exemple un service de localisation.

1.9.2 Les Propriétés non-Fonctionnelles : Les propriétés non fonctionnelles de service appelées aussi : qualités de service définissent les capacités de service à fonctionner dans de bonnes conditions en termes de disponibilité, performance, coût d'invocation, fiabilité, etc.

1.10 Qualité de Service (QoS)

La QoS est un ensemble de propriétés et caractéristiques d'une entité ou d'un service qui lui confèrent l'aptitude à satisfaire des besoins déclarés ou implicites. Ces besoins peuvent être liés à des paramètres tels que l'accessibilité, la disponibilité, le temps de réponse, le coût, la fiabilité, etc. Ces paramètres peuvent être alors considérés comme un critère de choix lorsqu'on a affaire à sélectionner parmi plusieurs services web découverts ceux qui respectent les contraintes de temps imposées.[11].

Dans ce qui suit, nous présentons l'ensemble des attributs de QoS génériques. Les attributs les plus communs sont décrits par les paramètres liés à la performance.

- **Le Débit** : Le nombre de requêtes servies par unité de temps.
- **Le Temps de Réponse** : Le temps séparant l'émission de la requête et la réception des résultats.
- **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'entrées invalides, incomplets ou conflictuels.
- **La Disponibilité** : La probabilité d'accessibilité d'un service.
- **La Réputation** : C'est une mesure de la crédibilité du service. Elle dépend principalement des expériences d'utilisateurs finaux.
- **La sécurité** : C'est un regroupement d'un ensemble de qualités à savoir : la confidentialité, le cryptage des messages et le contrôle d'accès.
- **Coût** : également appelé financière ou le cours, concerne le coût et les frais liés à un service. Le coût de la demande et l'utilisation de chaque service est le prix de service Web. Le prix des services est affecté par la valeur de la fonctionnalité. Fournir des fonctions plus complexes augmente le coût du service.

1.11 Conclusion

Dans ce chapitre on a défini le concept de "Service Web", leurs technologies et leur fonctionnement et ainsi l'architecture SOA. Ensuite on a présenté les inconvénients et les avantages des services web et leurs domaines d'application. Enfin on a vu l'aspect non fonctionnel des services Web en présentant quelques paramètres de QoS .

Dans le chapitre suivant nous aborderons le concept de l'approche de la sélection des services Web composites et les méthodes utilisées.

Chapitre 2

La Sélection de Service Web à base de QOS

2.1 Introduction

Avec la croissance explosive du nombre des services publiés sur internet, il est difficile de sélectionner des services web satisfaisants parmi les services web candidats qui offrent des fonctionnalités similaires. La qualité de service (QoS) est considérée comme le critère non fonctionnel plus important pour la sélection du service. Dans ce chapitre, nous présentons l'état de l'art qui introduit un problème spécifique émergent qui est celui de sélectionner le meilleur service en respectant les contraintes de QoS. Pour cela, nous décrivons les différentes stratégies de sélection et nous citons quelques approches utilisées pour résoudre ce problème. Ainsi nous citant quelques algorithmes utilisés par ces approches, ensuite nous présentons les travaux connexes.

2.2 Technique de Sélection de Service Web

Plusieurs travaux ont été proposés pour résoudre le problème *de sélection des services web*. Dans la littérature, on trouve plusieurs classifications des approches : selon la fonction à optimiser mono-objectif, multi-objectif ou hybride [12], Selon la stratégie de sélection utilisée soit locale ou globale [13].

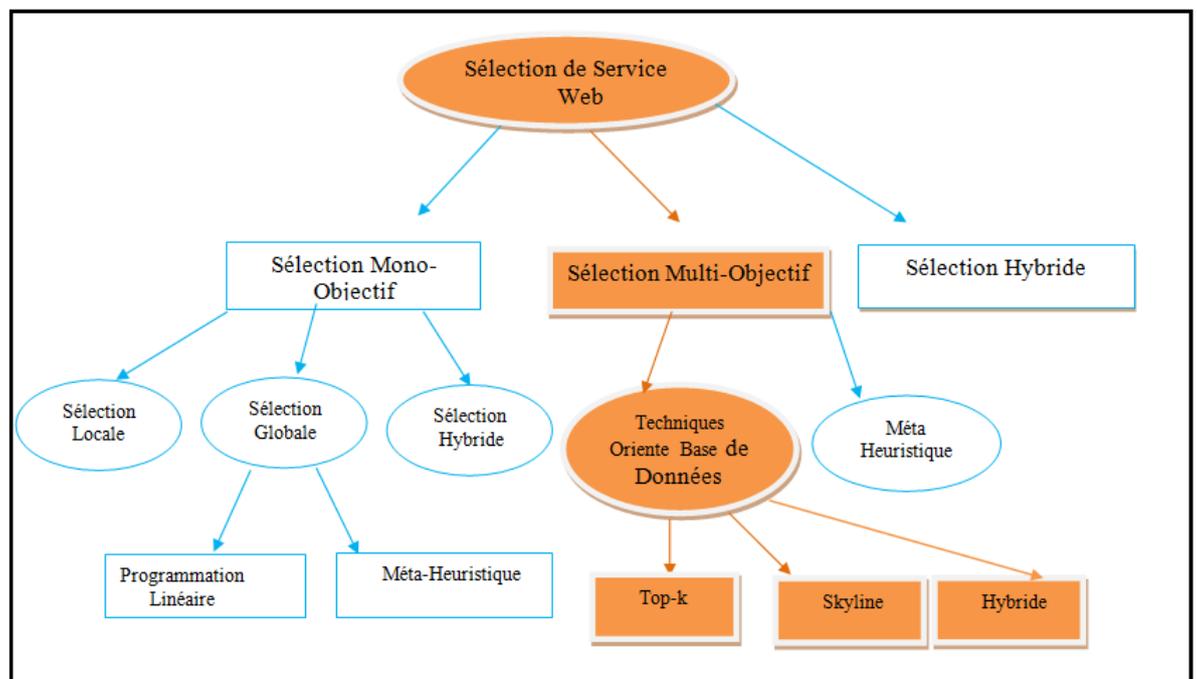


Figure 2.1 : Approches de sélection de services web à base QoS.

On distingue 04 grandes classes selon [14], [15] des approches de sélection des services web :

1. Les approches Exactes.
2. Les approches Heuristiques (approximatives).
3. Les approches Méta-heuristiques (approximatives).
4. Les approches Techniques de bases de données.

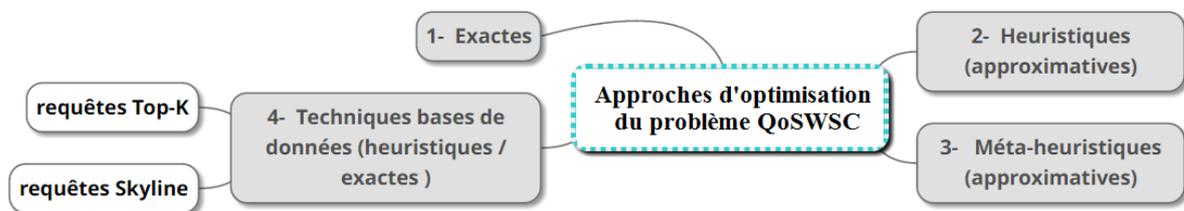


Figure 2.2 : Approches d'Optimisation du Problème QOSWSC

2.2.1 Approches Exactes

Toujours dans l'obtention de la solution optimale d'un problème traité, les méthodes exactes assurent cet objectif avec un parcours complet sur l'ensemble de l'espace de recherche afin de retirer toutes les solutions qui peuvent être bien meilleur que la solution optimale courante.

2.2.2 Approche Heuristique (Approximative)

Une méthode heuristique est une méthode de résolution de problème qui ne s'appuie pas sur une analyse détaillée ou exhaustive du problème. Elle consiste à fonctionner par approches successives en s'appuyant, par exemple, sur des similitudes avec des problèmes déjà traités afin d'éliminer progressivement les alternatives et ne conserver qu'une série limitée de solutions pour tendre vers celle qui est optimale. Les heuristiques prennent beaucoup plus moins de temps pour trouver la solution optimale par rapport au Méthode exacte.

2.2.3 Approche Méta-Heuristique

Les méta-heuristiques sont généralement des algorithmes stochastiques itératifs, qui progressent vers un optimum global, c'est-à-dire l'extremum global d'une fonction, par échantillonnage d'une fonction objectif. Elles se comportent comme des algorithmes de recherche, tentant d'apprendre les caractéristiques d'un problème afin d'en trouver une approximation de la meilleure solution (d'une manière proche des algorithmes d'approximation).

2.2.4 Les Méta-Heuristiques à Base de Solution Unique

Cette catégorie initialise la recherche avec une solution unique puis elle l'améliore au cours d'une série d'itération en se basent sur la notion du voisinage. La solution initiale subit des modifications selon son voisinage afin d'améliorer progressivement sa qualité. Nous trouvons dans ces classe deux familles d'algorithmes de recherche local : Pour une optimisation locale : recherche locale simple (la descente). Pour une optimisation globale : Recuit simulé.

2.2.5 Méta-Heuristiques à Base de Population de Solution

Cette catégorie initialise la recherche avec ensemble de solutions afin d'obtenir la meilleure (solution optimal) qui est la solution du problème traite. L'utilisation d'un ensemble de solutions augmente la possibilité d'arriver à une solution de bonne qualité. Les catégories de cette classe : les algorithmes évolutionnaires, les algorithmes génétiques, et les algorithmes à base d'intelligence par essaims : Colonies de fourmis, Fruit Fly.

2.2.6 Approches Basées sur les Techniques de Bases de Données

Cette catégorie traite la sélection comme un problème multi-objectif qu'il soit traité en local ou en global (sans le réduire à une fonction mono-objectif). Elle repose surtout sur la notion de Pareto dominance et sur le principe des requêtes Skyline ou Top-k. Elle utilise des algorithmes comme celui de Divide and conquer ou autres. Dans ce qui suit on illustre quelques notions de base pour cette approche.

2.2.6.1 La Requête Skyline

Les requêtes skyline repose sur le paradigme très populaire et puissant pour extraire des objets d'un ensemble de données multidimensionnel. Elles s'appuient sur le principe de dominance de Pareto qui peut être défini comme suit :

Définition 1: La Relation de Pareto Dominance.

Étant donné un espace $D = d_1, \dots, d_n$ à n dimensions et un ensemble de points E , on dit qu'un point $p \in E$ domine un autre point $q \in E$ au sens de Pareto selon D , noté par $p \succ q$, si p est meilleur ou égal à q dans toutes les dimensions et strictement meilleur dans au moins une dimension .

$$p \succ q \Leftrightarrow \forall d_i \in D, p[i] \geq q[i] \wedge \exists j \in D, p[j] > q[j]$$

Trois cas de comparaisons entre deux points p et q se présentent selon la notion de Pareto :

- le point p **domine** le point q ou bien ;
- le point p **est dominé par** q ou bien ;
- le point p et q sont **incomparables** : le point p ne domine pas q et le point q ne domine pas p .

Définition 2: Le Skyline :

L'ensemble des points Skyline de E selon D est l'ensemble des points de E qui ne sont dominés (selon D) par aucun autre point de E :

$$Skyline(D, E) = \{u \in E / \nexists p \in E; p \prec u\}$$

Exemple : Considérons une base de données contenant des informations sur des hôtels comme indiqué dans **Erreur ! Source du renvoi introuvable.** (où la dimension $d = 2$).

Hotel	Prix(€)	Distance (Km)
H1	4.0	150
H2	3.0	110
H3	2.5	240

Tableau 2.1: Exemple l'Algorithme de Skyline.

Considérons une personne qui cherche un hôtel aussi proche que possible de la plage et ayant un prix faible. On peut vérifier que le skyline résultant S contient les hôtels $H2$ et $H3$, car $H1$ est dominé par $H2$.

2.2.6.2 La Requête Top-K

L'idée principale est de permet de retourner à l'utilisateur les k point les plus pertinents. Ainsi, le Top- k dominating permet de retourner les Top- k points selon leurs forces de dominance. Ces k points sont déterminés grâce à leurs puissances de domination. La puissance de domination d'un point p dans un ensemble S est le nombre de points que p domine dans S .

$$dom(p) = |\{q \in E : q < p\}|$$

Top- k dominating permet de retourner l'ensemble de k points qui dominent le plus large nombre de points de l'ensemble total des points de l'espace de recherche. Dans ce qui suit nous présentons en détail les travaux de chaque catégorie en s'appuyant sur plusieurs papiers de la littérature du problème de sélection des services web.

2.2.6.3 La Méthode Hybride :

L'idée principale est de permet de combiner les deux méthodes précédemment citer le Top- k et Skyline afin de retourner à l'utilisateur les k point les plus pertinents en choisissons d'après les points incomparables ou Skyline. Ainsi, c a d on diminue l'espace de recherche. Dans ce qui suit nous présentons en détail les travaux de chaque catégorie en s'appuyant sur plusieurs papiers de la littérature du problème de sélection des services web.

- **La Fuzzification :**

C'est un processus qui consiste à transformer une grandeur numérique en un sous-ensemble flou. Qualifier une valeur numérique avec un terme linguistique.

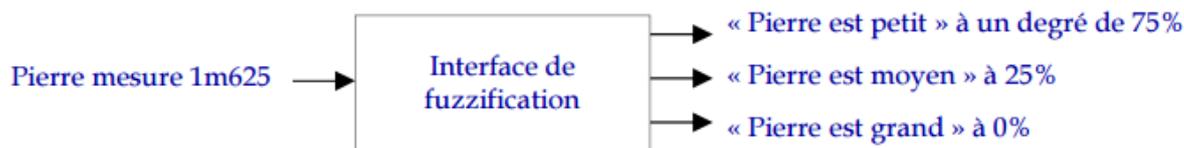


Figure 2.3: Exemple de Fuzzification

2.3 Travaux Connexes

2.3.1 Approches Exactes

- [17] et [18] présentent une méthode en considérant de multiples critères QoS en utilisant l'algorithme brunch and bound et traitent différents types de workflows.
- [19] proposent une méthode pour trouver la solution pour une composition de services en utilisant un graphe de dépendance et 0-1 LIP. Liu et al. [20] proposent une méthode basée sur l'enveloppe convexe (convexhull) et appliquent la méthode de décision multi-critères, multiple criteria decisionmaking (MCDM) pour fusionner les ressources multiples pour les contraintes globales et locales.
- [21] utilisent l'algorithme A* [22] qui est connu pour réaliser de meilleures performances que les algorithmes de recherche de chemins dans les graphes en utilisant des heuristiques pour guider la recherche.

2.3.2 Approche Heuristique (Approximative)

- [23] proposent un algorithme nommé qssac qui peut trouver une solution proche de l'optimale (mais les auteurs n'ont pas mentionné les résultats d'expérimentations portant sur le degré d'optimalité). Les auteurs proposent un deuxième algorithme dans le but de réduire le temps d'exécution, nommé optics. Ce dernier regroupe les services similaires en termes de QoS en M classes (et leurs représentants) pour chaque tâche, ensuite il énumère toutes les compositions possibles pour 02 ou K tâches du document BPEL, et les trie à l'aide d'une fonction objectif.
- [24], [25] utilisent l'algorithme hill-climbing (Méthode de Descente) pour réduire la complexité de temps de calcul et comparent leur méthode avec celle utilisant LIP.

- [26] et [27], ils utilisent les algorithmes d'apprentissage par renforcement pour trouver l'ensemble de solutions Pareto optimales qui satisfait les facteurs QoS multiples et les exigences de l'utilisateur.

2.3.3 Méta-Heuristiques à Base de Population de Solution

- [28] présentent l'algorithme de sélection clonale pour la résolution du problème de sélection. Ils considèrent une fonction mono objective qui manipule 5 paramètres QoS. Les auteurs montrent que la sélection clonale s'avère plus intéressante que l'algorithme génétique.
- [29] proposent une première adaptation de la sélection négative au problème de sélection et présente l'algorithme 'NSA'. Ils proposent ensuite une amélioration de l'algorithme initiale et propos NSA++.
- [30] ont présenté un algorithme d'optimisation d'essaim de particule quantique 'hqps0' pour résoudre le problème de sélection. Dans [31], les auteurs proposent les réseaux de pétri coloré (CPN) et une version discrète de PSO. [32] adoptent la version discrète de PSO qui est basée sur les systèmes immunitaires artificiels. Une méthode hybride combinant PSO avec la méta-heuristique recuit simulé est proposée dans [33] pour sélectionner la composition optimale basée sur les attributs QoS. La position d'une particule dans PSO représente les solutions potentielles, tandis que la vitesse est utilisée pour modifier une solution. Pour éviter le problème de stagnation prématurée dans une solution (localement optimale), une stratégie de recuit simulé est adoptée pour créer de nouvelles solutions de composition en perturbant la solution initiale.
- Dans [34], les auteurs ont proposé un algorithme génétique dans lequel ils ont présenté le problème sous forme d'un arbre. Et [35] ont adopté un algorithme génétique amélioré utilisant l'optimisation par colonies de fourmi pour sélectionner les anticorps de la population initiale afin d'améliorer la convergence.

2.3.4 Approches basées sur les Techniques de Bases de Données

- Les auteurs dans [36] 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 représentant 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.

- Dans [37] ils proposent une variante de Skyline qui repose sur une extension de Pareto dominance, ils proposent une fuzzification de Pareto dominance appelée ' α -dominance'. Cette relation permet la comparaison des services de manière plus efficace que la notion de Pareto dominance ; car elle permet de comparer les services qui sont incomparables au sens de Pareto. ' α -dominance' permet d'étendre la dominance de Pareto avec un degré α . Cette notion sera ensuite utilisée pour déterminer les services qui ne sont pas dominés (avec un degré α) par les autres. Les auteurs utilisent l'algorithme brunch and bound Skyline pour calculer le ' α - dominance Skyline'. Par contre, les auteurs ne traitent pas l'aspect de la composition ; l'approche qu'ils proposent permet de retourner les services Skyline d'une seule classe. L'intérêt de cette approche est le fait de pouvoir contrôler la taille de services Skyline (étendre ou restreindre le nombre de services) en faisant varier la valeur de α , ceci permet de prendre en considération certains services qui étaient éliminés par la notion de Skyline.
- [38] présentent une approche de sélection des services proche de celle proposée par [36] dans le sens où, cette approche utilise elle aussi le Skyline dans la sélection locale associée à un algorithme de classification, pour réduire l'ensemble des services dans chaque classe. L'approche proposée par [38] est définie en trois étapes : La première étape, extrait les services Skyline de chaque classe en utilisant

l'algorithme SFS 'Sort First Skyline' [39]. La deuxième étape, permet le clustering hiérarchique ascendant des services Skyline, de chaque classe abstraite, en utilisant l'algorithme de Ward. L'avantage de cette méthode par rapport à celle utilisée dans [36], est qu'elle donne un résultat plus stable (la méthode K-means est toujours dépendante de la phase d'initialisation aléatoire, et de ce fait, elle n'est pas stable). Un service représentatif est choisi de chaque cluster pour participer à l'optimisation globale. La troisième étape est la phase globale. Elle repose sur une fonction objective qui permet de vérifier si la composition des services représentatifs de chaque classe respecte les contraintes globales. La composition qui maximise la fonction objective et vérifie les contraintes est choisie.

Notre travail est consacré à cette dernière classe qui repose sur la sélection des services web composites basé sur la technique orientée base de données. Tous les travaux que nous venons de citer reposent sur la notion de Pareto dominance au sens strict qui pose un problème d'incomparabilité, et parfois retourne des services avec un mauvais compromis entre les valeurs QoS. Notre proposition repose sur une extension de la relation de dominance Pareto en utilisant la logique floue.

2.4 Conclusion

Dans ce chapitre on a présenté un état de l'art des différentes approches de sélection de services Web à base QoS. Ainsi les concepts que nous allons examiner dans notre approche ou la méthode proposée. Ensuite, on a présenté quelques travaux connexes.

Dans le prochain chapitre nous allons présenter en détail la méthode de sélection des services web composites selon la Dominance Floue.

Chapitre 3
La Conception
du Système

3.1.Introduction

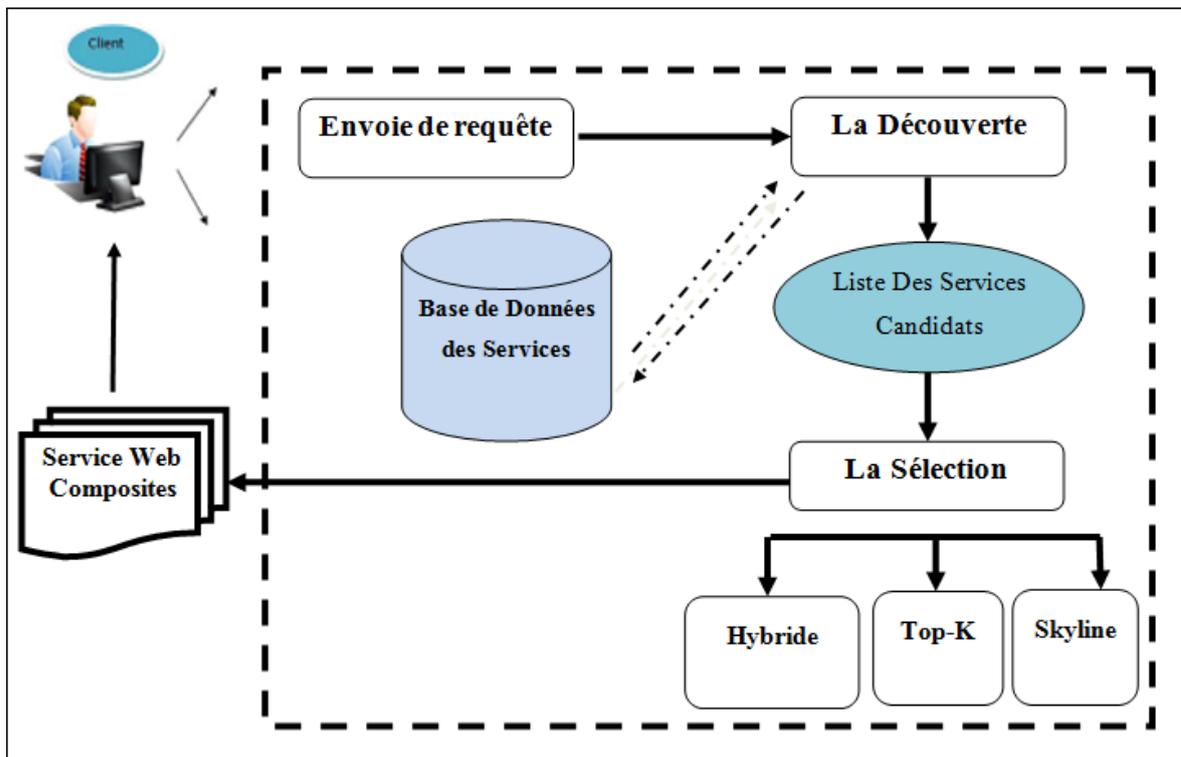
Il y a eu ces dernières années, un grand flux de services web déployés sur le Web. Face à cette croissance exponentielle en nombre et en fonctionnalités des services sur Internet, l'utilisateur est face à la difficulté de choisir manuellement entre un nombre important de services retournés par la requête. Il est devenu essentiel de proposer un système qui permet au client de choisir les meilleurs services qu'il désire sans intervention manuelle.

Nous nous intéressons dans ce mémoire plus particulièrement à la phase de sélection des services qui permet de fournir une composition de services Web. La sélection est une étape cruciale de laquelle dépend le résultat de la composition. Cette sélection n'est pas évidente, car il s'agit de choisir des services Web parmi un nombre important d'alternatives de services proposés par la phase de découverte.

Dans ce chapitre nous présentant notre contribution dans la sélection des services Web composite sur la base des besoins non fonctionnels (QoS). Tout d'abord, nous allons décrire les différentes étapes de la conception de notre système ou de processus général utilisé pour la sélection des services web composites. Ensuite, nous présenterons les différentes méthodes utilisées dans notre travail : Top-k, Skyline, et Top-k Skyline (Hybride). Ces trois méthodes sont appliquées en utilisant la dominance floue.

3.2 Architecture Générale du Système

La **Figure 3.1** présente le processus proposé pour la sélection des services web composites à partir des plusieurs services web.



La Figure 3.1: Architecture Générale du Système.

Le client envoie une requête, puis une phase de découverte est faite pour analyser la demande et de découvrir les services aptes à réaliser cette requête en faisant une recherche dans la base des services qui ont une relation avec la demande du client et les retournés comme une liste des services candidats. La Base de services se compose de différentes classes ou chaque classe contient des services qui offrent les mêmes fonctionnalités avec des valeurs QoS différentes.

La Sélection consiste à choisir un service composite composé à partir des services candidats. Chaque service du service composite possède les QoS les plus convenable et satisfiable aux exigences de l'utilisateur. La sélection faite par trois différentes méthodes : Skyline, Top-K, et Top-K-Skyline (Hybride) en utilisant la dominance floue. Ces méthodes vont être étudiées en détail dans les prochaines sections de ce chapitre. Afin de mieux argumenter et expliquer nos motivations, nous présentons dans la section qui suit un exemple illustratif.

3.3. Exemple Illustratif

On considère la classe *reserverhotel* contenant plusieurs services qui fournissent des tâches de réservations d'hôtels par exemple : *HotelerumBooking Engine*, *Campass*, etc.

mais avec différents QoS. Le **Tableau 3.1** donne une description sur les services utilisés dans cet exemple.

Service	Nom	Fournisseur	q1(ms)	q2(hits/sec)	q3(%)	q4(%)
1	HotelerumBooking Engine	Hotelerum	110.38	45.3	65.5	26.45
2	ComfyHotelReservation	OrgBuisnessSoftware	509.2	28.3	41.3	78.3
3	ResAvenue	Avenue	481	41.11	49.8	98.4
4	HotelReservation System	CMSplaza	1024.1	18.6	32.6	32.1
5	Campass	OnPeak	363.4	60.7	19.9	27.5
6	Booking Engine	ReservHOTEL	563.45	54.9	21.78	11.5
7	HotelReservation System	StivaSoft	771.34	38.7	30.2	34.73
8	Weltraum PMS	Weltraum Software	192.5	60	72.2	100
9	RoomScope	RoomScope	300.67	43.33	46.69	42.1
10	Trawex Cloud Suite	TrawexTechnologie	204.92	71.4	53.2	60.3

Tableau 3.1 :Les services Web de la classe reserverhotel.

Chaque service Web possède un identificateur, un nom, le nom du fournisseur, et ses caractéristiques QoS. On a choisi quatre paramètres QoS, q_1 , q_2 , q_3 et q_4 , qui représentent respectivement le Temps de Réponse, Débit (ie, Nombre total d’invocations / période de temps) ,Fiabilité (Nombre de messages d’erreur / messages au total) et le Cout.

3.4.La Sélection de Services Composites

C’est la tâche primordiale de notre travail qui consiste à trouver un service Web composite qui répond au besoin de l’utilisateur. Cela consiste à choisir parmi les services Web de chaque classe ce qui répond aux exigences de l’utilisateur pour former le service composite ; c.-à-d. de déterminer l’ensemble des top-services qui forme le service composite le plus adéquat à la requête fournit selon les QoS. **La Figure 3.1** illustre le processus de sélection de top-services i.e. le service composite.

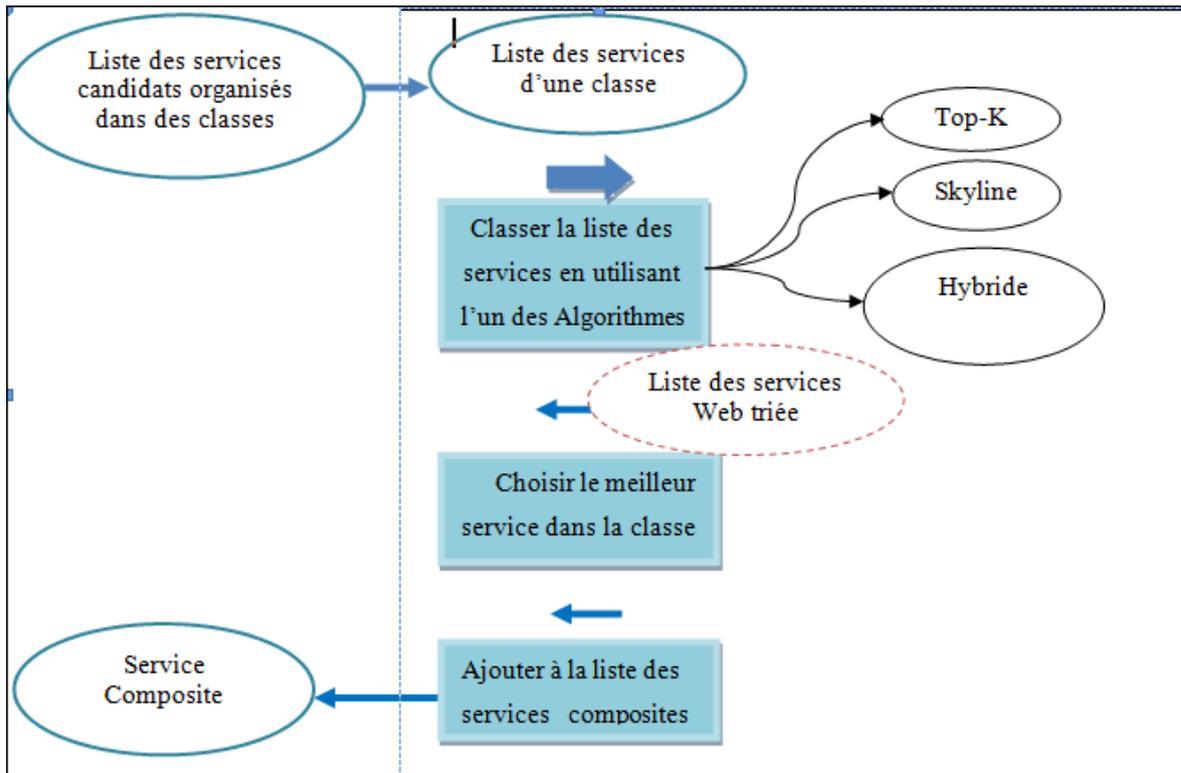


Figure 3.2 : Sélection des Services Composites.

Les services web de chaque classe sont ordonnés et le meilleur service est choisi à l'aide d'un processus de classement. Ce classement est garanti à l'aide d'une des méthodes orientées base de données pour la sélection des services web que nous utilisons : Top-K, Skyline, Top-K -Skyline (Hybride). Les résultats d'appliquer l'une de ces méthodes sur chaque classe est un service appelé le top service, l'ensemble des top-services obtenues forme le service composite qu'on cherche. Comme nous avons vu dans le chapitre précédent que la majorité des travaux repose sur la relation de dominance de Pareto au sens strict qui, en dépit du problème d'incomparabilité, retourne des services avec un mauvais compromis entre les valeurs QoS. Notre proposition repose sur une extension de la relation de dominance qui est la dominance floue. Dans les sections suivantes nous allons présenter les trois méthodes de sélection basée sur la dominance floue.

3.4.1 La Sélection des Top-k Services basée sur la Dominance Floue

Dans cette section, nous allons présenter la méthode de sélection des Top-k services en utilisant la dominance floue. Comme en est en train de travailler avec la logique floue,

on doit d'abord mettre les valeurs QoS dans l'intervalle [0,1]. Nous commençons par la normalisation des valeurs QoS. Ensuite, on applique la méthode Top-k basée sur la dominance floue.

a) Normalisation des Paramètres QoS

Soit S un ensemble de services qui offrent la même fonctionnalité $S = S_1, \dots, S_n$. Supposons que nous avons r valeurs quantitatives QoS pour un service S_i . Nous utilisons le vecteur $Q(S_i) = \{Nq_1(S_i), \dots, Nq_r(S_i)\}$ pour représenter les attributs QoS d'un service S_i où la fonction $Nq_k(S_i)$ représente le k ième attribut de qualité normalisée de S_i . On convertit les attributs négatifs (temps, coût) en des attributs positifs en multipliant leurs valeurs par le nombre (-1) de telle sorte que la valeur la plus grande représente la meilleure qualité. On normalise les différentes valeurs des attributs QoS dans l'intervalle $[0, 1]$, comme suit :

$$Nq_k(S_i) = \frac{q_k(S_i) - Q_{\min}(q_k)}{Q_{\max}(q_k) - Q_{\min}(q_k)}$$

Ou :

$Nq_k(S_i)$: est la valeur normalisée de l'attribut q_k du service web S_i .

$Q_{\min}(q_k)$, (resp. $Q_{\max}(q_k)$) : représente le minimum et le maximum des valeurs des attributs q_k , ($Q_{\max}(q_k) \neq Q_{\min}(q_k)$).

Web service	Nq1	Nq2	Nq3	Nq4
1	1.0	0.51	0.77	0.0
2	0.56	0.18	0.41	0.42
3	0.59	0.43	0.57	0.24
4	0.0	0.0	0.24	0.82
5	0.72	0.8	0.0	0.86
6	0.5	0.69	0.04	1.0
7	0.28	0.38	0.2	0.8
8	0.91	0.78	1.0	0.23
9	0.79	0.47	0.51	0.73
10	0.9	1.0	0.63	0.58

Tableau 3.2: Les Valeurs des QoS des Services Web après Normalisation.

Le **Tableau 3.2** montre les valeurs des attributs QoS des services web de l'exemple (voir Tableau 3.1) après la normalisation.

Dans les sections suivantes, nous présentons la fuzzification de la relation de Pareto dominance, et nous montrons son application à travers notre exemple. Nous présentons aussi l'algorithme de classement des services.

b) Fuzzification de la Relation de Pareto Dominance

Les services web appartenant à la même classe, mais sont différents dans leurs attributs QoS. Comme nous l'avons mentionné avant (**voir section 2.3.4**), les services peuvent être comparés par la relation de Pareto dominance sur leurs vecteurs QoS. Mais cette mesure a montré des défauts. Car on peut pas comparer les points incomparables, c.à.d. qui sont non Pareto dominé, ainsi peut éliminer des points pertinents qui peuvent avoir un bon compromis entre leurs QOS.

Reprenons notre exemple (**voir Tableau 3.1**) et considérons S_4 et S_5 . Selon Pareto dominance, ni S_4 domine S_5 , ni S_5 domine S_4 , les deux services sont alors incomparables avec cette mesure. Pour cette raison, il est nécessaire de fuzzifier la relation de Pareto dominance c.à.d. donner un degré à la dominance entre chaque deux services (comparaison numérique) au lieu d'une comparaison booléenne (dominé par ou n'est pas dominé par) au sens strict de Pareto. Cette comparaison exprime l'extension par laquelle un service est (plus ou moins) dominé par un autre.

Afin de calculer le degré de dominance floue entre deux services, il est nécessaire d'abord de distinguer entre les mesures retournées par les deux relations non-symétriques '*domine*' et '*est dominé par*'. Le score retourné par la relation '*domine*' est appelé *score dominant floue* le score retourné par la relation '*est dominé par*' est appelé *score dominé floue*. Dans notre travail, nous utilisons la relation '*est dominé par*' car elle été prouvé de retourner des meilleurs résultats que l'autre mesure [40].

Nous définissons dans ce qui suit la fuzzification de la relation.

Si nous prenons deux services S_i et S_j , le '*score dominant flou*' exprime le degré avec lequel S_i domine S_j , tandis que le '*score dominé flou*' exprime le degré avec lequel S_i est dominé par S_j .

- **Le Score Dominé Flou (Fuzzy-Dominated Score) :** Supposons que S représente un ensemble de services ayant la même fonctionnalité, S_i et $S_j \in S$. $Q = \{q_1, \dots, q_n\}$ représente un vecteur de n paramètres de QoS .

On définit, dans un premier temps, une fonction de comparaison monotone $\mu_{\varepsilon, \lambda}$ pour exprimer le degré avec lequel u est dominé par v , où u représente $q_k(s_i)$ et v représente $q_k(s_j)$ comme suit :

$$\mu_{\varepsilon, \lambda}(u, v) = \begin{cases} 0 & \text{if } (u - v) \geq \varepsilon \\ |u - v - \varepsilon| / |\lambda + \varepsilon| & \text{if } \lambda + \varepsilon \leq (u - v) < \varepsilon \\ 1 & \text{if } (u - v) < \lambda + \varepsilon \end{cases}$$

Où $\varepsilon, \lambda \in [-1, 0]$, $\varepsilon + \lambda \geq -1$ et $(\lambda + \varepsilon) \neq 0$

Nous définissons, ensuite, 'the **Fuzzy-Dominated score**' $FDed(S_i, S_j)$ pour exprimer le degré avec lequel S_i est dominé par S_j comme suit :

$$FDed(s_i, s_j) = \frac{1}{n} \sum_{k=1}^n \mu_{\varepsilon, \lambda}(q_k(s_i), q_k(s_j))$$

Reconsidérons maintenant notre exemple et comparons les Services Web S_4 et S_5 en utilisant $FDed()$, avec $\varepsilon = -0.1$ et $\lambda = -0.2$. Nous avons $FDed(S_4, S_5) = 0.5$ (c.à.d. S_4 est dominé par S_5 avec le degré 0.5) et $FDed(S_5, S_4) = 0.117$, ceci signifie que S_5 est meilleur que S_4 car S_4 est dominé avec un degré de 0.5 par S_5 . Bien que S_5 est dominé avec un degré de 0.117 par S_4 . En effet, ceci est plus expressif que S_4 et S_5 sont incomparables par Pareto dominance. Dans ce qui suit, nous utilisons $FDed()$ pour classer les services Web.

c) Averaged-Fuzzy-Dominated-Score

Pour pouvoir classer un service web S_i dans sa classe S , nous devons d'abord faire, des paires de comparaisons avec les autres services de la même classe et associer un score moyen par :

$$AFDedS(S_i) = \frac{1}{|S| - 1} \sum_{j=1, i \neq j}^n FDed(S_i, S_j)$$

Notons que S contient au moins 02 services.

d) Algorithme FTop-KS de Sélection services basé sur la Dominance Floue

On ordonne les services par un ordre croissant des scores $AFDedS(S_i)$ associés aux services. Le service qui a le score minimal est le meilleur service selon ses critères QoS. Le processus global de sélection des k-meilleurs services est résumé dans l'algorithme 3.1 *FTop-KS*

Entrée(s): **R** : requête de l'utilisateur,

S : Une liste de n services web.

k: le nombre de services à retourner (Top-k services).

λ, ϵ les paramètres d'ajustement de la fonction de comparaison μ .

Sortie(s): **CL** : // les k meilleurs services (Top-K)

Début

Top-K $\leftarrow \emptyset$; //ensemble des k meilleurs services

Resultat = \emptyset ;

pour tous s_j dans S faire

Score_i = 0 ;

fin pour ;

pour tous s_i dans S faire

pour tout s_j dans S et $s_i \neq s_j$ faire

Score_{ij} = $FDed(s_i, s_j)$;

Score_i = Score_i + Score_{ij} ;

fin pour

AScore_i = Score_i / (|S| - 1) ;

Resultat = Resultat $\cup \{(s_i, AScore_i)\}$;

fin pour

TrierAsc(Resultat) //trier en ascendant la liste Resultat selon AScore_i

pour $i = 1..k$ faire

Scourant! ExtrairePosition(i, Resultat) ;

insérer(Top-K, Scourant) ;

fin pour

retourner Top-k ;

Fin.

Algorithme 3.1 : 'FTop-KS' Algorithme pour la sélection des K Meilleurs services basée sur la dominance floue.

Le Tableau 3.3 montre le classement des services de l'exemple du Tableau 3.1 après le calcul de score $AFDedS()$ avec $\epsilon = -0.1$ and $\lambda = -0.2$.

Service Web	AFDedS()	Nq1	Nq2	Nq3	Nq4
10	0.105	0.9	1.0	0.63	0.58
9	0.171	0.79	0.47	0.51	0.73
8	0.187	0.91	0.78	1.0	0.23
5	0.235	0.72	0.8	0.0	0.86
1	0.273	1.0	0.51	0.77	0.0
6	0.323	0.5	0.69	0.04	1.0
3	0.405	0.59	0.43	0.57	0.24
7	0.467	0.28	0.38	0.2	0.8
2	0.498	0.56	0.18	0.41	0.42
4	0.61	0.0	0.0	0.24	0.82

Tableau 3.3 : le classement des services selon $AFDedS()$.

Nous pouvons observer que le meilleur service (Top-1) de la classe *reserverhotel* est le service S_{10} . Ce dernier est meilleur que les autres dans q_1 , q_2 et présente de bonnes valeurs dans les autres paramètres de qualités q_3 , q_4 . Nous remarquons que les services qui ont plusieurs qualités égales à '0' sont en bas du classement.

A travers ces résultats, nous pouvons confirmer que $Fed()$ peut donner de très intéressants résultats qui offrent un bon compromis entre les critères QoS à optimiser.

3.4.2 La Sélection des Services Skyline basée sur la Dominance Floue

Parce que La majorité des algorithmes de calcul de Skyline repose sur la notion de dominance selon Pareto au sens strict, ne réponds pas aux besoins de l'utilisateur nous sommes obligés à trouver une autre astuce. On utilise dans ce qui suit un algorithme de calcul de Skyline qui repose sur la notion de dominance floue.

Alors nous utilisons plus précisément, la fonction de fuzzification de Pareto dominance que nous avons définie précédemment et nous définissons un algorithme de calcul de Skyline des services web qui se base sur l'algorithme SFS (Sort First Skyline) [39] pour d'ordonner toutes les données en entrée à l'aide d'une fonction de score monotone, et

permet ainsi d'extraire les points Skyline de manière progressive en diminuant le nombre de comparaisons de dominance entre les points[40].

a) α -Dominated Skyline d'une Classe de Services

Le α -Dominated Skyline d'une classe S de n services $S = S_1, \dots, S_n$ dénoté par α -DSky S , comprend l'ensemble des services appartenant à la classe des services S qui ne sont pas α -dominated par les autres sur l'ensemble de leurs critères de qualités QoS .

$$\alpha - DSkyS(Q, S) = \{S_i \in S \mid \nexists S_j \in S; S_i \prec_{\mu_{\epsilon, \lambda}}^{\alpha} S_j\}$$

Prenons deux services $S_i, S_j \in S$, un paramètre $\alpha \in [0, 1]$ et la fonction $FDed()$ définie dans l'équation 3.1. On dit que le service S_i est α -dominated par le service S_j noté par $\prec_{\mu_{\epsilon, \lambda}}^{\alpha} S_i$ si $FDed(S_i, S_j) > \alpha$.

Si nous reprenons l'exemple précédent du Tableau 3.3. nous avons, dans le contexte $\epsilon = -0.1, \lambda = -0.2$, S_5 est 0,5-dominated par S_4 ($S_5 \prec_{\mu_{-0.1, -0.2}}^{0.5} S_4$), car $FDed(S_4, S_5) = 0.5$. Par contre S_4 n'est pas 0,5-dominated par S_5 car $FDed(S_4, S_5) = 0.017$.

En considérant toujours notre exemple, pour $\epsilon = -0.1, \lambda = -0.2$ et $\alpha = 0.7$, nous avons 0.7 -DSky $S(Q, S) = \{S_{10}, S_8, S_5\}$. Nous nous choisissons le premier service dans la liste comme top service.

b) Algorithme de Calcul des Services Skyline α -DSky S

Algorithme 3.2 présente les étapes de sélection des services Skyline.

Entrée(s):

S : Une liste de **n** service web ;

α : le paramètre de degré de dominance ;

λ, ϵ les paramètres d'ajustement de la fonction de comparaison $\mu_{\epsilon, \lambda}()$;

Sortie(s): α -DSkyS;

Début

α -DSkyS $\leftarrow \emptyset$;

pour tous $s \in S$ **faire**

calculer $f(s)$; ($f()$ est une fonction objectif monotone qui calcule la moyenne des R valeurs normalisées de QoS de s) ;

fin pour

$L \leftarrow \text{TrierDesc}(S)$; //Trier la liste S du plus grand au plus petit selon $f()$ et l'affecter à L ;

tant que $L \neq \emptyset$ **faire**

extraire le service s à la position $L[i]$ et le supprimer de L ;

$s \in \alpha$ -DSkyS tel que s est α -dominated par s' **alors**

négliger s et passer au service suivant ; (i.e. $i=i+1$).

sinon

$s \in \alpha$ -DSkyS tel que s' est α -dominated par s **alors**

remplacer s' par s et passer au suivant ;

sinon

α -DSky = α -DSky $\cup s$.ajouter s à α -DSkySet passer au suivant

;(i=i+1) ;

fin si

fin si

fin tant que

retourner α -SkyS;

Fin.

Algorithme 3.2 : ' α -DSkyS-SFS' : Algorithme de calcul des Skylines basé sur FDed() et SFS.

L'algorithme repose sur les principes de celui de l'algorithme SFS (Sort First Skyline)[39] pour le calcul de Skyline. En utilisant SFS, nous pouvons extraire les services Skyline de manière progressive tout en diminuant le nombre de comparaisons de dominance floue $FDed()$ entre les services grâce à l'utilisation d'une fonction de score monotone utilisée pour l'ordonnement des services en entrée.

3.4.3 La Sélection Hybride des Services basée sur la Dominance Floue

Dans cette section, nous allons présenter la méthode de sélection Hybride des services en utilisant la dominance floue. Cette méthode est une combinaison des deux méthodes présentées précédemment : le top-k en utilisant la dominance floue et le Skyline en utilisant la dominance floue. On extrait les points Skyline c.-à-d. commençons par la recherche des services Skyline parmi tous les services de la classe on utilise l'algorithme ' α -DSkyS-SFS'. Après nous appliquons l'algorithme '**FTop-KS**' afin de retourner le top service de la classe.

Entrée(s): R : requête de l'utilisateur,

S : Une liste de n services web.

k : le nombre de services à retourner (Top-k services).

α : le paramètre de degré de dominance ;

λ, ϵ les paramètres d'ajustement de la fonction de comparaison μ .

Sortie(s): CL : // les k meilleurs services (Top-K)

Début

$Top-K \leftarrow \emptyset$; // ensemble des k meilleurs services

$SkyS = \emptyset$; // ensemble des services skyline

$SkyS = \alpha$ -DSkyS-SFS($S, \alpha, \lambda, \epsilon$);

$Top-K = \text{FTop-KS}(SkyS, \lambda, \epsilon)$;

retourner $Top-k$;

Fin.

Algorithme 3.3 : 'FTop-KSky' Algorithme pour la sélection des Top-K Skyline services basée sur la dominance floue.

3.5 Conclusion

Dans ce chapitre nous avons présenté une méthode de sélection basée sur les requêtes Skyline et Top-k pour la sélection des services composites. Et nous avons présenté l'algorithme '*F*Top-K services' qui permet la sélection des Top-k services web basée sur la dominance floue et l'algorithme ' α -DSkyS-SFS' qui permet de calculer les services Skyline en se basant sur la dominance floue. Ensuite nous avons proposé une méthode hybride composée de Top-K et Skyline ou le dernier « Top-K » utilise les résultats retournés par le première « Skyline ». Afin de sélectionner les meilleurs services web composites on se basant sur des critères de qualités de service (QoS).

Dans le prochain chapitre nous allons présenter l'implémentation de notre application et les résultats de l'expérimentation.

Chapitre 4
Implémentation
du Système
(Etude de Cas)

4.1 Introduction

L'objectif de ce chapitre est la présentation de l'aspect implémentation de notre application, que nous avons entamé par la présentation de l'environnement matériel sur lequel notre système a été développé, les langages de programmation et les outils utilisés. Enfin nous présenterons une description de notre système appuyée par des résultats expérimentaux.

4.2 Environnement de Développement

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

Pour réaliser notre système, nous avons un PC I3 doté de Windows 7 (64bits) qui est décrit avec la figure suivante :

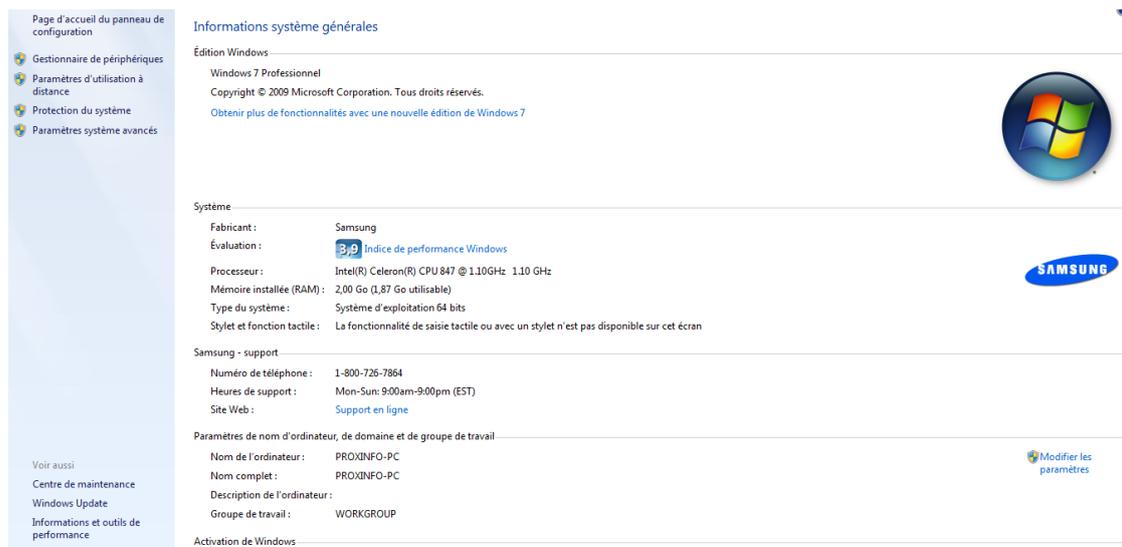


Figure 4.1 : Environnement logiciel utilisé.

4.2.1 Le Langage JAVA

Le langage Java est un langage de programmation et une plate-forme informatique évolué et orienté objet qui est créé par James Gosling et Patrick Naughton, employés de Sun Microsystems, avec le soutien de Bill Joy (cofondateur de Sun Microsystems en 1982), présenté officiellement le 23 mai 1995 au SunWorld. La société Sun a été ensuite rachetée en 2009 par la société Oracle qui détient et maintient désormais Java. Aujourd'hui, Java rassemble derrière lui une large communauté d'acteurs informatiques majeurs tels que HP,

IBM, Oracle, Borland [Java]. Il est rapide, sécurisé et fiable. En outre, beaucoup d'applications et de sites Web ne fonctionnent pas si Java n'est pas installé et leur nombre ne cesse de croître chaque jour. À cause de Sa simplicité, sa robustesse, sa portabilité ainsi que sa performance lui ont permis d'être le choix préféré pour le développement de notre application [40].

Les caractéristiques du langage java sont [41] :

- **Interprété** : Le fonctionnement de Java est assuré JVM (Java Virtual Machine) et JDK qui peuvent être installés dans les différents systèmes d'exploitation. Les instructions JVM sont traduites lors de leur exécution en instructions natives de la machine ; le compilateur Java traduit le code source Java en bytecode (code portable). Par la suite un interpréteur Java spécifique JVM traduit et exécute le bytecode. Il permet l'encapsulation et la génération dynamique d'autres technologies.
- **Portable** : Java fonctionne en mode interprété il est donc un langage portable. Il est indépendant de toute plate-forme : Windows, UNIX, DOS. Ce concept est à la base du slogan de Sun pour Java : WORA (Write Once, Run Anywhere : écrire une fois, exécuter partout). Il n'y a pas de compilation spécifique pour chaque plate-forme. Le code reste indépendant de la machine sur laquelle il s'exécute. Il est possible d'exécuter des programmes Java sur tous les environnements qui possèdent une Java Virtual Machine. Cette indépendance est assurée au niveau du code source grâce à Unicode et au niveau du code byte.
- **Orienté objet** : Comme la plupart des langages récents, Java est orienté objet. Chaque fichier source contient la définition d'une ou plusieurs classes qui sont utilisées les unes avec les autres pour former une application. Java n'est pas complètement objet car il définit des types primitifs (entier, caractère, flottant, booléen,...). Ainsi, Java est un langage orienté objet simple ce qui réduit les risques d'incohérence et son programme n'est pas un ensemble de procédures qui s'appellent les unes les autres mais un ensemble d'objets.
- **Simple** : Le choix de ses auteurs a été d'abandonner des éléments mal compris ou mal exploités des autres langages tels que la notion de pointeurs (pour éviter les incidents en manipulant directement la mémoire), l'héritage multiple et la surcharge des opérateurs.
- **Sécurisé** : La sécurité fait partie intégrante du système d'exécution et du compilateur. Un programme Java planté ne menace pas le système d'exploitation. Java assure une certaine sécurité au système à travers des tests qui vérifient en permanence la

conformité du pseudo-code à certaines règles ; Security Manager : protection des fichiers et accès au réseau.

- Multitâche : Il permet l'utilisation de threads qui sont des unités d'exécution isolées. La JVM utilise plusieurs threads.
- Riche bibliothèque : Java possède une riche bibliothèque de classes comprenant des fonctions diverses telles que les fonctions standards, le système de gestion de fichiers, les fonctions multimédia et plusieurs d'autres fonctionnalités.

4.2.2 Eclipse

Eclipse est un projet, décliné et organisé en un ensemble de sous-projets de développements logiciels, de la fondation Eclipse visant à développer un environnement de production de logiciels libre qui soit extensible, universel et polyvalent, en s'appuyant principalement sur Java.

Son objectif est de produire et fournir des outils pour la réalisation de logiciels, englobant les activités de programmation (notamment environnement de développement intégré et frameworks) mais aussi d'AGL recouvrant modélisation, conception, test, gestion de configuration, reporting... Son EDI, partie intégrante du projet, vise notamment à supporter tout langage de programmation à l'instar de Microsoft Visual Studio

Bien que Eclipse ait d'abord été conçu uniquement pour produire des environnements de développement, les utilisateurs et contributeurs se sont rapidement mis à réutiliser ses briques logicielles pour des applications clientes classiques. Cela a conduit à une extension du périmètre initial d'Eclipse à toute production de logiciel : c'est l'apparition du framework Eclipse RCP en 2004.

Figurant parmi les grandes réussites de l'open source, Eclipse est devenu un standard du marché des logiciels de développement, intégré par de grands éditeurs logiciels et sociétés de services.

Les logiciels commerciaux Lotus Notes 8, IBM Lotus Symphony WebSphere Studio Application Developer sont notamment basés sur Eclipse[40].

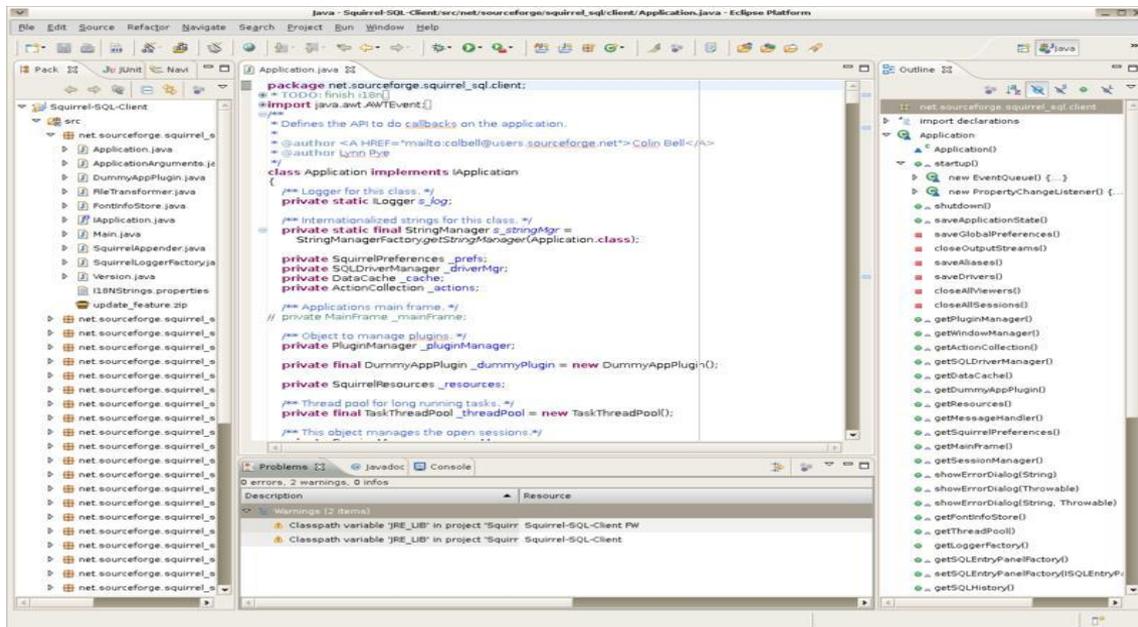


Figure 4.2 : Interface principale de Eclipse.

4.2.3 Gestion de Base de Données (phpMyAdmin) :

phpMyAdmin (anciennement MySQL administrator) est un logiciel de gestion et d'administration de bases de données MySQL. Via une interface graphique intuitive, il permet, entre autres, de créer, modifier ou supprimer des tables, des comptes utilisateurs, et d'effectuer toutes les opérations inhérentes à la gestion d'une base de données. Pour ce faire, il doit être connecté à un serveur MySQL[42].

MySQL est un système de gestion de bases de données relationnelles (SGBDR). Il fait partie des logiciels de gestion de base de données les plus utilisés au monde. MySQL fait référence au Structured Query Language, le langage de requête utilisé.

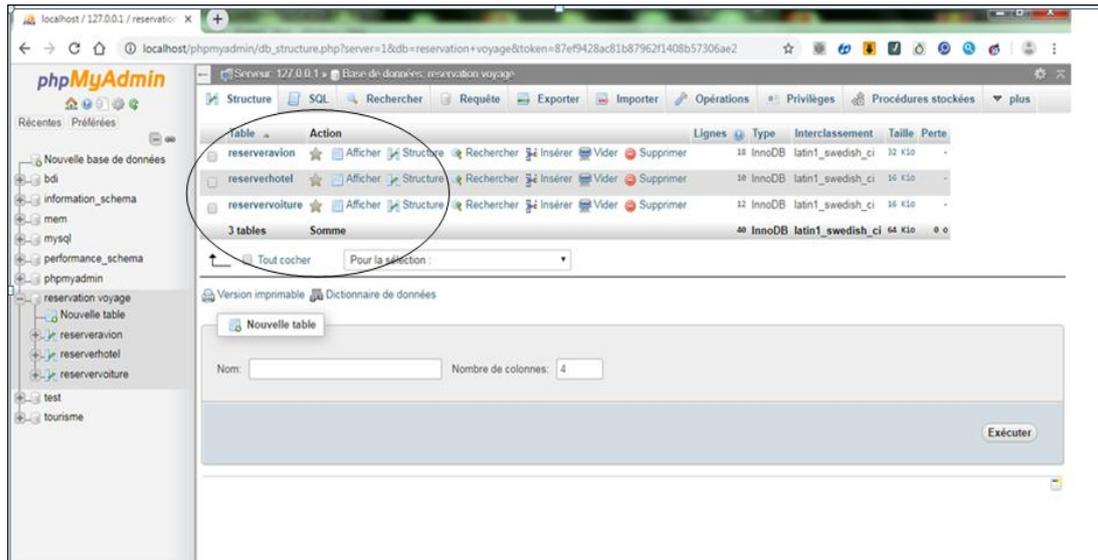


Figure 4.3: Interface Serveur phpMyAdmin.

4.3 Présentation des Interfaces Graphiques

4.3.1 Interface d'Accueil

La qualité de l'interface est l'une des caractéristiques qui attire l'utilisateur . De ce la nous avons essayé de la représenter dans une bonne forme tout en respectant l'aspect de simplicité.

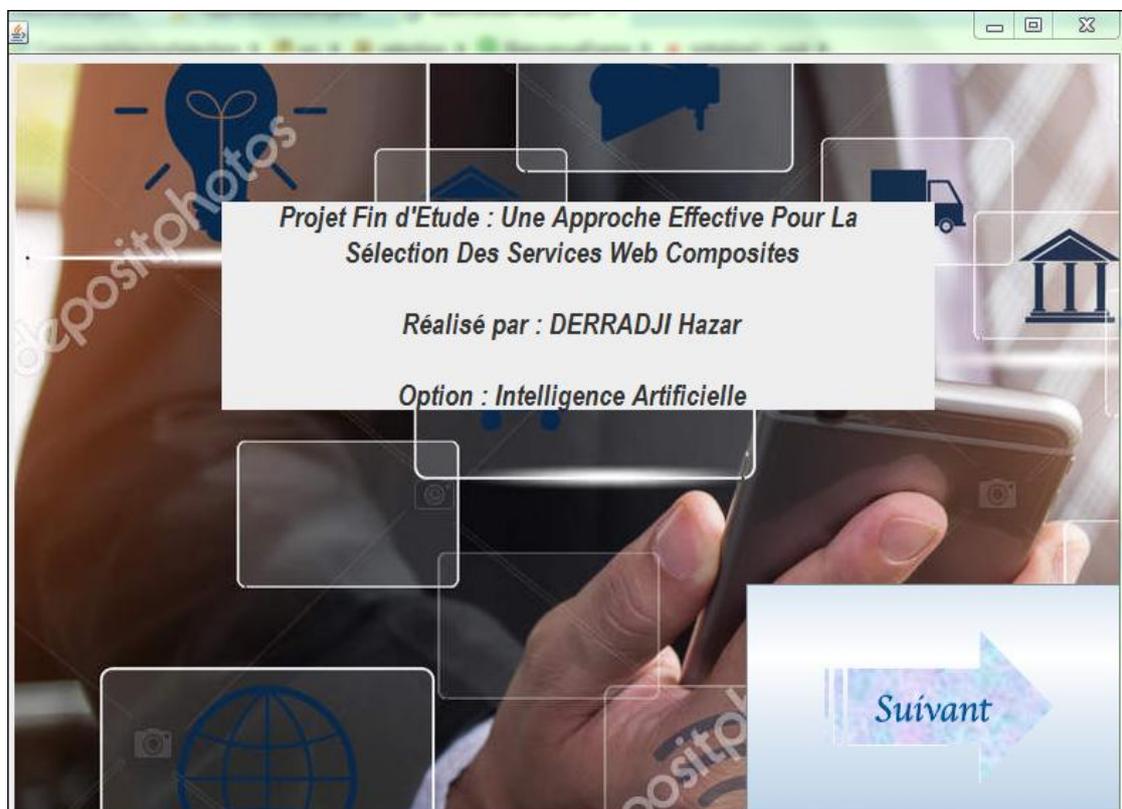


Figure 4.4: Interface d'Accueil.

4.3.2 Interface Principale

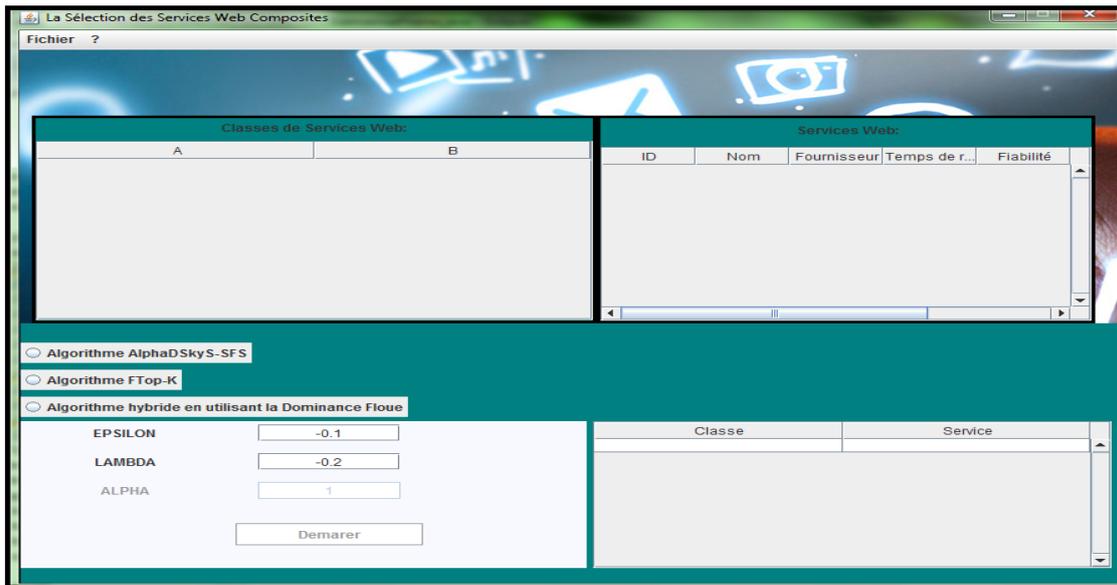


Figure 4.5: Interface Principale.

4.3.3 Chargement de la Base de Données

Pour le chargement des bases de données on sélectionne le menu fichier et sélectionner l'option charger base de données. Enfin, les tables sont disponibles sous la forme suivante, permettant de d'avoir les bases existantes :

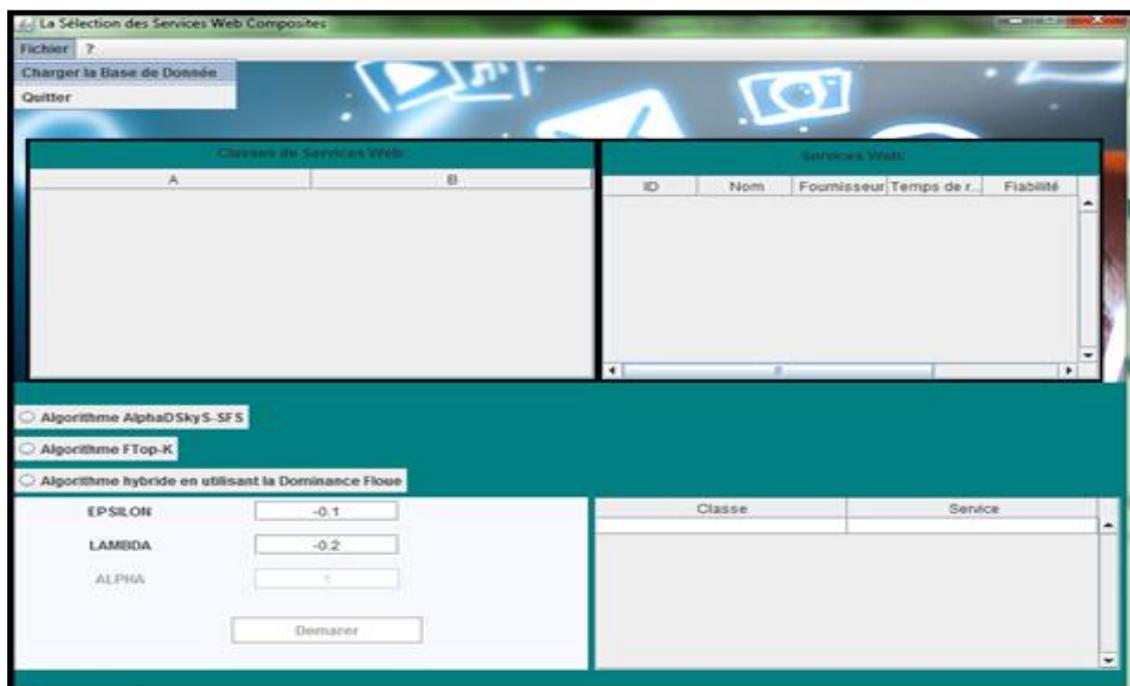


Figure 4.6: Chargement des Bases Données .

Après on saisit la requête comme si dessous :

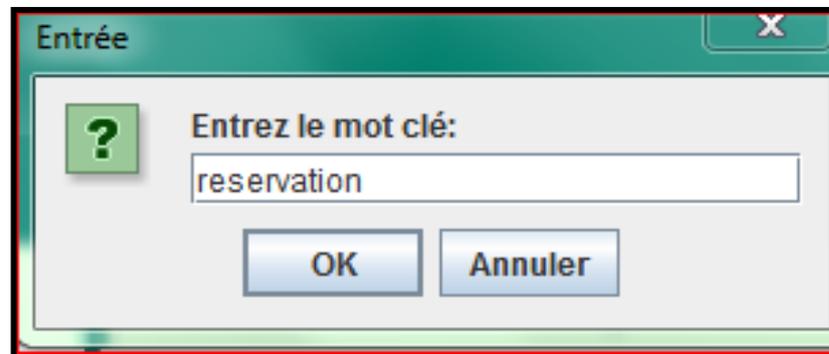


Figure 4.7: la Requête Demandée par l'Utilisateur.

On a comme résultats les classes suivantes :reserveravion, reserverhotel et reservervoiture
Si on sélectionne une classe comme par exemple reserverhotel on obtient les rubriques ou les différents services qui répondent à la requête donnée par l'utilisateur contenant leurs différentes qualités de QOS.

The screenshot shows the 'Services Web' table with the following data:

ID	Nom	Fournisseur	Temps de ...	Fiabilité
1	Hotelerum...	Hotelerum	110.38	45.3
2	Comfy Hot...	OrgBusine...	509.2	28.3
3	ResAvenue	Avenues	481.0	41.11
4	Hotel Res...	CMSplaza	1024.1	18.6
5	Compass	OnPeak	363.4	60.7
6	Booking En...	ReservHO...	563.46	54.9
7	Hotel Booki...	StivaSoft	771.34	38.7
8	Weltraum ...	Weltraum ...	192.5	60.0
9	RoomScope	RoomScope	300.67	43.33
10	Trawex Clo...	Trawex Tec...	204.92	71.4

Figure 4.8:Liste des services candidats de la classe reserverhotel.

Si aucune Base ne correspond à la Requête donnée ce message est apparaître.

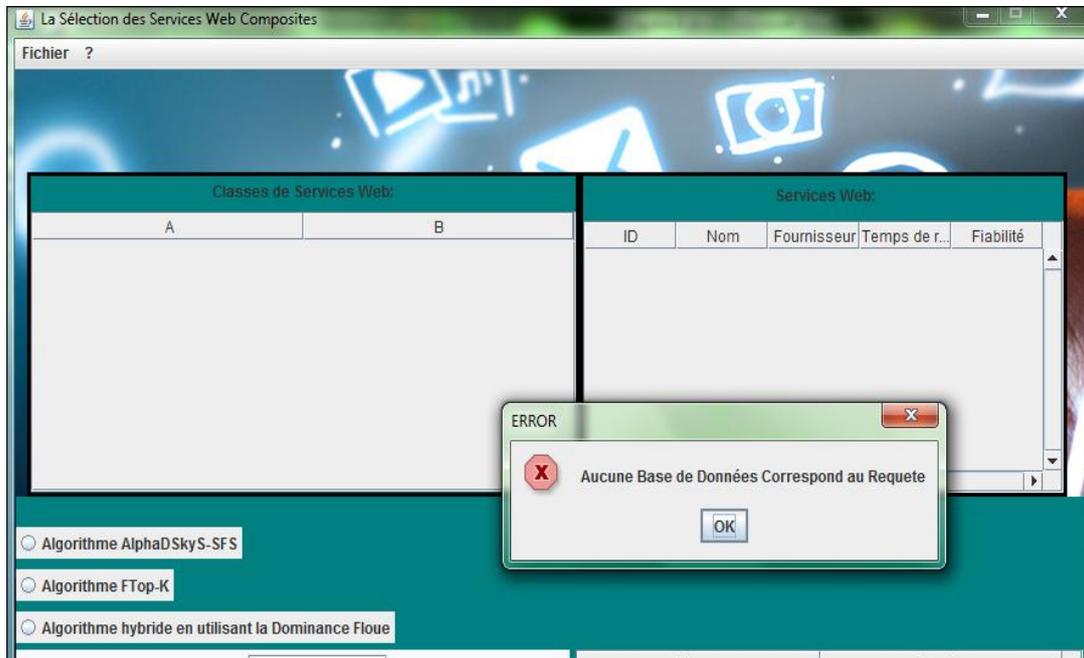


Figure 4.9 :Liste des services candidats de la classe reserverhotel.

Les critères de qualité de services utilisés dans notre prototype sont :

- A. **Temps de réponse** (ResponseTime) : est une performance qui représente la vitesse avec laquelle un service Web répond à une requête.
- B. **Fiabilité** (Reliability) : est la capacité d'un service à remplir ses fonctions requises dans les conditions indiquées pour une période de temps déterminée.
- C. **Coût d'un service** (Cost) : est le coût à payer pour consommer le service, ce coût peut être fourni par le fournisseur du service.
- D. **Débit** : Nombre total d'invocations / période de temps.

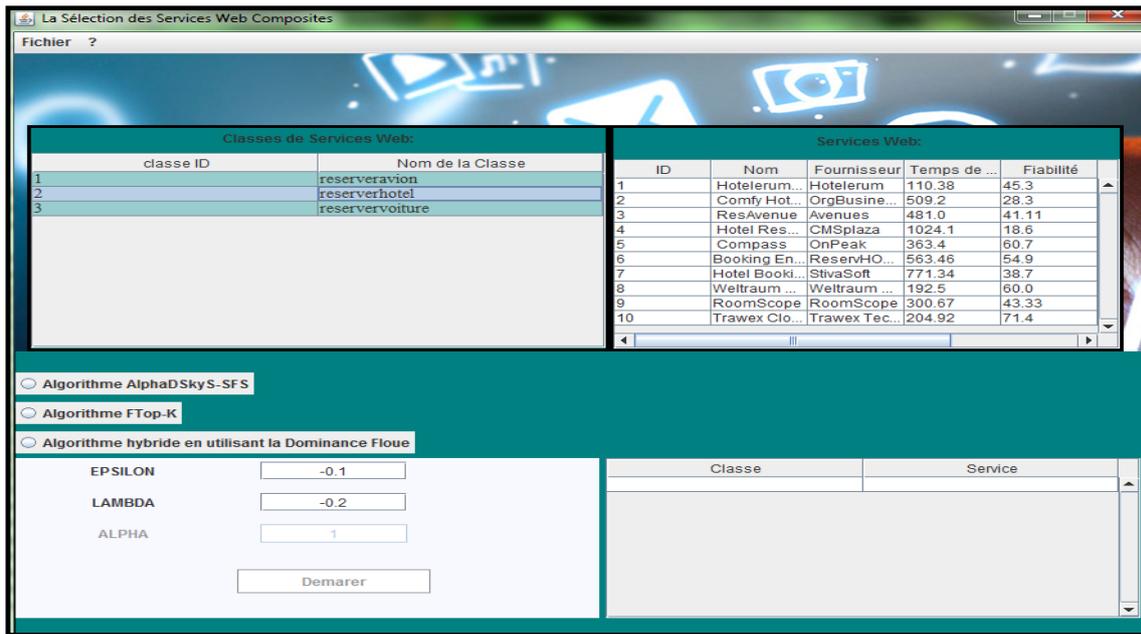


Figure 4.10 : Les Services Candidats de la classe reserverhotel et leurs QOS.

4.4 Etude de Cas 1

On fixe les paramètres avec l'utilisation des différents algorithmes.

4.4.1 Algorithme FTop-KS « Top-K en utilisant la Dominance Floue »

Selon les paramètres ϵ, λ on obtient le résultat suivant :

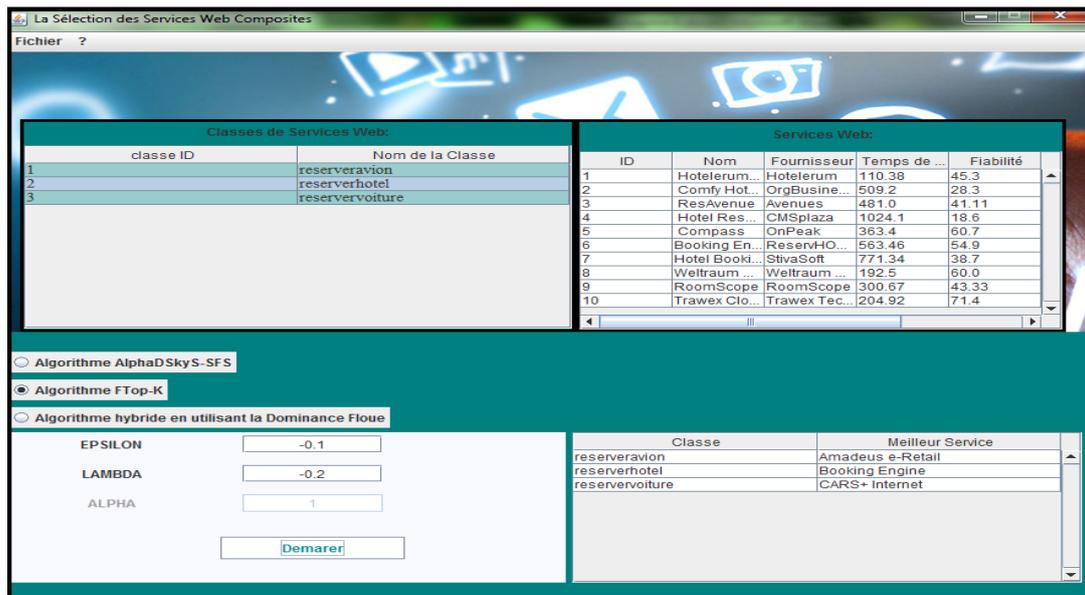


Figure 4.11: Algorithme FTop-KS « Top-K en utilisant la Dominance Floue ».

4.4.2 Algorithme α -DSkyS « Skyline en utilisant la Dominance Floue »

Selon les paramètres ϵ, λ et α on obtient le résultat suivant :

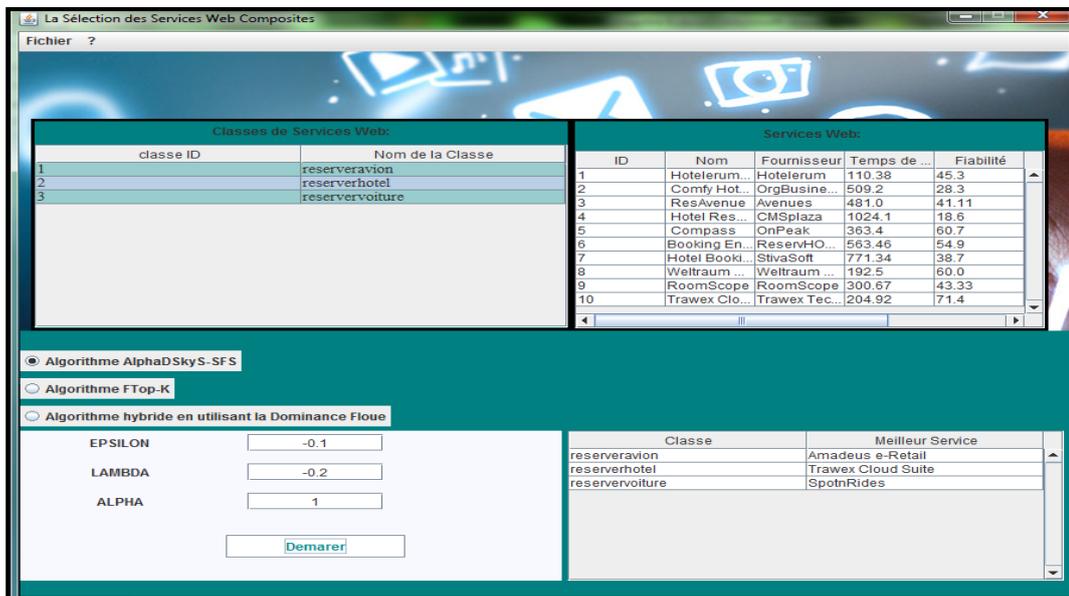


Figure 4.12: Algorithme α -DSkyS-SFS « Skyline en utilisant la Dominance Floue ».

4.2.3 Algorithme Hybride en utilisant la Dominance Floue

Cet algorithme commence par l'algorithme Skyline et se termine par l'algorithme Top-K afin de sélectionné le meilleur service web composites on tenant on considération les paramètres ϵ, λ et α et on a comme résultat comme suit :

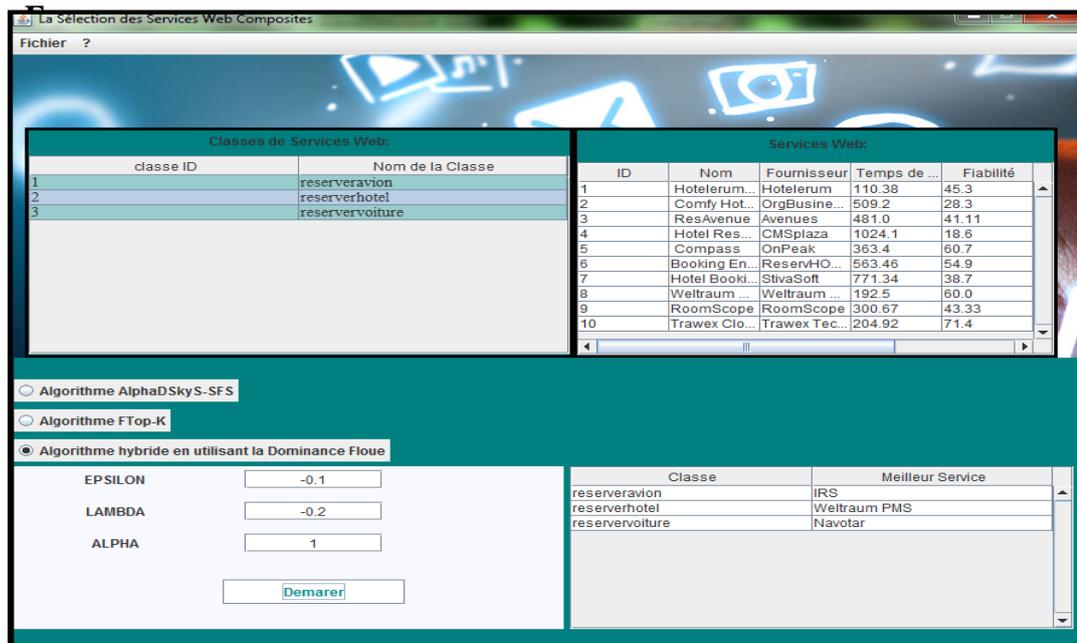


Figure 4.13: Algorithme Hybride 'FTop-KSky' en utilisant la Dominance Floue.

4.5 Etude de Cas 2

On changeons les paramètres ϵ, λ et α et on fixe l'algorithme

4.5.1 Exemple 1 :

$\epsilon = -0.1$, $\lambda = -0.2$ et $\alpha = 0.5$ on fixe l'Algorithme Hybride en utilisant la Dominance Floue on trouve comme résultats comme suit :

The screenshot displays the following data:

Classes de Services Web:		Services Web:				
classe ID	Nom de la Classe	ID	Nom	Fournisseur	Temps de ...	Fiabilité
1	reserveravion	1	Hotelerum...	Hotelerum	110.38	45.3
2	reserverhotel	2	Comfy Hot...	OrgBusine...	509.2	28.3
3	reservervoiture	3	ResAvenue	Avenues	481.0	41.11
		4	Hotel Res...	CMSplaza	1024.1	18.6
		5	Compass	OnPeak	363.4	60.7
		6	Booking En...	ReservHO...	563.46	54.9
		7	Hotel Bookl...	SilvaSoft	771.34	38.7
		8	Welltraum ...	Welltraum ...	192.5	60.0
		9	RoomScope	RoomScope	300.67	43.33
		10	Trawex Clo...	Trawex Tec...	204.92	71.4

Classe	Meilleur Service
reserveravion	Amadeus e-Retail
reserverhotel	Trawex Cloud Suite
reservervoiture	PROVAB

Figure 4.14 : Exemple 1.

4.5.2 Exemple 2 : les paramètres $\epsilon=-0.3$, $\lambda=-0.2$ et $\alpha=0.7$.

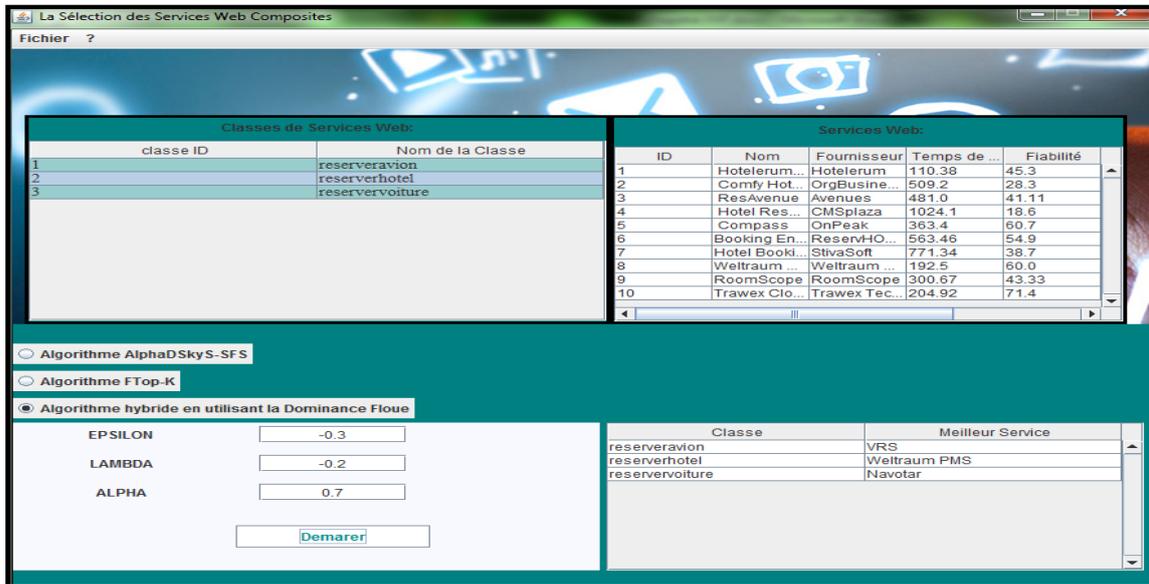


Figure 4.15 : Exemple 2.

4.5.3 Exemple 3: les paramètres $\epsilon=-0.3$, $\lambda=-0.1$ et $\alpha=0.4$.

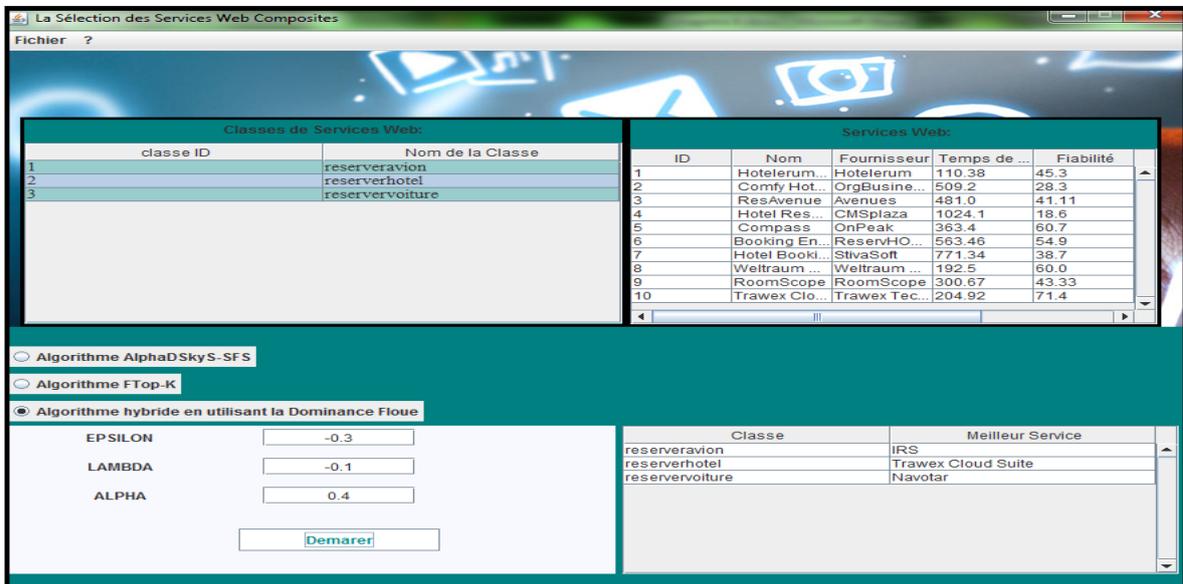


Figure 4. 16: Exemple 3.

Conclusion générale

Conclusion et perspectives

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.

La sélection des services web est l'une des problématiques les plus importantes de l'architecture orientée service. Elle constitue aussi l'une des étapes les plus importantes dans le processus de composition. Au cours de ce mémoire, nous nous sommes intéressés au problème de sélection des services web dans une composition sur la base des besoins non fonctionnels (QoS).

Dans ce mémoire, nous avons présenté un travail qui a pour but de faire une optimisation multi-objective basée sur les techniques avancées des bases de données en utilisant la dominance floue. Cette méthode utilise trois algorithmes : l'algorithme 'FTop-KS' pour sélectionner les Top-k services Web basée sur la dominance floue, l'Algorithme ' α -DSkyS' qui utilise la requête Skyline basée sur la dominance floue. Ainsi un algorithme hybride de deux techniques Top-k et Skyline qui repose sur la solution du premier algorithme ' α _DSKyS' après on applique l'algorithme 'FTop-KS'. Ce qui permet la réduction de l'espace de recherche et à la participation des meilleurs services dans le processus de composition.

« Conclusion générale »

Plusieurs améliorations et extensions peuvent être envisagées pour enrichir l'approche proposée :

- La prise en considération des aspects et des critères relatifs au Cloud : définir la sélection dans le contexte du Cloud Computing où d'autres critères de sélection de services sont à prendre en considération comme la distance entre services dans les Data Centers dans les Clouds.
- Intégrer l'incertitude dans la QoS : les valeurs des paramètres QoS ne sont pas déterministes, mais elles évoluent et changent en fonction de l'environnement du Service web. De ce fait, il serait judicieux de développer des algorithmes qui gèrent la QoS incertaine.
- Implémenter des algorithmes de calcul de Skyline plus performants : Il serait plus intéressant d'utiliser des algorithmes de type index comme l'algorithme BBS pour accélérer la recherche.

Bibliographies

Bibliographies

[1] : W3C World Wide Web Consortium; "Web Services Architecture"; W3C Working Group Note 11; February 2004; <http://www.w3.org/TR/ws-arch>

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

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

[4] : Appui Management de l'Economie. Programme de coopération MEDA. Développement d'applicatifs métiers pour le MTP. "Mise en oeuvre des services web avec JAX-WS".

[5] : KHELLAF Radhia, "verification de la compatibilite des services web pour une composition ", Thèse Magister, Université Constantine2, 2014 .

[6] : « <http://openclassrooms.com/coures/les-services-web> ».

[7] : J. Pathak. Interactive and verifiable web services composition, specification reformulation and substitution. Phd Thesis in Computer Sciences. Iowa State University 2007.

[8] : ALILI Sofian , et Mr ZIANI CHERIF Bassim, "La sélection des services Web À base d'algorithme de la recherche Harmonique" Mémoire MASTER, Université Abou Bekr Belkaid Tlemcen, 2015.

[9] : DEHANE Aicha Djihad, et Melle KEBIR Zohra, "Evaluation des techniques de codage d'ontologies sur les performances de la composition de services web", Thèse Master, Université Abou Bekr Belkaid Tlemcen, 2012.

[10] : KABACHE.N , Mr ZIADI .Ahmed Khaled, "Selection et Composition de Services Web avec Respect des Contraintes d'Utilisateur et Qualité de Service", Mémoire MASTER, Université Abderrahmane Mira Bejaia. 2018.

[11] : Ghaffour Ayoub et Tahir Fouzi " Selection Des Services web : Une approche à base de Skylines et Clustering Hiérarchique Ascendant " , Mémoire MASTER, Université Abou Bekr Belkaid Tlemcen, 2013.

«Bibliographies»

[12] : Hadjila, F. (2014). *Composition et interopération des services web sémantiques*. PhD thesis.

[13] : Alrifai, M., Risse, T., and Nejdl, W. (2012). A hybrid approach for efficient web service composition with end-to-end qos constraints. *ACM Transactions on the Web (TWEB)*, 6(2) :7.

[14] : Jatoth, C., Gangadharan, G., and Buyya, R. (2015). Computational intelligence based qos-aware web service composition : A systematic literature review. *IEEE Transactions on Services Computing*.

[15] : Khanouche, M. E., Amirat, Y., Chibani, A., Kerkar, M., and Yachir, A. (2016). Energy-centered and qos-aware services selection for internet of things. *IEEE Transactions on Automation Science and Engineering*, 13(3) :1256–1269.

[16] : KAMAL, S., IBRAHIM, R., and GHANI, I. (2014). Review on service selection schemes based on user preferences. *Journal of Theoretical & Applied Information Technology*, 70(2).

[17] : Yu, T. and Lin, K.-J. (2005a). Service selection algorithms for composing complex services with multiple qos constraints. In *International Conference on Service-Oriented Computing*, pages 130–143. Springer.

[18] : Liu, M., Wang, M., Shen, W., Luo, N., and Yan, J. (2012). A quality of service (qos)-aware execution plan selection approach for a service composition process. *Future Generation Computer Systems*, 28(7) :1080–1089.

[19] : Gabrel, V., Manouvrier, M., and Murat, C. (2014). Optimal and automatic transactional web service composition with dependency graph and 0-1 linear programming. In *International Conference on Service-Oriented Computing*, pages 108–122. Springer.

[20] : Liu, D., Shao, Z., Yu, C., and Fan, G. (2009). A heuristic qos-aware service selection approach to web service composition. In *Computer and Information Science, 2009. ICIS 2009. Eighth IEEE/ACIS International Conference on*, pages 1184–1189. IEEE.

[21] : Rodriguez-Mier, P., Mucientes, M., and Lama, M. (2011). Automatic web service composition with a heuristic-based search algorithm. In *Web Services (ICWS), 2011 IEEE International Conference on*, pages 81–88. IEEE.

«Bibliographies»

[22] : Stewart, B. S. and White III, C. C. (1991). Multiobjective a. *Journal of the ACM (JACM)*, 38(4) :775–814.

[23] : Xia, Y., Chen, P., Bao, L., Wang, M., and Yang, J. (2011). A qos-aware web service selection algorithm based on clustering. In *Web Services (ICWS), 2011 IEEE International Conference on*, pages 428–435. IEEE.

[24] : Lecue, F. and Mehandjiev, N. (2009). Towards scalability of quality driven semantic web service composition. In *Web Services, 2009.ICWS 2009. IEEE International Conference on*, pages 469–476. IEEE.

[25] : Klein, A., Ishikawa, F., and Honiden, S. (2011). Efficient heuristic approach with improved time complexity for qos-aware service composition. In *Web Services (ICWS), 2011 IEEE International Conference on*, pages 436–443. IEEE.

[26] : Moustafa, A. and Zhang, M. (2013). Multi-objective service composition using reinforcement learning. In *International Conference on Service-Oriented Computing*, pages 298–312. Springer.

[27] : Feng, L.-i., Obayashi, M., Kuremoto, T., Kobayashi, K., and Watanabe, S. (2013). Qos optimization for web services composition based on reinforcement learning. *International Journal of Innovative Computing, Information and Control*, 9(6) :2361–2376.

[28] : Hadjila, F., Chikh, M. A., and Merzoug, M. (2012). Qos-aware service selection based on clonal selection. In *Proceedings of ICACIS '12 Batna. Algerie. 2012*.

[29] : Zhao, X., Wen, Z., and Li, X. (2014). Qos-aware web service selection with negative selection algorithm. *Knowledge and Information Systems*, 40(2) :349–373.

[30] : Liu, Y., Miao, H., Li, Z., and Gao, H. (2011). Qos-aware web services composition based on hqpso algorithm. In *Computers, Networks, Systems and Industrial Engineering (CNSI), 2011 First ACIS/JNU International Conference on*, pages 400–405. IEEE.

[31] : Liu, X. and Yin, Z. (2009). Web service composition with global constraint based on discrete particle swarm optimization. In *Web Mining and Webbased Application, 2009. WMWA '09. Second Pacific-Asia Conference on*, pages 183–186. IEEE.

«Bibliographies»

[32] :Zhao, X., Song, B., Huang, P., Wen, Z., Weng, J., and Fan, Y. (2012). An improved discrete immune optimization algorithm based on pso for qos-driven web service composition. *Applied Soft Computing*, 12(8) :2208–2216.

[33] : Wang, P., Chao, K.-M., and Lo, C.-C. (2010a). On optimal decision for qos-aware composite service selection. *Expert Systems with Applications*, 37(1) :440–449.

[34] : Yu, Y., Ma, H., and Zhang, M. (2013). An adaptive genetic programming approach to qos-aware web services composition. In *Evolutionary Computation (CEC), 2013 IEEE Congress on*, pages 1740–1747. IEEE.

[35] : Liu, H., Zhong, F., Ouyang, B., and Wu, J. (2010). An approach for qos-aware web service composition based on improved genetic algorithm. In *Web Information Systems and Mining (WISM), 2010 International Conference on*, volume 1, pages 123–128. IEEE.

[36] : 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.

[37] : Benouaret, K., Benslimane, D., and Hadjali, A. (2011). On the use of fuzzy dominance for computing service skyline based on qos. In *Web Services (ICWS), 2011 IEEE International Conference on*, pages 540–547. IEEE.

[38] : Hadjila (2014). Composition et interopération des services web sémantiques.PhD thesis, université de Tlemcen.

[39] : Chomicki, J., Godfrey, P., Gryz, J., and Liang, D. (2005). Skyline with presorting : Theory and optimizations. In *Intelligent Information Processing and Web Mining*, pages 595–604. Springer.

[40] : Halfaoui.A , *La sélection des services web dans une composition à base de critères non fonctionnels ,Thèse Doctorat*. Université Abou Bekr Belkaid Tlemcen, 2017.

[41] :URL"<https://www.java.com>"+"[https://fr.wikipedia.org/wiki/Eclipse_\(projet\)](https://fr.wikipedia.org/wiki/Eclipse_(projet)) "Consulté le10/06/2019 ".

«*Bibliographies*»

[42] : Okat.B(2018),Une Approche Personnalisé pour La Selection de Web en tenant compte des contraintes Temps Reel .Memoire MASTER. Universite Mohamed Khider-BISKRA.

[43] :"https://fr.wikipedia.org/wiki/MySQL_Workbench"+"<https://fr.wikipedia.org/wiki/phpMyAdmin>"+"https://en.wikipedia.org/wiki/Visual_Paradigm", Consulté le 06/06/2019