RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

UNIVERSITE
ECHAHID HAMMA LAKHDAR - EL OUED
FACULTÉ DES SCIENCES EXACTES
Département D'Informatique

Mémoire de Fin D'étude

Présenté pour l'obtention du Diplôme de

MASTER ACADEMIQUE

Domaine : **Mathématique et Informatique**

Filière : **Informatique**

Spécialité : **Systèmes Distribués et Intelligence Artificielle**

Présenté par : Bekakra Aicha

Thème

Une approche parallèle pour la sélection des services web basée sur la logique floue

.

Soutenue le *22-09-2021* Devant le jury :

M. Farida Retima	MAA	Encadreur
M. Mouadh Bali	MCA	Président
M. Ammar Boucherit	MAA	Examinateur

Année Universitaire : 2020/2021

Made with LaTeX.

Dédicaces

Au propriétaire de la biographie parfumée et de la pensée éclairée, il a reçu le grand crédit après Dieu tout-puissant pour être arrivé à ce stade, mon père bien-aimé , que Dieu le sauve où qu'il soit.

À celui qui a fait le premier pas et l'exemple du rôle éclairé, je demande à Dieu de me donner sa justice, ma précieuse mère , que Dieu la sauve et prenne soin d'elle.

À l'amie de l'invitation permanente et à la deuxième mère ma sœur Radia.

Au Sindh et au deuxième père, mon frère Abdel Kamel .

À ma pom-pom girl dans les premières rangées et à l'inspiration, ma sœur Salima.

Au compagnon du chemin et au jumeau de l'âme Dadi.

À la mère de Fadi Saberin .

À mes frères... Abdelkader Albachir et Laiche

À mes frères et sœurs, chacun en son nom.

Aux femmes de mes frères

Au bonheur de la maison Ikram

Bekakra Aicha

Remerciement

Tout d'abord, merci Dieu toujours et jamais sur lui et l'honorer d'avoir terminé cette recherche scientifique. Je demande à Dieu que les étudiants en sciences lui profitent et en fassent une charité pour mon âme, ma pure tante, que Dieu la bénisse.

Deuxièmement, je remercie tous ceux qui m'ont aidé de près ou de loin, en particulier M. Atia Ibrahim, pour ce qu'il a dit et trouvé, et je remercie mon amie Mabrouka pour sa patience avec moi, car je n'oublie pas le professeur farida Retima qui supervise la bonne supervision et le suivi.

Enfin, je remercie tous les professeurs du de la Faculté de l'Université El Oued, et chaque de systèmes distribués et d'intelligence artificielle 2021 et tous les amis et proches.

Résumé

Les applications distribuées convergent rapidement vers l'adoption du paradigme orienté service (SOA) , les services web constituent le moyen le plus convenable pour mettre en œuvre cette architecture. ce pendent parmi plusieurs services similaires [12] , la sélection d'un service web est devenue une tâche composante très importante pour satisfaire le besoin de l'utilisateur,différence entre ces services web c'est leurs propriétés non fonctionnelles c.-à-d. (la qualité de service (QoS)) qui influencent directement sur la sélection du meilleur service.

Dans ce mémoire, nous considérons le problème de sélectionner le service web adéquate en fonction des préférences de satisfaction du consommateur.
Pour résoudre ce problème, nous proposons les contributions suivantes selon deux axes : premièrement, nous avons proposé un MapReduce Skyline pour accélérer le filtrage des services web et introduire le parallélisme pour traiter un grand nombre des services web. Deuxièmement, nous avons utilisé la logique floue comme un outil aide à décision pour sélectionner le meilleur service.

Mots-clés :Architecture Orienté Service (SOA), Service web, sélection des services web, QoS, Logique floue, MapReduce, Skyline.

الملخص

تتقارب التطبيقات الموزعة بسرعة نحو اعتماد مفهوم البنية الموجهة للخدمات (SOA) لأن خدمات الويب هي أنسب طريقة لتنفيذ هذه البنية. بينما، من بين العديد من الخدمات المماثلة، ، أصبح اختيار خدمة الويب مهمة مكونة مهمة للغاية لتلبية احتياجات المستخدم، والفرق الرئيسي بين خدمات الويب هذه هو خصائصها غير الوظيفية، أي جودة الخدمة (QoS) التي تؤثر بشكل مباشر على اختيار أفضل خدمة.

في هذه المذكرة، نأخذ في الاعتبار مشكلة اختيار خدمة الويب الصحيحة بناءً على تفضيلات ترضي المستهلك. لحل هذه المشكلة، نقترح المساهمات التالية وفقا للمحورين : أولاً، اقترحنا ماب ريدوس سكاي لاين (Map Reduce Skyline) لتسريع تصفية خدمات الويب وإدخال التوازي من أجل معالجة عدد كبير من خدمات الويب في نفس الوقت. ثانيًا، استخدمنا المنطق الضبابي كأداة لدعم القرار لاختيار أفضل خدمة.

الكلمات المفتاحية : البنية الموجهة بالخدمات (SOA) ، خدمة الويب، اختيار خدمات الويب، جودة الخدمة، المنطق الضبابي، ماب ريدوس سكاي لاين (Map Reduce Skyline) .

Abstrait

Distributed applications are rapidly converging towards the adoption of the Service Oriented Paradigm (SOA), as web services are the most suitable way to implement this architecture. While among many similar services, the selection of a web service has become a very important component task to satisfy the user's need, the major difference between these web services is their non-functional properties, i.e. (quality of service (QoS)) which directly influence the selection of the best service.

In this brief, we consider the problem of selecting the correct web service based on consumer satisfaction preferences. To solve this problem, we offer the following contributions along two axes : first, we have proposed a Map Reduce Skyline to speed up the filtering of web services and introduce parallelism in order to process a large number of web services at the same time. Second, we used fuzzy logic as a decision support tool to select the best service.

Keywords : Service Oriented Architecture (SOA), Web service, selection of web services, QoS, Fuzzy logic, Map Reduce Skyline.

Table des matières

Table des figures

Liste des tableaux

Première partie

L'état de l'art

Introduction Générale

Contexte

La technologie a été toujours un secteur d'innovation continue, la communication et le partage de données et de ressources qui sont récemment un sujet qui fait l'objet de plusieurs travaux dans le domaine de la technologie[2].

Avec l'apparition 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 qualité de service (QdS ou QoS en anglais : Quality of Service). ces critères auront une grande importance lorsque le besoin des clients est complexe[11].

Le modèle SOA (Service Oriented Architecture) a été créé comme une réponse aux inconvénients des technologies orientées composantes.il utilise un ensemble de normes : SOAP (Simple Object Access Protocol), UDDI (Universal Description Discovery and Integration), WSDL (Web Services Description Language).cette architecture permet d'une manière flexible aux applications d'interagir les uns avec les autres sur les réseaux [12].

La sélection des services web est une étape cruciale pour satisfaire le besoin de l'utilisateur.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 [11].

Nous nous intéressons dans ce mémoire plus particulièrement à la phase de sélection de services web à base de qualité de service. cette phase consiste à choisir parmi les services web disponible, ceux qui répondent au mieux aux exigences de l'utilisateur sur la base des besoins non fonctionnels. 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[11].

Problématique

Avec le développement des technologies web et le cloud computing, un grand nombre des services web sont générées. elles peuvent être utilisées par les utilisateurs pour satisfaire leurs besoins. par conséquent, l'augmentation exponentielle des services web qui offrent le même type d'informations pour répondre aux exigences des utilisateur, pose un grand défi. donc, il est devenu important d'utiliser la qualité de service (QoS) comme un critère essentiel pour sélectionner le meilleur service web qui garantit la qualité requise par un utilisateur.
la recherche des services web basée sur une description qualitative permettant une recherche flexible où le service web retourné peut être proche à la demande désirée. en plus de cela, dans de nombreux cas, la valeur de la propriété QoS peut être difficile à définir avec précision. donc, il est préférable que les préférences utilisateur utilisées comme propriétés QoS soient floues, car mieux adaptées à l'interprétation en des termes linguistiques.
L'objectif de ce mémoire est la proposition d'une solution pour évaluer la QoS des services web pour sélectionner le meilleur service en fonction des préférences de satisfaction du consommateur de service. un autre objectif est d'accélérer le calcul et d'introduire le parallélisme pour traiter un grand nombre des services web disponibles.

Contributions

L'objectif de ce travail consiste à proposer une approche parallèle pour la sélection des services web basée sur la logique floue. la première contribution est la proposition d'une solution pour la sélection de service web basée sur la logique floue. cette proposition vise à représenter la QoS des services web et la QoS requise par l'utilisateur sous forme des termes linguistiques.

En outre, il permet l'évaluation des valeurs incertaines de QoS mesurées, et aide à effectuer la sélection de service web la plus proche au besoin d'utilisateur. la deuxième contribution est la proposition d'un MapReduce Framework pour accélérer le calcul de filtrage des services web disponibles selon le besoin de l'utilisateur.cette proposition vise à appliquer la logique floue à un petit nombre de services web résultant de MapReduce plutôt qu'un grand nombre des services web.

Plan du mémoire : Ce mémoire est découpé en deux parties :
La première partie « **l'état de l'art** » :

Chapitre I : (les services web et leur architecture) : présente des concepts fondamentaux pour les services web.

Chapitre II : (la sélection des services web) : présente un aperçu des approches existantes dans la littérature pour la sélection des services web . La deuxième partie « **Contribution** » :

Chapitre III : (Conception) : ce chapitre décrit la conception de notre système.

Chapitre IV : (Implémentation et Expérimentations) : nous décrivons un prototype implémentant notre application ainsi que les résultats obtenus.

La conclusion générale : résume les résultats de notre travail, et présente quelques perspectives sur des travaux futurs.

Chapitre 1

SERVICES WEB ET LEURS ARCHITECTURES

1.1: Introduction

Dans ce chapitre, nous décrivons les services web : la définition, l'intérêt et l'architecture Architecture Orientée Services (SOA), 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. et enfin, nous terminons notre chapitre par les avantages et les inconvénients de services web.

1.2: Définitions des services web

Un service web est un composant logiciel autonome permettant la communication et l'échange de données entre des applications et des systèmes hétérogènes dans des environnements distribués. la combinaison entre un ensemble de normes web avec des langages dérivés de XML permet le déploiement à distance, la découverte et finalement la pagination. il s'agit donc d'un ensemble de fonctions proposées sur Internet. généralement, un service web est présenté comme une interface publique qui permet la communication avec des applications et d'autres services web à des messages a base de language XML [12].

Selon IBM 1 Les services web sont la nouvelle vague d'applications web. ce sont des applications standard, autonomes et autodescriptives qui peuvent être déployées, localisées et appelées à partir du web. les services web exécutent des processus allant de simples demandes à des processus métier complexes. [1, 12].

Selon W3C 2 Un service web est un composant logiciel identifié par une Uniform Ressource Identifier (URI) conçu pour prendre en charge l'interopérabilité de machine à machine sur un réseau. il contient une interface décrite en Web Service Description Language (WSDL). des systèmes interagissent avec un service web à l'aide de messages Simple Object Access Protocol (SOAP). généralement en utilisant le protocole HyperText Transfer Protocol (HTTP) avec une séquence XML ainsi que d'autres normes sur le web. [3, 12].

Les différentes définitions proposées s'accordent sur l'idée qu'un service web est un nouveau type de composant logiciel. de rendre ces services facilement invocables et de les mettre à disposition par l'intermédiaire de protocoles Internet standardisés (HTTP, XML).

1.3: Architecture Orientée Services (SOA)

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 présenté dans la figure 1.1 implique trois entités d'interaction : les fournisseurs de services, les annuaires de services et les demandeurs des services [2].

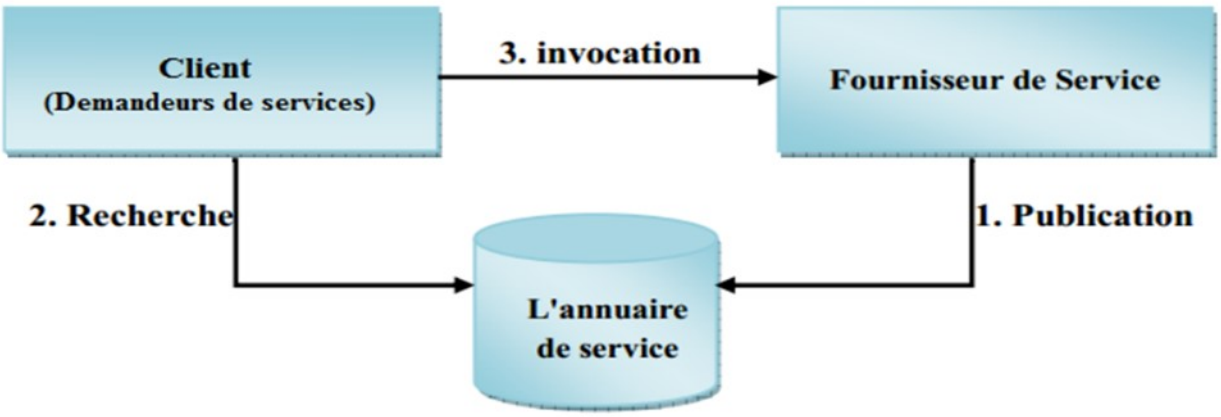

FIGURE 1.1 – Modèle fonctionnel de l'architecture SOA [2].

1.3.1: But d'architecture orientée service :

L'Architecture Orientée Services (SOA – Service-Oriented Architecture en anglais) permet aux concepteurs de systèmes d'information d'organiser un ensemble de logiciels isolés en un ensemble de services interconnectés, accessibles par une interface et des protocoles standard.

1.3.2: Avantage de l'architecture orientée service :

L'avantage principal de l'architecture orientée services en général et Web en particulier, est que ce paradigme pourrait répondre aux besoins en termes de flexibilité et d'adaptabilité rapide exprimés par les concepteurs .

L'architecture SOA comprend trois acteurs :

- **Fournisseur du service :** Le fournisseur de service met en application le service web et le rend disponible sur Internet. il désigne l'entité propriétaire du service.
- **Client ou demandeur du service :** 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).
- **L'annuaire de service :** L'annuaire de services est un registre de services. ce registre fournit un endroit central où les programmeurs peuvent publier de nouveaux services ou pour les découvrir. en d'autres termes, l'annuaire joue le rôle d'intermédiaire entre les clients et les fournisseurs de services.

Les interactions entre ces trois acteurs est composées de plusieurs étapes (recherche, publication, invocation) :

- **Publication du service :** le fournisseur diffuse les descriptions de ses services web dans l'annuaire.

— **Recherche du service :** le client cherche un service particulier, il s'adresse à l'annuaire qui va lui fournir les descriptions et les URL des services demandés afin de lui permettre de les invoquer.
— **L'invocation du service :** une fois que le client récupère l'URL et la description du service, il les utilise pour l'invoquer auprès du fournisseur de service.

1.4: Caractéristiques des services web

Les services web possèdent des caractéristiques qui leurs permettent une meilleure intégration dans les environnements hétérogènes :

1. **Interopérabilité :**
   l'interopérabilité est un critère primordial pour les services web. elle permet aux clients de communiquer avec des services programmés dans des langages de programmation différents et de s'exécutent sur des plateformes (logicielles ou matérielles) différentes. les services web se basent sur des standards XML pour simplifier la construction des systèmes distribués et la coopération entre ces derniers .
2. **Réutilisable :**
   Le principe de réutilisabilité permet de réduire le coût de développement en favorisant la réutilisation des parties de codes qui ont déjà été réalisées. Par exemple, une fonctionnalité est développée sous forme de service web peut être réutilisée et combinée à d'autres fonctionnalités afin de composer de nouveaux services pour satisfaire une partie des besoins .ceci permet un gain de temps considérable, une réduction importante de la taille des applications par éviter la duplication et facilite également la maintenance .
3. **Simplicité d'utilisation :**
   L'utilisation des standards tels que XML et HTTP a mis en valeur la simplicité de la manipulation des services web, comme ils sont adaptés aux systèmes interconnectés d'une manière flexible.
4. **Couplage faible :**
   Le couplage est une métrique indiquant le degré d'interaction entre deux ou plusieurs composants logiciels. le couplage faible est un concept qui permet de minimiser les dépendances et assurer l'évolution, la flexibilité et la tolérance. cela signifie qu'il y a une séparation logique qui isole une application cliente et un service afin d'éviter une dépendance physique entre les deux. c'est à dire le demandeur du service ne connaît pas forcément le fournisseur. pour ce faire, le client communique avec le service par échange de messages dans un format standard. il est ainsi possible de modifier un service sans briser la compatibilité avec les applications clients existantes .

5. **Modulaire :**
   Les services web fonctionnent de manière modulaire et non pas intégrée. cela signifie qu'au lieu d'intégrer dans une seule application globale toutes les fonctionnalités, on crée (ou on récupère) plusieurs applications spécifiques qu'on les fait inter-opérer entre elles, et qui remplissent chacune, une de ces fonctionnalités .
6. **Autonome et auto descriptif :**
   Le cadre des services web contient en lui-même toutes les informations nécessaires à l'utilisation des applications, sous la forme de trois fonctions : trouver, décrire et exécuter dans une manière facilement reconnu par n'importe quelle application externe .la capacité des services à se décrire par eux-mêmes permet d'envisager l'automatisation de l'intégration de services [17].

1.5: Technologie des services web

Les services web utilisent 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. la Figure 1.2 présente les technologies utilisées pour les services web[10].

FIGURE 1.2 – Technologies des composants des services web [10].

Extensible Markup Language (XML) :

XML est un standard promulgué par le W3C. on retrouve dans XML une généralisation des idées contenues dans HTML et SGML. XML permet de définir des balises extensible et indépendante de l'aspect graphique que doit revêtir la page dans le navigateur web. Dans XML, les balises permettent d'associer toutes sortes d'informations au fil du texte. il a été conçu pour des documents arbitrairement complexes, tout en s'appuyant sur cinq grands principes simples et clairs :

— lisibilité à la fois par les machines et par les utilisateurs.
— définition sans ambiguïté du contenu d'un document.
— définition sans ambiguïté de la structure d'un document.
— séparation entre documents et relation entre documents.
— séparation entre structure des documents et présentation des documents.

Simple Object Access Protocol (SOAP) :

SOAP est une spécification de communication entre les services web, par échange de messages en XML au travers du web. il est simple et facile à implémenter dans les serveurs web ou dans les serveurs d'application. SOAP est indépendant des langages de programmation et des systèmes d'exploitation utilisés pour l'implémentation des services web.SOAP est un protocole de communication entre les applications fondé sur XML, visant à satisfaire deux objectifs principales : servir un protocole de communication sur les intranets, et permettre la communication entre applications et services web, en particulier dans un contexte d'échange inter-entreprise (le réseau internet) .

Web Services Description Langage (WSDL) :

WSDL est un langage qui permet de décrire les services web,en particulier, les interfaces des services web. ces descriptions sont des documents XML. WSDL décrit un service web en deux étapes fondamentales : une abstraite et une concrète. dans chaque étape, la description utilise un nombre de constructions pour favoriser la réutilisation de la description et pour séparer les préoccupations de conception indépendantes (Figure 1.4)[10].

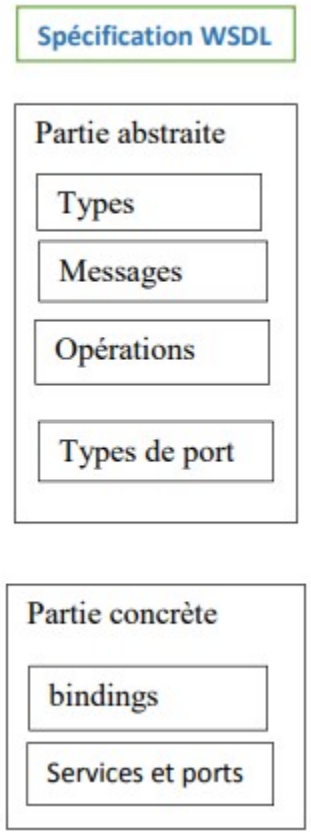

FIGURE 1.3 – Spécification d'un service web avec WSDL [10] .

Un document WSDL dé nit une suite de descriptions de composants. le chier WSDL est principalement composé de 6 éléments, avec d'autres éléments optionnels :

- **Définitions :** élément racine du document, il donne le nom du service, déclare les espaces de noms utilisés, et contient les éléments du service (obligatoire).
- **Types :** décrit tous les types de données utilisés entre le client et le prestataire. WSDL n'est pas exclusivement lié à un système spécifique de typage, mais utilise par défaut la spécification XML schéma (définition abstraite).
- **Message :** décrit un message unique, que ce soit un message de requête seul ou un message de réponse seul. l'élément dé nit le nom du message et peut contenir (ou pas) des éléments " part ", qui font référence aux paramètres du message ou aux valeurs retournées par le message (définition abstraite) .
- **PortType :** combine plusieurs messages pour composer une opération.chaque opération se réfère à un message en entrée et à des messages en sortie (définition abstraite).
- **Binding :** décrit les spécifications concrètes concernant la manière dont le service sera implémenté : protocole de communication et format des données pour les opérations et messages définis par un type de port particulier. WSDL possède des extensions internes pour définir des services SOAP ; de ce fait, les informations spécifiques à SOAP se retrouvent dans cet élément (définition concrète).
- **Service :** dé ni les adresses permettant d'invoquer le service donné, ce qui sert à regrouper un ensemble de ports reliés. la plupart du temps, c'est une URL invoquant un service SOAP (définition concrète).

Universal Description Discovery and Intégration (UDDI) :

L'objectif primaire d'UDDI est la spécification d'un canevas pour décrire et découvrir des services web. le noyau d'UDDI travaille avec la notion de "business registry", qu'est un service sophistiqué de noms et répertoires. plus précisément, UDDI définit des structures de données et APIs pour publier les descriptions des services dans le registre et pour interroger le registre afin de chercher des descriptions publiées. parce que les APIs de UDDI sont aussi spécifiés en WSDL avec une attache SOAP, le registre peut être invoqué comme un service web (en conséquence, ses caractéristiques peuvent être décrites dans le même registre, comme un autre service). les spécifications du registre UDDI ont deux buts principaux en ce qui concerne la découverte d'un service : le premier, soutenir les développeurs dans la découverte d'informations sur les services. le deuxième, permettre la liaison dynamique et en conséquence habiliter les clients pour interroger le registre et obtenir des références aux services d'intérêt [14].

1.6: Architectures des services web

Pour permettre l'interopérabilité entre les applications distantes, quels que soient les systèmes d'exploitation et les langages de programmation, il existe deux types d'architecture pour les services web : la première, connue sous le nom d'architecture de référence, est traditionnellement utilisée pour les services web isolés et contient trois couches principales. la deuxième architecture est plus complète, elle utilise les couches standards de la première architecture en ajoutant d'autres couches plus spécifiques. c'est ce qu'on appelle une architecture étendue.

1.6.1: Architecture de référence

L'architecture classique de « référence » des services web est basée sur le modèle SOA (architecture orientée services) qui comprend trois acteurs : le client, l'annuaire des services, et le fournisseur de services. cette architecture a également trois objectifs principaux :

1. déterminer les composants fonctionnels.
2. définir les relations entre ces composants.
3. établir un ensemble de restrictions sur chaque composant pour garantir les caractéristiques générales de l'architecture.

Par conséquent, il représente l'environnement de déploiement et d'exécution du service. il est composé de trois couches de bases :

— **Couche de données :** elle contient les différentes bases de données utilisées par le service.
— **Couche applicative :** c'est la plateforme de développement qui assure la mise en œuvre du service web.
— **Couche de description :** elle affiche les fonctions de service via un fichier WSDL.

Cependant, cette infrastructure n'est pas suffisante pour permettre une utilisation effective des services web dans les domaines dont les exigences vont au-delà de la capacité d'interactions simples. par conséquent nous introduisons dans le paragraphe suivant une nouvelle architecture plus complète et plus adaptée au contexte du web.

1.6.2: Architecture étendue

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 etendue. une architecture étendue est constituée de plusieurs couches se superposent les unes aux autres. cette architecture est appelée « pile des services web ». cette pile est constituée de plusieurs couches, chaque couche s'appuyant sur un standard particulier. on retrouve, audessus de la couche de transport, les trois couches formant l'infrastructure de base décrite précédemment. nous apportons une explication des différents types de couches de l'architecture indiquée dans la (figure 1.5).

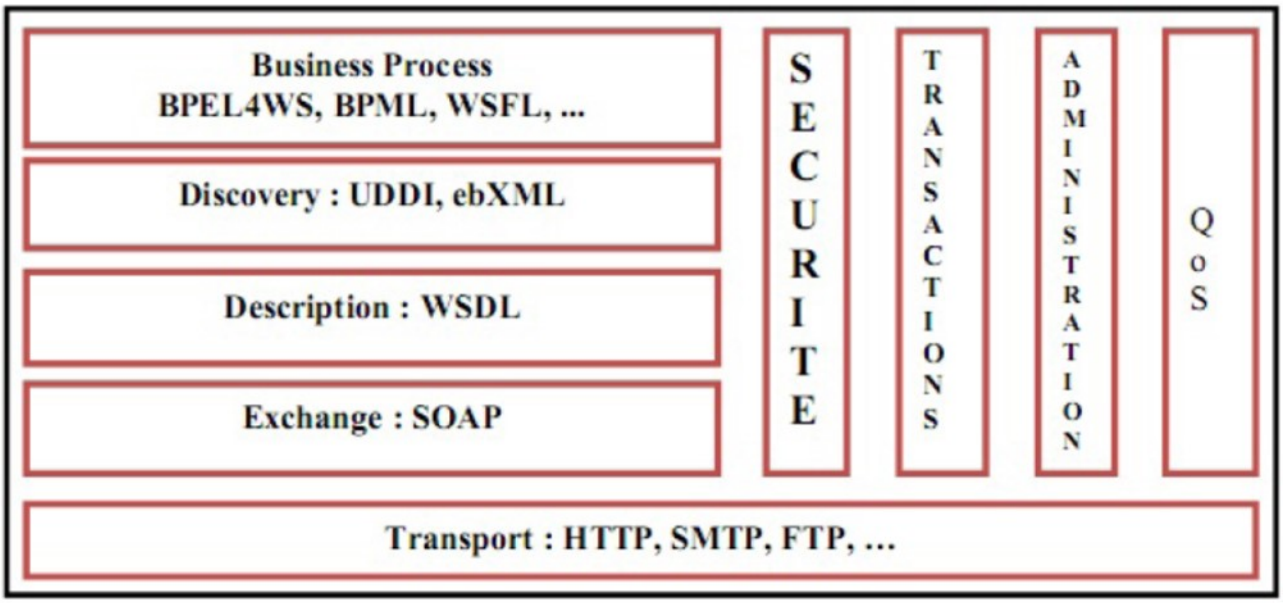

FIGURE 1.4 – Architecture en pile des services web [6].

1. **Couche transport :**
   Cette couche permet de déplacer les requêtes de service du consommateur de services au fournisseur, et les réponses des services du fournisseur au consommateur de service pour pouvoir se découvrir et dialoguer entre eux. le protocole le plus utilisé dans cette couche est HTTP. cependant, d'autres protocoles peuvent être utilisés, tels que le SMTP, FTP. . .
2. **Couche communication :**
   Cette couche est responsable du formatage des données échangées de sorte que les messages peuvent être compris à chaque extrémité (le fournisseur et le client de service pour communiquer entre eux). deux styles d'architecturaux totalement différents sont utilisés pour ces échanges de données. on a l'architecture orientée opérations distribuées (protocoles RPC) basée sur XML et qui comprend XML-RPC et SOAP. et l'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. ce langage est la notation standard basée sur XML pour construire la description de l'interface d'un service.
4. **Couche découverte de services :**
   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 web est assurée par un annuaire UDDI .

5. **Couche Business Processus (Business Process) :**
   Cette couche supérieure permet l'intégration de services web, elle établit la représentation d'un «Business Process» comme un ensemble de services web. de plus, la description de l'utilisation de différents services composant ce service est disponible par l'intermédiaire de cette couche.
6. **Couches transversales (Sécurité, Transactions, Administration, QoS) :**
   Ces couches rendent viable l'utilisation effective des services dans le monde industriel. ce sont les aspects concernant la qualité de services :

— **Sécurité :** La gestion de la sécurité est actuellement le défi le plus important à la mise en place d'architectures distribuées à base de services web. elle représente la normalisation des moyens permettent de couvrir les problématiques d'authentification et de gestion des droits d'accès. en plus, l'ensemble de règles qui peuvent être appliqués lors de d'authenti-fication, de l'autorisation, et du contrôle d'accès des consommateurs qui invoquent les services. les consommateurs peuvent aussi demander l'utilisation de règles de sécurité. par exemple, dans le cas où ils veulent garantir l'intégrité des données transmises.
— **Transaction :** Normalisation des moyens permettant de garantir l'intégrité des transactions longues impliquant plusieurs services web. un environnement d'intégration de services peut contenir un élément chargé de garantir la cohérence des données utilisées et des résultats retournés par une collaboration (ou composition) de services.
— **Administration :** Est le processus chargé de surveiller et de contrôler de manière proactive le comportement d'un service ou d'un ensemble de services[7].
— **QOS :** La qualité de service (QdS) ou quality of service (QoS) est la capacité à véhiculer dans de bonnes conditions un type de trafic donné, en termes de disponibilité, débit, délais de transmission, taux de perte de paquets .

1.7: Avantages et inconvénients des services web

Tout service web a ses avantages et ses inconvénients.

1.7.1: 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 :

— réduisent le temps de mise en marché des services offerts par les diverses entreprises.
— Utilisent des normes et protocoles ouverts.
— Grâce aux services web, les coûts sont réduits par l'automatisation interne et externe des processus commerciaux.
— 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.

1.7.2: Inconvénients des services web

La technologie des services web comporte plusieurs inconvénients dont :

— **Fiabilité :**Il est difficile d'assurer la fiabilité d'un service car on ne peut garantir que ses fournisseurs ainsi que les personnes qui l'invoquent.

— **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.

— **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.

— **Syntaxe et sémantique :**On se concentre beaucoup sur comment invoquer des services (syntaxe) et pas assez sur ce que les services web offrent (sémantique)[4].

1.8: Conclusion

La technologie des services web permet d'offrir des fonctionnalités a travers une communication à distance via l'internet. l'architecture orientée services nécessite un nouveau style de construction des applications informatiques distribués. ce style est conçu pour permettre l'interopérabilité de machine-à-machine sur un réseau. dans ce chapitre, on a défini le concept de "service web", leurs technologies, leur fonctionnement et ainsi quelques avantages et les inconvénients des services web. dans le chapitre suivant, nous aborderons l'approche de la sélection des services web et les méthodes utilisées.

Chapitre 2

SELECTION DES SERVICES WEB

2.1: Introduction

Avec la croissance explosive du nombre des services web publiés sur internet, il arrive très fréquemment que de nombreux services web répondent à un même besoin fonctionnel, mais qui offrent des propriétés de qualité de services différentes. une sélection de service web doit être alors réalisée pour déterminer quels sont les services web pertinents qui satisferaient les besoins d'un utilisateur. la qualité de service (QoS) est considérée comme le critère non fonctionnel plus important pour la sélection du service web.dans ce chapitre, nous présentons l'état de l'art qui introduit un problème spécifique émergeant qui est celui de sélectionner le meilleur service en respectant les contraintes de QoS. pour cela, nous décrivons les différentes stratégies de sélection de service web et nous citons quelques approches utilisées pour résoudre ce problème.

.

2.2: Sélection de services web basée sur la qualité de service :

La sélection de services web est l'une des discussions les plus importantes dans l'architecture SOA. donc, pour identifier les meilleurs services parmi un ensemble de services web découverts, il faut sélectionner les services les plus adéquats qui répondent aux préférences de l'utilisateur sur la base des besoins fonctionnels et non fonctionnels. parmi un groupe de services ayant des fonctions similaires, les besoins non fonctionnels des services web sont généralement exprimés à l'aide des critères de QoS.
Le fournisseur de service web publie le service dans un registre UDDI. ce registre est un lieu de stockage des services web avec des moyens pour faciliter la recherche.le consommateur de service demande un service web offert par le fournisseur.
Dans la sélection de services web basée sur la QoS deux cas se présentent : [Jaegerand Mühl,2006] : si la réponse à la requête d'un client exige la sélection d'un seul service web , alors la sélection est clair.le candidat qui présente la meilleure QoS sera désigné pour répondre à la demande du client. Si la réponse à la requête d'un client exige la combinaison de plusieurs services existants, alors la sélection dans ce cas sera plus complexe du fait qu'il faut choisir la combinaison des services qui répondent mieux aux besoins des clients.
Il existe plusieurs méthodes utilisées dans la littérature pour résoudre le problème de sélection de services basée sur la QoS, telles que la programmation linéaire, logique floue...etc.

Les interactions entre les différents rôles du modèle SOA pour la sélection de service web sont montrées dans Figure 2.1 il y a trois rôles dans ce modèle :

- Utilisateur ou agent (service, application).
- Registre de service UDDI.
- Fournisseur de services web.

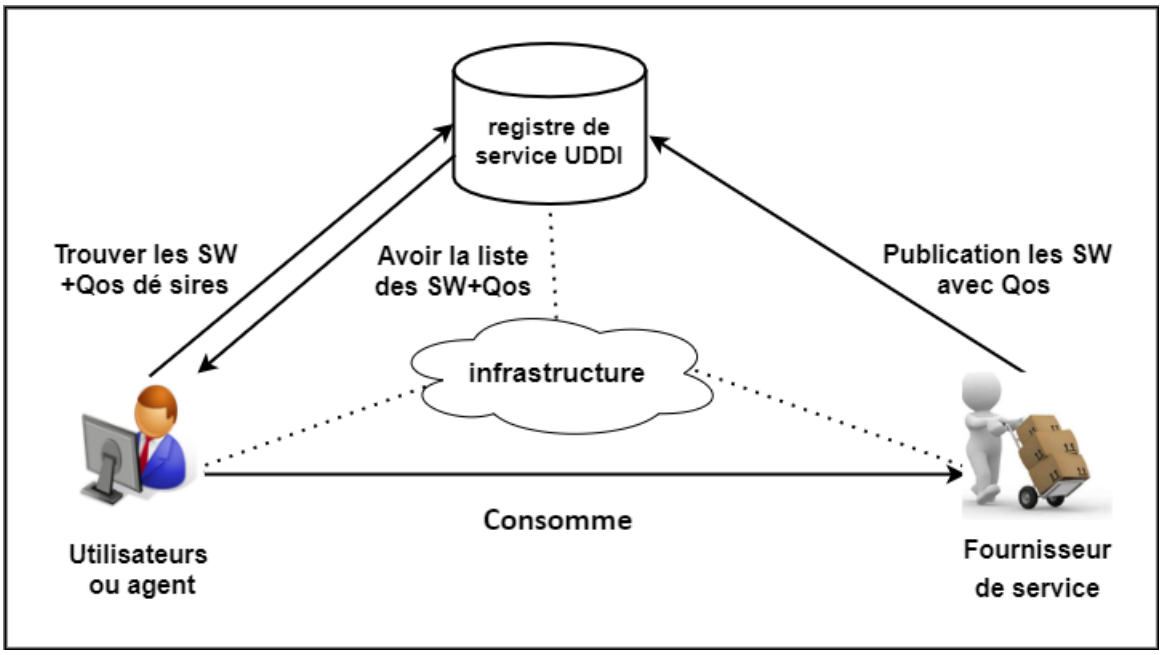

FIGURE 2.1 – Diagram sélection de services web basée sur la qualité de service.

2.3: Critères utilisés pour les méthodes de sélection de services web

Les méthodes utilisées pour choisir un service web sont discutées en relation avec un ensemble de caractéristiques, qui répondent le mieux aux exigences en matière de besoins fonctionnels et / ou non fonctionnels des utilisateurs.ces caractéristiques sont décrites comme suit :

2.3.1: Propriétés fonctionnelles

Les propriétés fonctionnelles d'un service web désignent les opérations qu'il peut fournir.ils sont décrits dans la description de service en termes d'entrées, sorties, pré- conditions et effets du service qui reflètent le fonctionnement du service les moyens de sélection de services actuels reposent sur l'adéquation syntaxique exacte entre l'interface de service offerte par les fournisseurs et l'interface de service requise par l'utilisateur.
cette technique de recherche syntaxique s'est avérée faible.les fonctionnalités offertes sont décrites de manière hétérogène et le résultat peut ne pas correspondre aux attentes des clients, car c'est une recherche basée sur les mots clés. de nombreux travaux prennent par conséquent l'hypothèse que la sélection de service web s'effectue sur une base des besoins fonctionnels.

2.3.2: 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.

Qualité de Service (QoS) La QoS est un ensemble de propriétés et caractéristiques 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 fait une sélection de services parmi plusieurs services web découverts qui visent à satisfaire ses usagers. de nombreux paramètres ont été ainsi proposés dans la littérature. les paramètres de QoS les plus importants sont les suivants :

- **Débit :**le nombre de requêtes servies pendant un intervalle de temps.
- **Temps de réponse :**le temps requis pour compléter une requête du service web.
- **Fiabilité (sûreté) :**la capacité d'un service d'exécuter correctement ses fonctions.
- **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.
- **Robustesse :**la probabilité qu'un service peut réagir proprement à des messages d'entrée invalides, incomplètes ou conflictuelles.
- **Disponibilité :**la probabilité d'accessibilité d'un service.
- **Prix d'exécution :**c'est le prix qu'un client du service doit payer pour bénéficier du service.
- **Réputation :**c'est une mesure de la crédibilité du service.
- **Sécurité :**c'est un regroupement d'un ensemble de qualités à savoir : la confidentialité, le cryptage des messages et le contrôle d'accès.

2.4: Technique de sélection de services web

La sélection de services web basée sur la qualité de service a fait l'objet de plusieurs travaux.de façon générale, on distingue 03 grandes classes comme l'illustrée la figure 2.2.

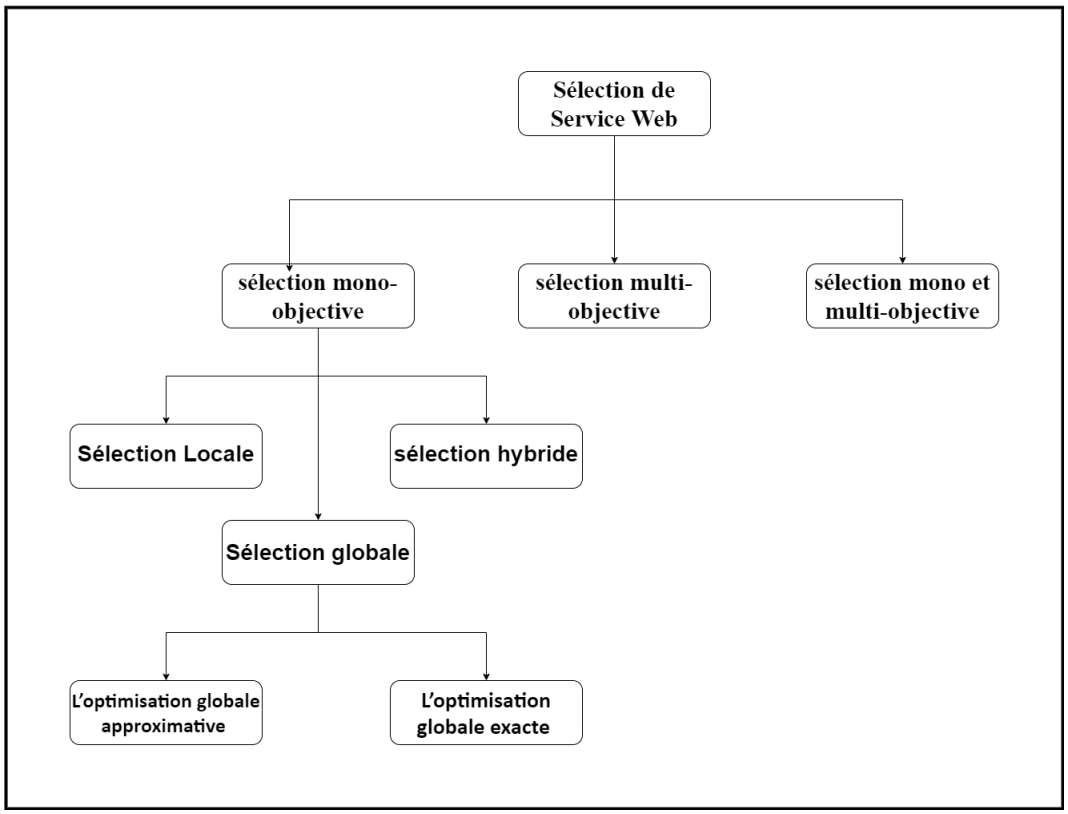

FIGURE 2.2 – Approches de sélection de services web à base de QoS.

2.4.1: Sélection mono-objective

On considère une seule fonction objective qui associe des poids aux différents attributs de QoS (toutes les valeurs de qualité de service soient agrégées en un seul score).elle compose en 03 sous approches (classes).

Sélection locale

Elle a pour objectif de choisir le meilleur service web pour chaque tâche individuelle à part entière.elle vise à considérer seulement les contraintes de QoS relatives à chaque tâche plutôt qu'en considérant les contraintes de QoS globales exprimées pour l'ensemble des tâches.

Sélection globale

Contrairement à la méthode précédente, on choisit la combinaison de services web qui garantit la meilleure qualité globale en tenant compte des contraintes de QoS et des préférences globales assignées pour l'ensemble des tâches.afin de pouvoir appliquer une stratégie de sélection globale, on calcule la valeur des critères de QoS relative au service composite.le calcul de ces critères s'appuie sur l'application de fonctions d'agrégation. ensuite, on choisit la combinaison qui offre les meilleures valeurs de QoS parmi toutes les combinaisons possibles par rapport aux contraintes et aux préférences de l'utilisateur.
On distingue deux sous classes d'optimisation globale : exacte et approximative.

- **L'optimisation globale approximative (méta-heuristiques) :**elle applique des recherches heuristiques ou des méta-heuristiques, qui adoptent une recherche locale ou globale pour donner des résultats proches à l'optimal, tout en ayant un temps d'exécution abordable (polynomial).
- **L'optimisation globale exacte (la programmation par contrainte ou les énumérations exhaustives) :** elle se base sur la programmation entière ou la programmation par contrainte.ces méthodes donnent des résultats optimaux mais elles ont un temps d'exécution exponentiel.

Sélection globale

Cette approche est un compromis des deux précédentes approches, en commençant la recherche par une optimisation globale, puis continuant le travail avec une optimisation locale.l'approche peut également manipuler des contraintes globales.

2.4.2: Sélection multi-objectives

Elle peut être définie comme la recherche de la meilleure ou des meilleures solutions possibles d'un problème donné. souvent, les problèmes d'optimisation sont des problèmes multi-objectifs. si les objectifs sont antagonistes, alors on n'a pas une seule solution optimale mais un ensemble de solutions de compromis.

2.4.3: Sélection mono et multi-objectives

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

2.5: Travaux Connexes

2.5.1: Sélection mono objective

Sélection Globale

Plusieurs travaux ont été proposés pour l'optimisation globale exacte ou approximative.parmi ces travaux nous citons :

— **Optimisation Exacte :**

Dans [Zeng et al , 2003], les travaux de Zeng et ses collègues utilisent les techniques de programmation entière et mixtes ((Mixe d'intègre programme ming ou MIP)pour trouver la composition optimale des services. [Ardagna and Pernici, 2005], étendent le modèle de programmation linéaire afin d'inclure les contraintes locales.dans ce modèle, les contraintes globales sont exprimées sur la composition entière, et par l'utilisateur final.tandis que les contraintes locales peuvent être spécifiées par le concepteur de la composition. [L. Xu, 2011] Proposent une méthode de sélection de services web à base QOS en considérant une variation de valeurs de QOS en fonction de l'intervalle de temps.pour cela l'utilisateur introduit les poids et l'intervalle d'intérêt qui peut englober plusieurs intervalles ayants des valeurs de QOS différentes, mais ils ne gèrent pas les contraintes globales.

— **Optimisation Approximative :**

Plusieurs travaux sont inspirés des algorithmes heuristiques, parmi ces travaux les plus connus sont :

Dans [G. Canfora, 2005] et [J. Wang, 2005], les auteurs présentent un algorithme génétique qui instancie les services abstraits avec des composants concrets. dans les deux travaux, les points de mutation sont choisis aléatoirement .le codage des chromosomes utilise la représentation entière pour [Canfora et al., 2005] et la représentation binaire pour [J. Wang, 2005]. Dans [Zhang and Ren, 2011], les auteurs proposent une normalisation des valeurs de QOS. ils utilisent un algorithme génétique adaptative, i.e. que les taux de mutations et de croisement dépendent de l'affinité des compositions (leur fitness).

Sélection Locale

Dans ce type de sélection locale, le travail le plus populaire est expliqué comme suit : Dans [K.Benouaret, 2011] et [K.Benouaret and Hadjali, 2011], les auteurs considèrent la sélection de service web en prenant en compte les préférences sur les sorties de services. pour ce la, ils appliquent un tri local (pour chaque classe) en adoptant la notion de fuzzy dominance.cette dernière permet la comparaison de deux services de manière plus efficace que la notion de pareto dominance. enfin, ils retiennent les K premières solutions.

Sélection Hybride

Dans ce type de sélection hybride, le travail le plus connu est comme suit :
Dans [M.Alrifai, 2009], les auteurs transforment le problème d'optimisation globale (exacte), en un problème d'optimisation approximative.pour cela, les auteurs divisent chaque intervalle de Qos de chaque classe, en d sous intervalles. Ensuite, ils choisissent un seul sous intervalle pour l'utiliser au niveau de l'étape suivante.pour sélectionner les intervalles, l'approche adopte le modèle MIP [Nemhauser et Wolsey, 1988] (la programmation en nombre entiers ou Mixed IntegerProgramming)

2.5.2: Sélection multi objectives

Dans la sélection Multi-Objective, plusieurs travaux ont été proposés pour la recherche multi-objective des services skylines.la majorité utilise les méta- heuristiques telles que l'algorithme NSGAII (non dominated sorting genetic algorithm II) [K. Deb and Meyarivan, 2002], et SPEAII (Strengh Pareto elitistalgorithm II) [E. Zitizler, 2005] Dans [Claroetal,2005] les auteurs considèrent 04 fonctions objectives :

- minimisation du coût.
- minimisation de la durée.
- maximisation du chiffre d'affaire.
- maximisation de la réputation.

2.5.3: Sélection mono et multi objective

[Alrifai et al, 2010] proposent des approches combinant la sélection multi-objective (skylines) et mono-objective.en particulier ils proposent 03 algorithmes à base de skylines : le premier est nommé « exact skylines », il extrait les services skylines de chaque classe et applique une optimisation globale à base de MIP sur ces résultats. le deuxième algorithme est nommé représentative- skylines, il extrait en premier lieu les services skylines de chaque classe, ensuite il réalise un clustering hiérarchique descendant de chaque catégorie de skyline. l'algorithme de clustering applique un découpage binaire descendant des clusters à l'aide de l'algorithme k-means. chaque cluster possède un service représentant qui a la plus grande affinité (i.e. la fonction fitness).

Le tableau 2.1 présente notre étude comparative entre les différentes travaux existants basée sur un ensemble de critères bien ciblés. les critères considérés dans cette étude sont : Temps de réponse, Qualité de services, Technique utilisé pour la sélection de services web, Préférences d'utilisateur, Scalabilité, La sécurité, Méthode ou modèle utilisé pour représenter les données, Efficacité [16, **?**].

	Technique de sélection de services web								
	Mono objective					Multi objective			Mono et multi objective
Critères	**[L.Zeng, 2004]**	**[Ardagna and Pernici, 2005]**	**[Dréo et al.,2006]**	**[Alrifai et al.,2010]**	**[K. Benouaret and Hadjali, 2011]**	**[K. Deb and Meyarivan, 2002]**	**[E. Zitizler, 2005]**	**[T. Gonsalves, 2009]**	**[Alrifai et al., 2010]**
Temps de réponse	Peu de ces travaux qui ont pris en compte le critère de temps d'exécution					La plupart de travaux ne traitent pas le critère de temps d'exécution			Il prend en considération le critère de temps d'exécution
Qualitéde services	Oui					Oui			Oui
Technique utilisé pour la sélection des services web	- Les techniques de programmation entière et mixtes (Mixed Integer Programming ou MIP) - Programmation linéaire - Recherches heuristiques ou des méta-heuristiques - Un algorithme génétique - Fuzzy dominance					- Les méta heuristiques (l'algorithme NSGAII)			Algorithme Skyline et l'algorithme de clustering
Préférences d'utilisateur	Oui					Oui			Oui
Scalabilité	La plupart de travaux ne traitent pas ce problème					Oui			Oui
La sécurité	Non					Non			Non
Méthode ou modèle utilisé pour représenter les données	Un modèle de graphe					Un modèle de graphe			Un clustering hiérarchique des skylines
Efficacité	Oui					Oui			Oui

TABLE 2.1 – Étude comparative de différentes approches de sélection de services basées sur l'aspect non-fonctionnel.

Discussion

Les sections précédentes ont présenté une revue sur les solutions pour la sélection de services web basée sur la QoS.parmi ces solutions, on a prend en considération cinq travaux dans l'approche mono objective, trois travaux dans l'approche multi objective, et un travail dans l'approche mono et multi objective.le critère de temps d'exécution est absent dans la plupart de ces travaux.le critère du passage à l'échelle concerne la stabilité du fonctionnement lorsque le nombre d'entités augmente. la plupart de ces travaux étudiées traitent ce défi. presque tous les travaux étudiés dans cette étude prennent en considération : la qualité de service et la préférence d'utilisateur.
L'objectif majeur de la plupart des approches est de concevoir une solution pour la sélection de services web basée sur la QoS. plusieurs défis ont été détectés dans le développement des futures architectures et plus d'efforts devraient être pris en compte pour conduire les propositions actuelles vers de meilleurs.
À partir de cette étude comparative de différentes approches de sélection de services web ; nous déduisons que le tempe d'exécution est manquée dans la plupart des solutions proposée. ainsi, le nombre croissant des services web nécessitent une méthode de traitement efficace. par conséquent, ces deux dernières remarques sont menées vers nos contributions qui seront présentées dans le prochain chapitre.

2.6: Conclusion

Ce chapitre a présenté les deux grandes parties suivantes :

- Un aperçu des approches existantes dans la littérature pour la sélection de services web à base de QoS.
- Une étude comparative entre quelques travaux étudiés basée sur un ensemble de critères bien ciblés.

Le prochain chapitre présente nos contributions pour la sélection de services web basée sur la QoS.

Deuxième partie

Contribution

Chapitre 3

CONCEPTION

3.1: Introduction

Dans ce chapitre, nous considérons le problème de la sélection des services web en fonction des préférences de satisfaction du client. cependant, le nombre croissant des services web disponibles posent un grand défi. pour résoudre ce problème, nous proposons dans ce chapitre un Framework MapReduce pour accélérer le calcul et introduire un parallélisme pour traiter de grandes quantités des services web . en outre, une autre proposition dans notre approche est l'utilisation de la logique floue pour évaluer les attributs de QoS.

3.2: Présentation des concepts utilisés pour l'approche proposée

3.2.1: MapReduce

Définition

MapReduce est un paradigme (modèle) de programmation parallèle proposé par Google en 2004. il est principalement utilisé pour le traitement distribué sur de gros volumes de données aux seins d'un cluster de nœuds.il est conçu pour la scalabilité et la tolérance aux pannes.
Ce modèle de programmation fournit un cadre à un développeur afin d'écrire une fonction Map et une fonction Reduce.il permet à traiter un grand ensemble de données de manière massivement parallèle. MapReduce divise une tâche en petites parties et les affecte à plusieurs ordinateurs.plus tard, les résultats sont collectés et intégrés pour former le résultat de l'ensemble de données.un programme MapReduce peut se résumer à deux fonctions Map () et Reduce ()

- La première,**MAP**va transformer les données d'entrée en une série de couples clé/valeur.elle va regrouper les données en les associant à des clés, choisies de telle sorte que les couples clé/valeur aient un sens par rapport au problème à résoudre. Par ailleurs, cette opération doit être parallélisée : on doit pouvoir découper les données d'entrée en plusieurs fragments, et faire exécuter l'opération MAP à chaque machine du cluster sur un fragment distinct. La fonction Map s'écrit de la manière suivante : Map (clé1, valeur1) → List (clé2, valeur2).
- La seconde, **REDUCE**,va appliquer un traitement à toutes les valeurs de chacune des clés distinctes produite par l'opération MAP. au terme de l'opération REDUCE, on aura un résultat pour chacune des clés distinctes. ici, on attribuera à chacune des machines du cluster une des clés uniques produites par MAP, en lui donnant la liste des valeurs associées à la clé. chacune des machines effectuera alors l'opération REDUCE pour cette clé.la fonction Reducer s'écrit de la manière suivante : Reducer (clé2, List (valeur2)) → List (valeur2).

L'exemple classique : c'est le ≪ Word Count ≫ qui permet de compter le nombre d'occurrences d'un mot dans un fichier. en entrée, l'algorithme reçoit un fichier texte qui contient les mots suivants : voiture la le elle de elle la se la maison voiture.
Dans cet exemple, la clé d'entrée correspond au numéro de ligne dans le fichier et tous les mots sont comptabilisés à l'exception du mot ≪ se ≫.le résultat de la fonction Map est donné ci-dessous.

(Voiture, 1) /(la,1) /(le,1) / (elle,1) / (de,1) / (elle,1) / (la,1) / (la,1) / (maison,1) / (voiture,1)

Avant de présenter la fonction Reduce, deux opérations intermédiaires doivent être exécutées pour préparer la valeur de son paramètre d'entrée.la première opération appelée « shuffle » permet de grouper les valeurs dont la clé est commune.la seconde opération appelée « sort » permet de trier par clé. à la différence des fonctions Map et Reduce, shuffle et sort sont des fonctions fournies par le Framework Hadoop, donc, il n'a pas à les implémenter.
Après l'exécution des fonctions shuffle et sort le résultat de l'exemple est le suivant[13] :

(de, [1]) / (elle, [1,1]) / (la, [1, 1,1]) / (le, [1]) / (maison, [1]) / (voiture, [1,1]

Suite à l'appel de la fonction Reduce, le résultat de l'exemple est le suivant :

(de, 1) / (elle, 2) / (la, 3) / (le, 1) / (maison, 1) / (voiture,2)

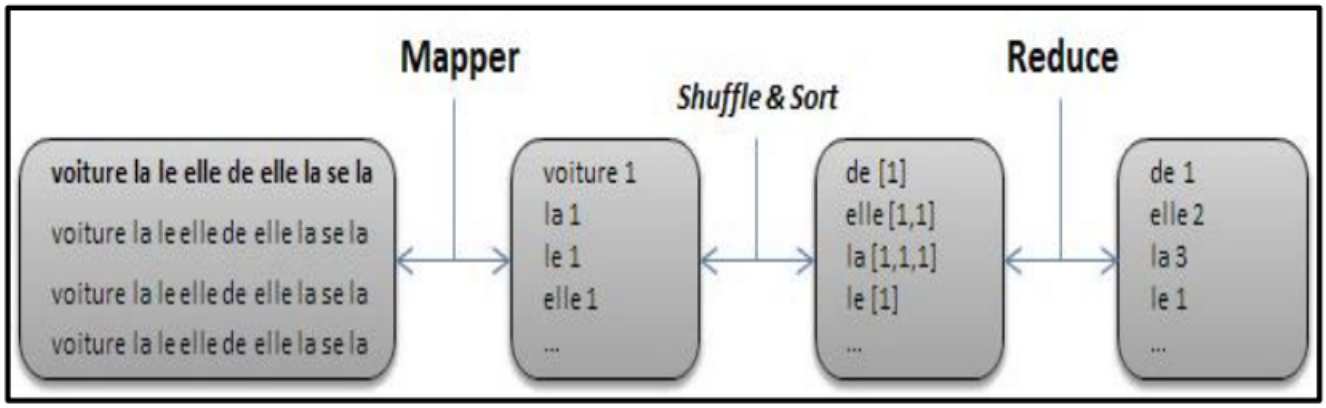

FIGURE 3.1 – Exemple d'un programme MapReduce (WordCount).

Motivation d'utiliser MapReduce

Le nombre croissant des services web qui fournissent la même fonctionnalité mais diffèrent en termes des paramètres de QoS , mène vers un problème de gestion de QoS. la solution pour répondre à ce problème consiste à utiliser un middleware distribué, qui est chargé de sélectionner le meilleur service de manière parallèle et rapide.

3.2.2: Calcul Skyline (basée sur la relation de dominance)

Définition

Börzsönyi et al[5].sont les premiers à avoir abordé la problématique du calcul des Skylines dans le contexte des bases de données.ils ont proposé d'étendre les systèmes de bases de données par une opération Skyline.cette opération filtre un ensemble de points intéressants à partir d'un ensemble potentiellement important de points de données.
Cette méthode aide les utilisateurs à prendre des décisions intelligentes sur des données complexes, où des nombreux critères contradictoires sont considérés. le Skyline est utiles pour extraire des points intéressants à partir d'un grand ensemble de données selon plusieurs critères. un point est considéré comme intéressant, s'il n'y a pas d'autre point mieux que cela dans tous les critères d'évaluation[5].

Relation de dominance

Un concept nécessaire à la définition formelle des Skylines est la relation de dominance :
Définition (Relation de dominance) : On dit qu'un point $p \in P$ domine un autre point $q \in P$, noté $p > q$, si p est strictement préféré ou égal à q sur toutes les dimensions de D et p est préféré à q sur au moins une dimension.
$\forall i \in [1, d] : p_i >= q_i$ et $\exists j \in [1, d] : p_j > q_j$ ....(1)
Donc, un ensemble de points SKY(P) $\subseteq$ P qui ne sont dominés par aucun autre point de P. les points de SKY(P) sont appelés points de Skyline[15].
L'algorithme le plus populaire utilisé pour le calcul skyline est le BNL (Block Nested Loop).cet algorithme vise à comparer tous les points de l'ensemble de données deux à deux et de retourner comme Skyline tous les points qui ne sont dominés par aucun autre .

Algorithme BNL :

Au début, la liste contient le premier point de repères, alors que pour chaque point suivant p, là il existe trois cas :
1- p :dominé par le point dans la liste (p ← upprimer).
2- p :domine n'importe quel point dans la liste (p ← nséré, tous les points dominés par p← upprimer)
3- p : ni dominé, ni domine n'importe quel point dans la liste (p← nséré, sans laisser tomber tout point)

Motivation d'utiliser le calcul Skyline

Cette relation est un concept utile pour extraire des services web intéressants à partir d'un grand ensemble des services selon plusieurs critères de QoS pour satisfaire le besoin du client.

3.2.3: Logique floue

Définition :

Zadeh : La logique floue a été introduite par Lotfi Zadeh en 1965 de l'université de Californie de Berkeley.ce concept vise à fournir des modèles approximatifs pour des systèmes incertains dont la modélisation avec des approches mathématiques précises est très difficile.le système d'inférence floue basée sur la base des règles d'inférence floue peuvent être utilisées pour surmonter l'utilisation des données mathématiques ou des structures algorithmiques complexes dans la modélisation du système.les entrées et les sorties d'un système d'inférence floue sont fournies dans une présentation linguistique et ont des valeurs linguistiques telles que faible, moyen et grand.les règles d'inférence floue sont un ensemble de règles Si-Alors qui peut décider en fonction des valeurs d'entrée formelles et donner des valeurs de sortie.

Motivation d'utiliser la logique floue

Les services web sont connus pour être imparfaites et incertaines par nature.un moyen de limiter cette incertitude est de manipuler des informations supplémentaires associées aux services web ce qu'on appelle la qualité de services (QoS). l'utilisation de la logique floue est motivée par sa méthodologie de résolution de problèmes.elle peut être mise en œuvre sur n'importe quel système de gestion indépendamment de la taille et de la complexité du problème. cette solution est appropriée pour être utilisée dans la représentation d'une description d'attribut non fonctionnelle.il permet de raisonner non pas sur des variables numériques, mais sur des variables linguistiques, c'est-à-dire, sur des variables qualitatives.

3.3: Présentation de l'approche proposée

3.3.1: L'architecture globale

L'architecture de l'approche proposée, qui exprime l'extension de l'architecture de base SOA, en ajoutant une nouvelle composante Découverte entre le client et l'annuaire UDDI de telle façon la phase de la recherche des services web similaires serait donc simple et facile.
Initialement, le fournisseur publie les services web avec leurs QoS dans le registre UDDI qui dispose d'une base de données destinée aux QoS.
A partir de la requête du client en termes d'attributs des QoS, les services web disponibles sont partitionnés par le serveur maître en plusieurs blocs de données. Pour chaque bloc, les services web sont filtrés par le combiner de manière parallèle.ce type de filtrage vise à garder seulement les services web qui sont le plus proche au besoin du client.
Les services web générés par tous les serveurs esclaves sont fusionnés et intégrés par le Reducer pour faire un deuxième filtrage.
Nous appliquons la logique floue sur la liste des services web obtenus par le deuxième filtrage où ces services web sont incomparables.la logique floue permet de sélectionner le meilleur service web pour satisfaire le besoin du client.

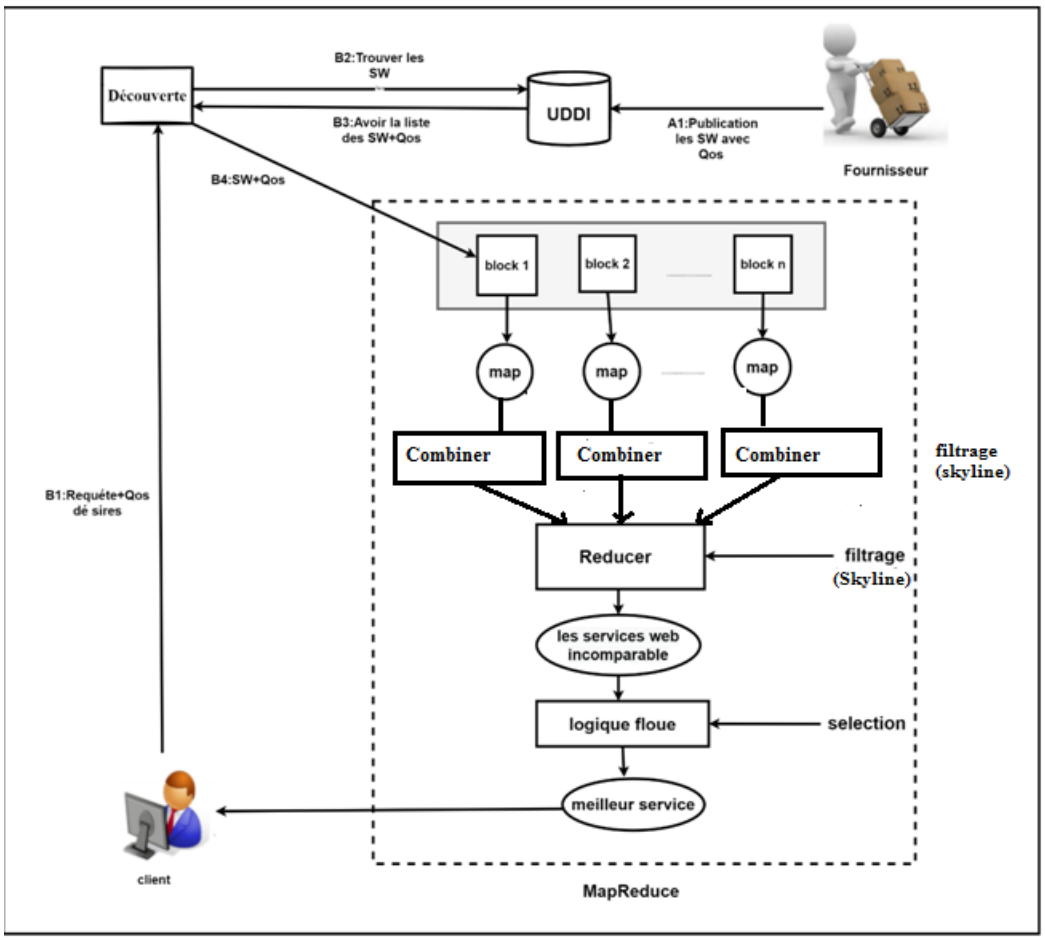

FIGURE 3.2 – Architecture de l'approche proposée.

3.3.2: L'architecture détaillée

Nous allons décrire en détail l'aspect structurel et fonctionnel des différents acteurs posées dans l'architecture de l'approche proposée :

Client : qui souhaite d'avoir un service avec des attributs de QoS désirés en lançant une requête au registre UDDI via le module découverte.

Découverte : ce module s'occupe de la requête émise par le client en l'envoyant vers le registre UDDI en récupérant les services web avec leurs qualités.
Les tâches effectuées par ce module sont :

1. Obtenir la requête de l'utilisateur.
2. Trouver la liste des services web disponibles avec leurs propriétés non fonctionnelles (QoS) à partir du registre UDDI.
3. Passer cette liste au composant de MapReduce.

Registre UDDI : il a pour but de permettre d'automatiser les communications entre fournisseurs et clients et de gérer les métadonnées des services. UDDI est une spécification qui définit les mécanismes qui permettent aux entreprises de publier leurs services et de découvrir les détails techniques d'un service web (WSDL). Ainsi, lorsque l'on veut mettre à disposition un nouveau service, on crée un fichier appelé Business registry qui décrit le service en utilisant un langage dérivé d'XML suivant les spécifications UDDI.
Les opérations pouvant être effectuées par UDDI sont :

- **Recherche :**qui se fait par le nom et/ou par des mots clés de service désiré.
- L'ajout et la suppression de services.

Fournisseur : le Fournisseur du service correspond à la personne ou à l'organisation propriétaire du service qui réalisera effectivement le service demandé.il publie son service en fournissant la description des services (les propriétés fonctionnel et non fonctionnel) au format WSDL dans l'annuaire de services afin que les clients puissent découvrir et accéder au service.

MapReduce

Map : C'est la première étape de la programmation MapReduce.

- Tout d'abord, la fonction Map prend une entrée un block de données et produit un ensemble de paires (clé / valeur), où la clé représente le numéro de service et la valeur représente les valeurs des attributs QoS.
  (Clé/valeur) $\Rightarrow$ (num service/les valeurs de QoS)

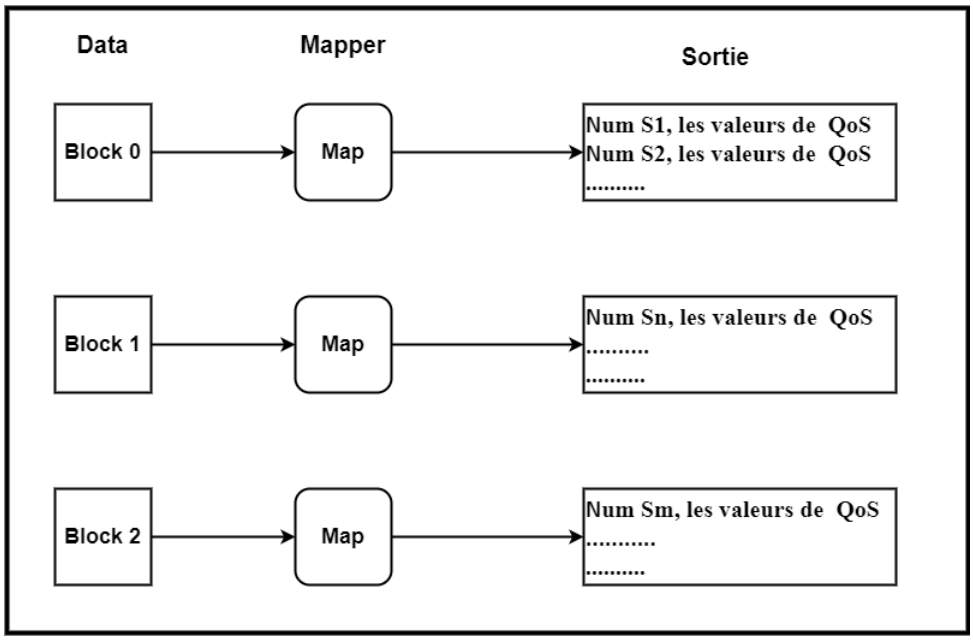

FIGURE 3.3 – Fonctionnement de Mapper.

Combiner : il s'agit d'une étape facultative dans le modèle MapReduce. ils sont utilisés pour améliorer les performances des fonctions MapReduce.

- Les entrées de l'étape de combiner sont les sorties de la fonction Map.
- A ce stade, le processus de filtrage est appliqué de manière parallèle à chaque block parmi tous les blocks résultants. Le processus de filtrage est effectué à l'aide d'une opération de Skyline basée sur la relation de dominance.cette opération vise à comparer tous les services au niveau de chaque block deux à deux et de retourner comme Skyline tous les services qui ne sont dominés par aucun autre.cette comparaison est en fonction des valeurs de qualité de service selon la demande de client.

  Dans ce cas, on récupère seulement les services web si elle est supérieure ou égale à tous les services web dans tous les critères QoS.et éliminer les services web dominées par d'autres dans chaque block.
- Les sorties de cette étape sont des services web qui sont incomparables au niveau de chaque block. nous considérons cette étape comme un processus de mini-reduce.

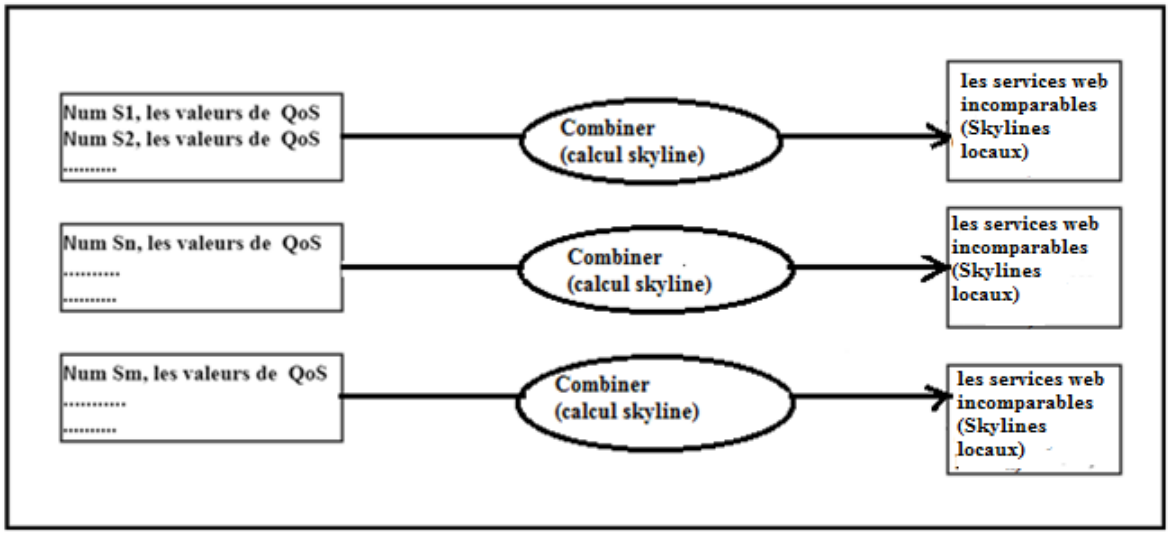

FIGURE 3.4 – Fonctionnement de Combiner.

Exemple

Dans cet exemple, on suppose qu'un utilisateur s'intéresse au service web de haute fiabilité et efficacité, et un temps de réponse et un cout de service le plus petit possible.
Supposons que le Tableau 3.1 montre les valeurs de critères de QoS de quatre services web.

	Fiabilité	Efficacité	Prix	Temps de réponse
SW 1	0.3	0.1	0.8	0.6
SW 2	0.4	0.7	0.5	0.9
SW 3	0.9	0.2	0.5	0.4
SW 4	0.1	0.5	0.7	0.3

TABLE 3.1 – Montre les valeurs de critères de QoS de quatre services web

Pour comprendre la méthode de filtrage selon le calcul skyline basée sur la relation de dominance, on obtient le résultat indiqué dans la table 3.2 :

— service web SW2 domine le service web SW1, donc le service SW1 est éliminé .
— service web SW2 est dominé par le SW3, donc le SW2 est éliminé.
— service web SW3 et SW4 sont incomparables.

Donc, SW3 et SW4 sont les meilleurs compromis possibles entre fiabilité, efficacité, prix, et temps de réponse.

	Fiabilité	Efficacité	Prix	Temps de réponse
SW 3	0.9	0.2	0.5	0.4
SW 4	0.1	0.5	0.7	0.3

TABLE 3.2 – Services web incomparables

Reducer :

— entrées de l'étape de Reducer sont les sorties du processus de combiner.

— à ce stade, la fonction Reduce fusionne tous les blocks filtrés (skylines locaux) qui sont obtenus par le combiner pour faire un filtrage final entre eux.

— sortie, on obtient un ensemble de services web final (skylines globaux) qui sont incomparables et proche à la demande d'utilisateur.

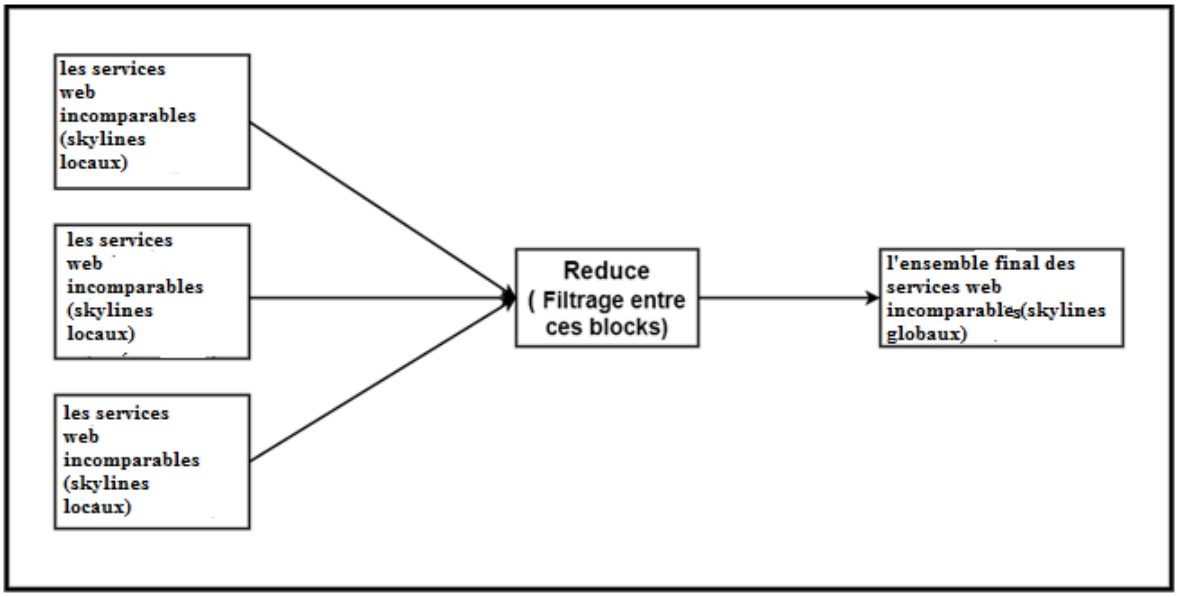

FIGURE 3.5 – Fonctionnement de Reducer.

Logique floue :

Nous appliquons la logique floue sur l'ensemble de services web générée dans le processus de MapReduce.la logique floue transforme les valeurs de critères de QoS de chaque service web en valeurs floues à l'aide des fonctions d'appartenance.le résultat de ces fonctions d'appartenance permet aux règles d'inférence de dériver les valeurs d'utilité de chaque service web.
Le service web qui a le poids le plus élevé est le candidat sélectionné. Le déroulement de cette méthode est décrit dans la Figure 3.6.

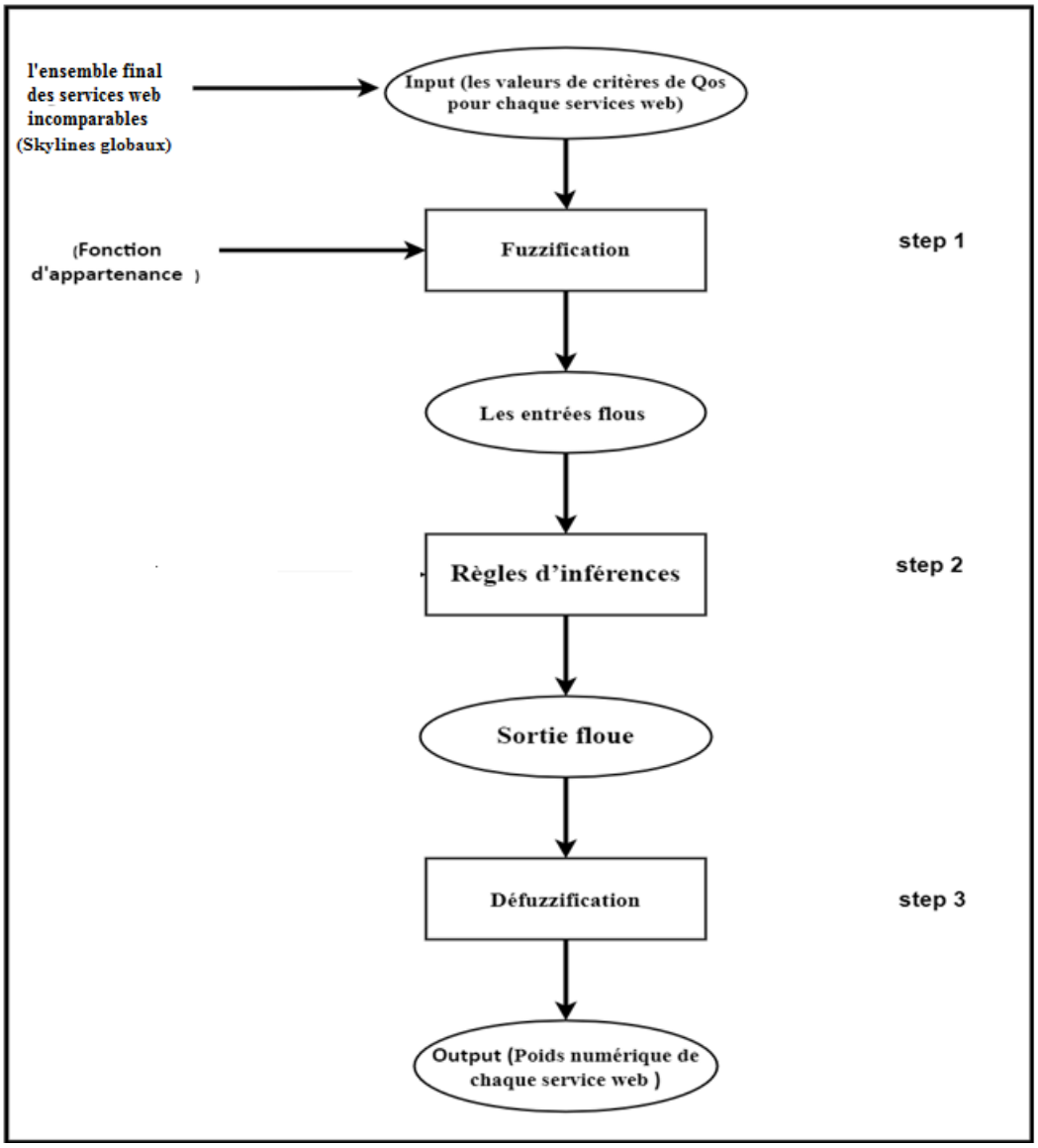

FIGURE 3.6 – Fonctionnement de la logique floue.

Exemple

Considérons dans cet exemple le service web (SW1) avec deux dimensions (fiabilité, temps de réponse) représenté sur la table 3.3

Service web 1 (SW1)	
Fiabilité	0.3
Temps de réponse	0.6

TABLE 3.3 – Service web (SW1) avec deux dimensions

Dans cet exemple : les valeurs de critères de Qos sont fuzzifiées en utilisant trois fonctions d'appartenance triangulaires : FH (x), FM (x) et FL (x). ces fonctions d'appartenance sont présentées dans les équations (2), (3) et (4), respectivement.ils évaluent les valeurs « High, Medium, Low» pour les critères de QoS, respectivement.

$$F_H(x) = \begin{cases} \frac{x-a}{b-a} & \text{if } a \le x \le b \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

$$F_M(x) = \begin{cases} \frac{x-b}{c-b} & \text{if } b \le x \le c \\ \frac{d-x}{d-c} & \text{if } c < x \le d \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

$$F_L(x) = \begin{cases} \frac{x-a}{b-a} & \text{if } a \le x \le b \\ \frac{c-x}{c-b} & \text{if } b < x \le c \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

Où les constantes : a, b, c et d sont fournies par un expert en fonction des données.cette fonction est simple et prend les valeurs d'utilisateur, puis il transfère directement à l'ensemble flou.pour les systèmes qui nécessitent une variation dynamique significative sur une courte période de temps, la fonction triangulaire devrait être utilisée. Si on met : a = 0 ; b = 0,1 ; c = 0,6 ; d = 1 nous obtenons les valeurs floues suivantes (décrites dans le tableau 4.3) :

Critères de QoS	Fiabilité	Temps de réponse
Les valeurs floues pour les critères de QoS	0% high 40%medium 60 % Low	0% high 100 %medium 0 % Low

TABLE 3.4 – Valeurs floues

Règles d'inférence :

If fiabilité 0 % high and temps_ réponse 0 % high then (weight SW1) 0% high
If fiabilité 0%high and temps _réponse 100 % medium then (weight SW1) 0% high
If fiabilité 0%high and temps _réponse 0 % Low then (weight SW1) 0% Low
If fiabilité 40 %medium and temps _réponse 0% high then (weight SW1) 0%high
If fiabilité 40% medium and temps _réponse 100% medium then (weight SW1) 40 % medium
If fiabilité 40%medium and temps _réponse 0% Low then (weight SW1) 0% Low
If fiabilité 60% Low and temps_réponse 0% high then (weight SW1) 0% high
If fiabilité 60% Low and temps _réponse 100% medium then (weight SW1) 60% Low
If fiabilité 60% Low and temps _réponse 0% Low then (weight SW1) 0% Low

Nous avons basé dans nos règles d'inférence sur l'opérateur AND qui correspond à l'opérateur MIN.dans nos règles d'inférence, nous avons plusieurs règles qui génèrent plusieurs valeurs de la même variable linguistique, nous avons choisi un opérateur OU pour combiner les valeurs de la même variable.
Par exemple, nous avons quatre règles qui génèrent la variable linguistique «low» : weight-SW1 est low à 0% et 60%. Si on utilise un opérateur OU (opérateur Maximum), la variable " weight- SW1 est low" aura une valeur finale de 60%. par conséquent, nous obtenons : weight- SW1 est high à 0%, medium à 40 % et low à 60%.

Déffuzzification :

Après avoir combiné les règles, il faut maintenant produire un chiffre net qui représente le poids d'un service.
La technique la plus populaire est la méthode de centre de gravité,elle est exprimée comme :

$$x = \frac{\sum_{i=1}^{n} m^1 w^{i\,b}}{\sum_{i=1}^{n} m^i} \qquad (5)$$

Où :

— X = sortie défuzzifiée.
— mi = valeur d'appartenance de la sortie.
— wi = valeurs possibles de la sortie.

Cette méthode est utilisée car elle est plus rapide, plus facile et donne un résultat assez précis[9].
En appliquant la relation (5), on trouve :

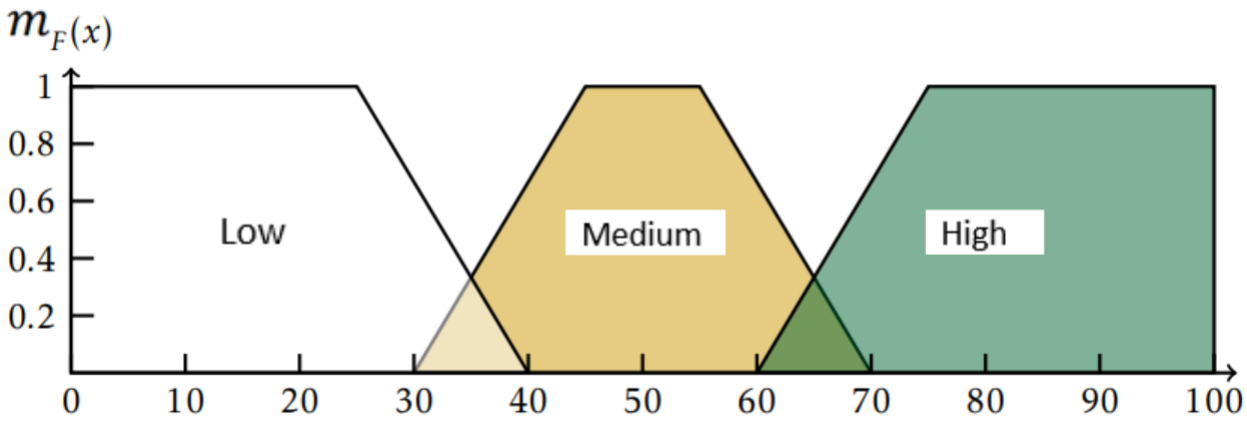

FIGURE 3.7 – Ensembles flous pour le poids de service.

$$x = \frac{(0+10+20+30)\,0.6 + (40+50+60)\,0.4}{0.6+0.6+0.6+0.6+0.4+0.4+0.4} = 26$$

Donc, le poids de ce service web est 26%.

3.4: Modélisation

3.4.1: Digramme de séquence de publication un service web :

Au début, le fournisseur doit publier les services web disponibles avec leurs critères de QoS au niveau de l'annuaire UDDI.

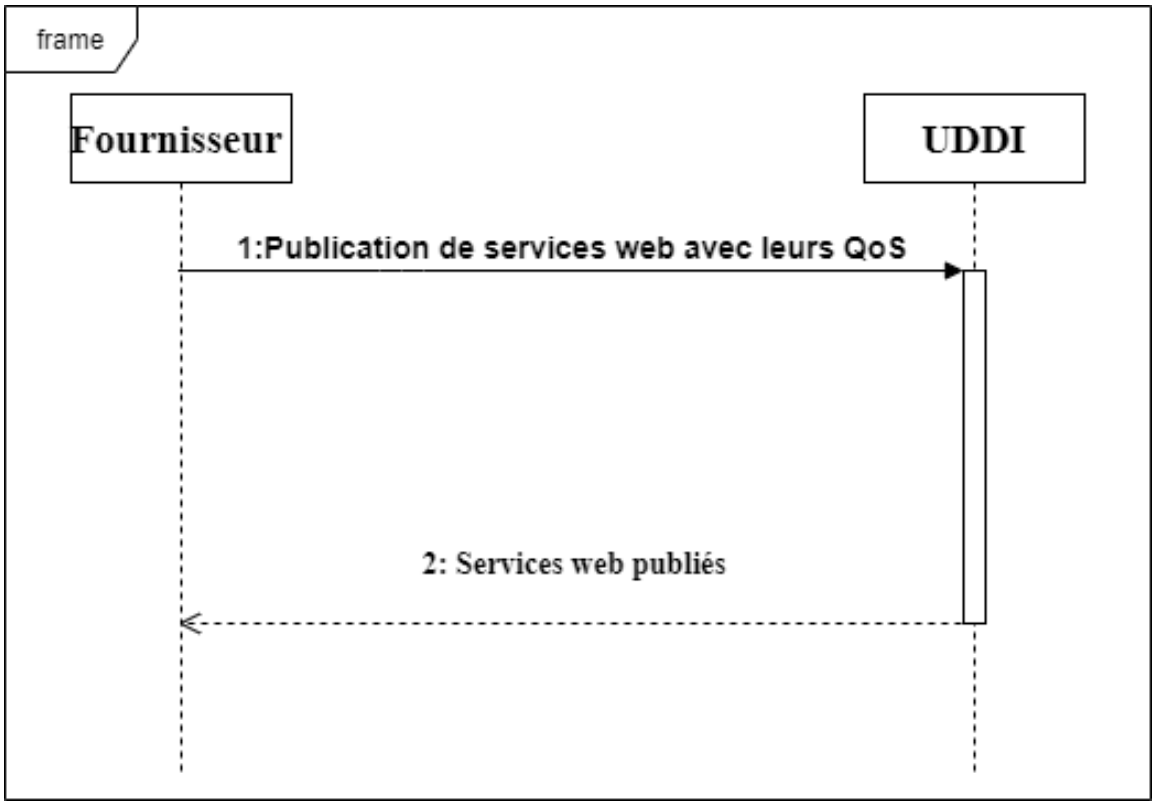

FIGURE 3.8 – Diagramme de séquence de publication des services.

3.4.2: Digramme de séquence de sélection un service web :

Une fois que nous disposons d'un registre UDDI dont lequel il y a eu des services web publiés, le client pourra ainsi lancer sa requête de recherche de services web. cette requête est d'abord envoyée au module découverte qui joue un rôle intermédiaire entre le client et le registre UDDI.après la recherche dans ce dernier, il y a eu une liste des sw similaires qui est retournée à ce module qui fait l'extraction des différents critères de QoS et envoi ensuite cette liste (Service web + Critères) au module mapreduce.ce dernier effectue un processus de filtrage en deux étapes puis il applique la logique floue pour choisir le meilleur service.
L'ensemble des interactions entre les différents acteurs pour aboutir à la sélection d'un meilleur service peut être décrit par le diagramme de séquence suivant :

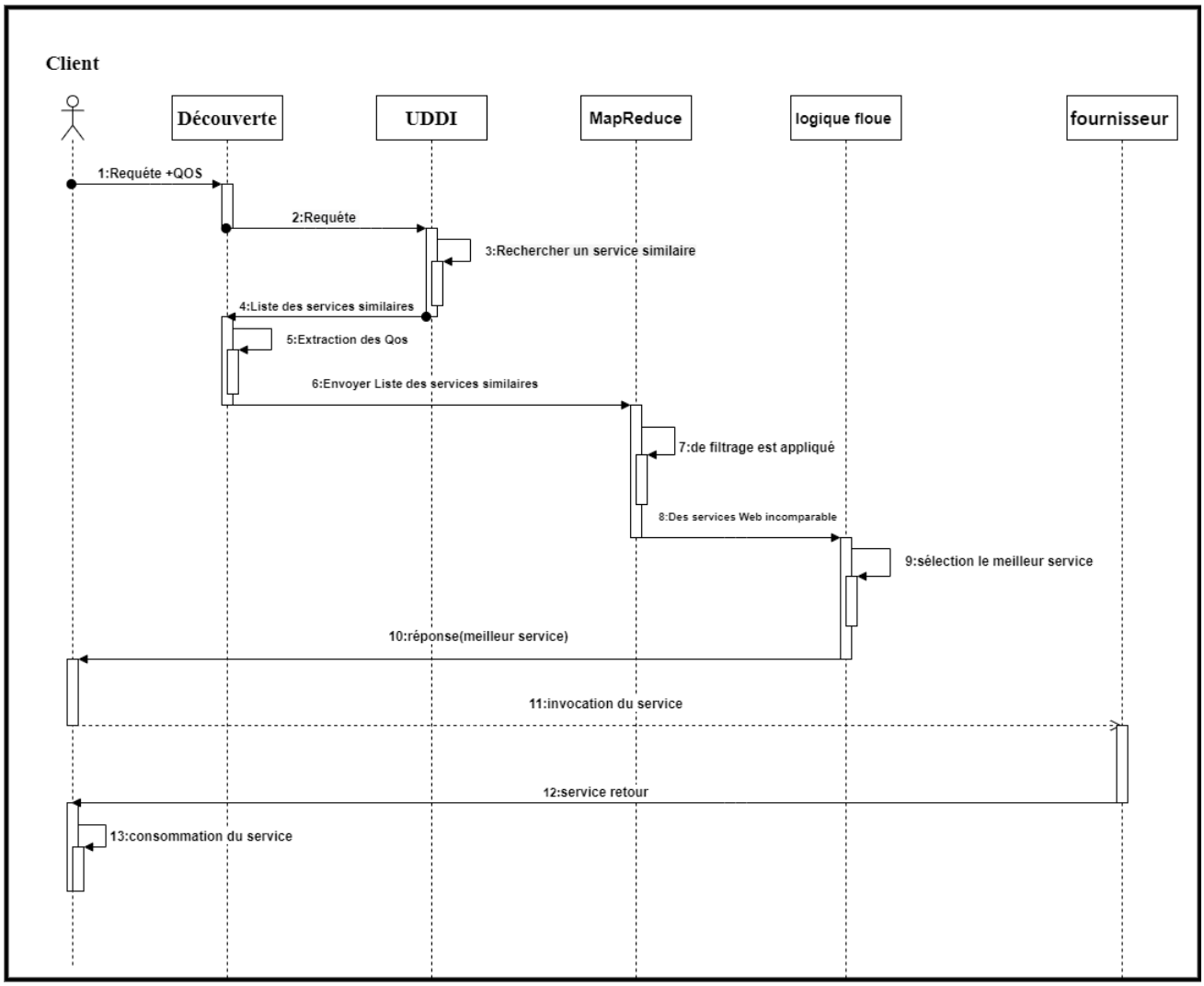

FIGURE 3.9 – Diagramme de séquence de sélection de services web.

3.5: Algorithmes proposés de MapReduce et la logique floue

3.5.1: Algorithme de MapReduce

Algorithme 1 : Mapreduce+ Skyline

Entrée : la liste des N blocks des services web ;

Sortie : La liste des Skylines Globaux (GSkylines);

Début Algorithme 1

1. Pour chaque block i (1..N) **faire**

2. Début pour

3. NewBlock i←Map (block i); **// Mapper**

4. Fin pour

5. Pour chaque Newblock i (1..N) **faire**

6. Debut pour

7. LSkyBlock i←BNL (Newblock i); **// Combiner →obtenir les skylines locaux pour chaque block**

8. Fin pour

9. GSkylines← BNL (LSkyBlock i, …. LSkyBlock N) **//Reducer**

10. Afficher(GSkylines) **// la liste des Skylines globaux (services web incomparables)**

Fin Algorithme 1

3.5.2: Algorithme la logique floue

Algorithme 2: la logique floue

Entrée : Des services Web incomparables Skylines globaux (GSkylines);

Sortie : Poids numérique de service web (PNSW);

Début

1. Pour chaque SW ∈ GSkylines **faire**

// Fuzzification (Entrée: les valeurs de QoS (QS), Sortie: les valeurs de QoS en terme linguistique (QS_linguistique));

2. QS_linguistique←Fuzzification (QS);

// Les règles d'inférence (Entrée : les valeurs de QoS en terme linguistique (QS_linguistique), Sortie : poids linguistique de service web(PLSW))

3. PLSW←Inference rules (QS_linguistique);

//Deffuzification

4. PNSW← Deffuzification(PLSW):

5. Fin pour

Fin Algorithme 2// Le service web qui possède le poids numérique le plus élevé est le candidat sélectionné.

3.6: Conclusion

Dans ce chapitre, on a proposé une approche à base MapReduce et la logique floue pour le problème de sélection de service web à partir de la requête de client. nous avons décrit l'architecture générale et détaillée de notre solution avec quelques exemples illustratifs. dans le chapitre suivant, nous allons présenter les résultats de la validation de l'approche proposée, afin d'évaluer leur efficacité.

Chapitre 4

Implémentation et analyses

4.1: Introduction

L'objectif de ce chapitre est la présentation de l'aspect implémentation de notre application.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 présentons 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 utilisé un PC I3 doté de Windows 10 (64bits) qui est décrit dans la figure suivante :

FIGURE 4.1 – Environnement logiciel utilisé.

4.3: Langage Python

Le langage de programmation Python a été créé en 1989 par Guido van Rossum, aux Pays-Bas.le nom Python vient d'un hommage à la série télévisée Monty Python's Flying Circus dont G. van Rossum est fan. la première version publique de ce langage a été publiée en 1991.
La dernière version de Python est la version 3. Plus précisément, la version 3.7 a été publiée en juin 2018.la Python Software Foundation 1 est l'association qui organise le développement de Python et anime la communauté de développeurs et d'utilisateurs.
Ce langage de programmation présente de nombreuses caractéristiques intéressantes :

- **Il est multiplateforme. C'est-à-dire qu'il fonctionne sur de nombreux systèmes d'exploitation :** Windows, Mac OS X, Linux, Android, iOS, depuis les mini-ordinateurs Raspberry Pi jusqu'aux supercalculateurs.
- **Il est gratuit :**vous pouvez l'installer sur autant d'ordinateurs que vous voulez (même sur votre téléphone !).
- C'est un langage de haut niveau.il demande relativement peu de connaissance sur le fonctionnement d'un ordinateur pour être utilisé.

— C'est un langage interprété.un script Python n'a pas besoin d'être compilé pour être exécuté, contrairement à des langages comme le C ou le C++[8].

4.4: Hadoop

(High-availability distributed object-oriented platform) est un système distribué permettant de stocker et d'analyser des données.le grand intérêt d'Hadoop est de proposer un Framework pour effectuer des analyses de données de façon parallélisée sur plusieurs machines.d'autre part, Hadoop permet d'utiliser des machines normales et de les associer en groupe de façon à ce que les puissances de calcul soient additionnées.ces groupes de machines sont appelés des clusters et chaque machine est appelée un nœud (ou node). en effet, pour effectuer de gros traitements il n'est pas nécessaire d'utiliser de gros serveurs de type "mainframe", plusieurs PC individuels de puissance classique suffisent. FIGURE 4.2 – Spectacles Pile logicielle simplifiée de l'écosystème Hadoop.

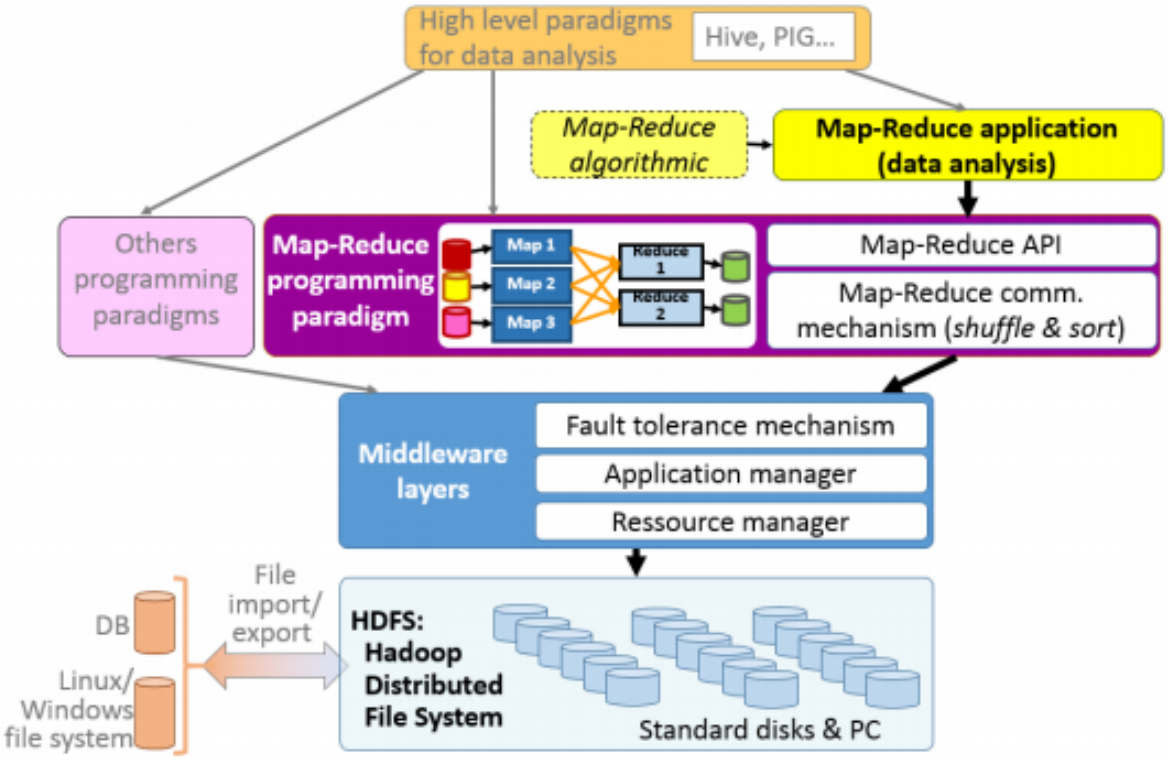

FIGURE 4.2 – Pile logicielle simplifiée de l'écosystème Hadoop.

4.4.1: Avantages principaux d'Hadoop sont

— permet une grande scalabilité pour stocker et traiter de grandes quantités de données.
— est tolérant aux pannes.
— est optimisé pour un type varié de données (comme le texte ou les images).

— possède un écosystème riche et souvent gratuit composé de différents outils capables de résoudre une problématique précise.

4.4.2: Socle technique d'Hadoop

Le socle technique d'Hadoop est composé de :
Toute l'architecture support nécessaire pour l'implémentation de **MapReduce**, c'est-à-dire :

— l'ordonnancement des traitements.
— localisation des fichiers.
— distribution de l'exécution.

D'un système de fichiers **HDFS** qui est :

— **Distribué :** les données sont réparties sur les machines du cluster.
— **Répliqué :**en cas de panne, aucune donnée n'est perdue.
— **Optimisé :** pour collecter les données et les traitements.

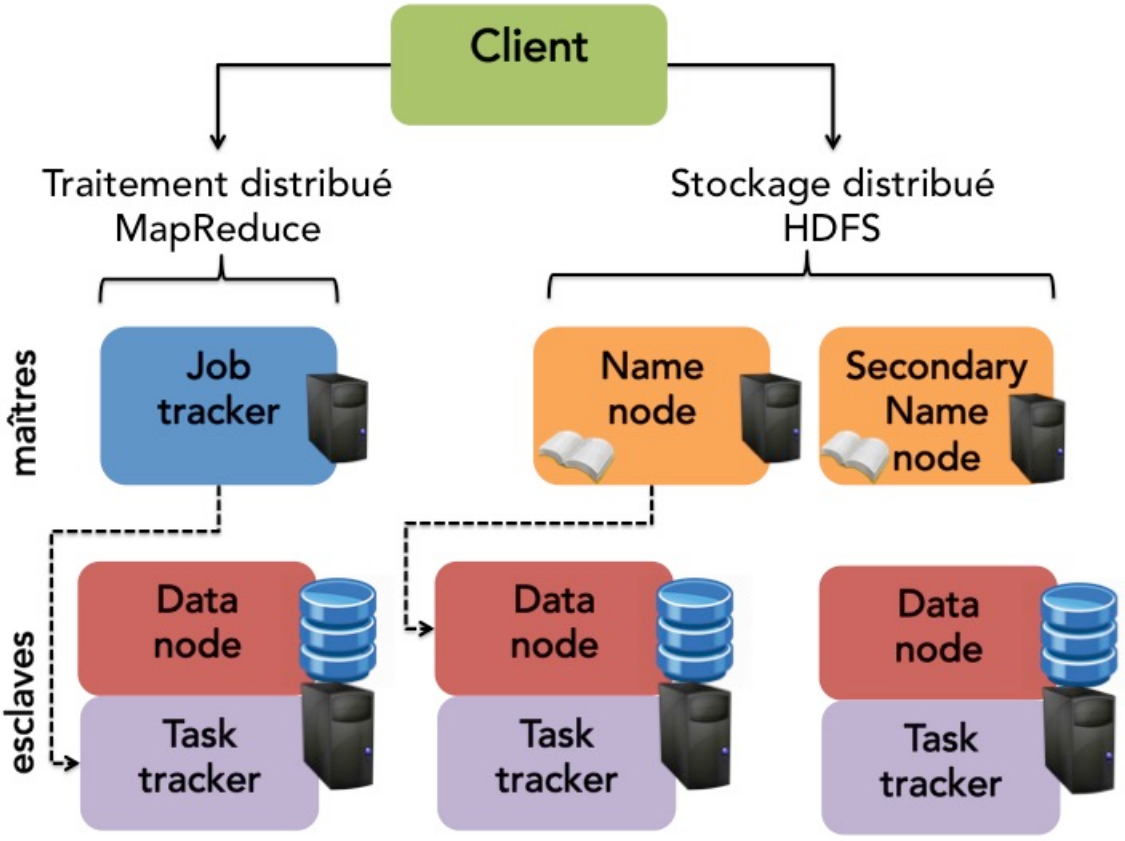

FIGURE 4.3 – Socle technique d'Hadoop .

Job tracker : Le job tracker est un nœud maître qui va se charger de l'ordonnancement des traitements et de la gestion de l'ensemble des ressources du système.il reçoit (du client) la ou les tâches MapReduce à exécuter (un .jar Java) ainsi que les données d'entrée et le répertoire où stocker les données de sorties. Il est pour cela en communication avec le name node d'HDFS.le job tracker est en charge de planifier l'exécution des tâches et de les distribuer sur des task trackers.
Task tracker : Un task tracker est une unité de calcul du cluster. il assure, en lançant une nouvelle machine virtuelle java (JVM), l'exécution et le suivi des tâches MAP ou REDUCE s'exécutant sur son nœud qu'il reçoit du job tracker.il dispose d'un nombre limité de slots d'exécution et donc un nombre limité de tâches MAP, REDUCE ou SHUFFLE pouvant s'éxécuter simultanément sur le nœud. il est aussi en communication constante avec le job tracker pour l'informer de l'état d'avancement des tâches (heartbeat call).
Name node : Le nœud maître appelé name node permet de stocker tous les noms et blocs des fichiers ainsi que leur localisation dans le cluster. on peut donc le voir comme un gros annuaire.
Data node :sont les data nodes qui ont pour rôle la gestion des opérations de stockage locales (création, suppression et réplication de blocs) sur instruction du name node.

4.5: Docker

Docker est un projet open source (Apache 2.0) écrit en GO (un langage de programmation orientée service) et hébergé sur GitHub : https ://github.com/docker (https ://github.com/docker). Initialement porté par la startup DotCloud (renommée depuis Docker) fondée par deux français anciens de l'Epitech.
Docker est composé de trois éléments :

- le daemon Docker qui s'exécute en arrière-plan et qui s'occupe de gérer les conteneurs (Container avec runC).
- une API de type REST qui permet de communiquer avec le daemon.
- le client en CLI (command line interface) : commande docker Par défaut, le client communique avec le daemon Docker via un socket Unix (/var/run/docker.sock) mais il est possible d'utiliser un socket TCP.

  Docker c'est aussi un dépôt d'images (aussi appelé registry) :
  https ://store.docker.com (https ://store.docker.com) Il contient les images officielles maintenues par Docker mais aussi celles mises à disposition par d'autres contributeurs.

FIGURE 4.4 – Docker.

4.6: Visual Studio

Visual Studio Code est un éditeur de code extensible développé par Microsoft pour Windows, Linux et macOS2.
Les fonctionnalités incluent la prise en charge du débogage, la mise en évidence de la syntaxe, la complétion intelligente du code, les snippets, la ré factorisation du code et git intégré.les utilisateurs peuvent modifier le thème, les raccourcis clavier, les préférences et installer des extensions qui ajoutent des fonctionnalités supplémentaires.
Le code source de Visual Studio Code provient du projet logiciel libre et open source VSCode de Microsoft publié sous la licence MIT permissive, mais les binaires compilés sont des logiciels gratuits pour toute utilisation.

4.7: Gestion de Base de Données (phpMyAdmin) :

4.7.1: XAMPP :

Un ensemble de logiciels permettant de mettre en place facilement un serves web et un serveur FTP.il s'agit d'une distribution de logiciels libres (Apache MySQL Perl PHP) offrant une bonne souplesse d'utilisation, réputée pour son installation simple et rapide. Ainsi, il est à la portée d'un grand nombre de personnes, puisqu'il ne requiert pas de connaissances particulières et fonctionne de plus sur les systèmes d'exploitation les plus répandus.

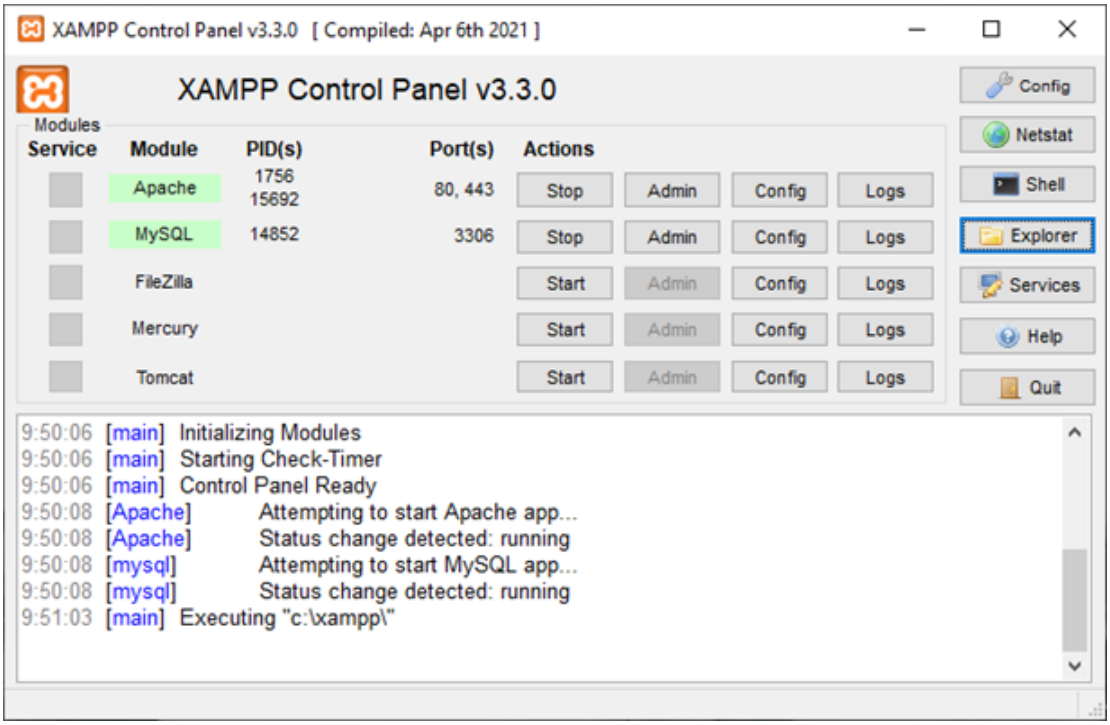

FIGURE 4.5 – XAMPP.

4.7.2: MySQL :

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.

4.7.3: PhpMyAdmin (anciennement MySQL administration) :

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.

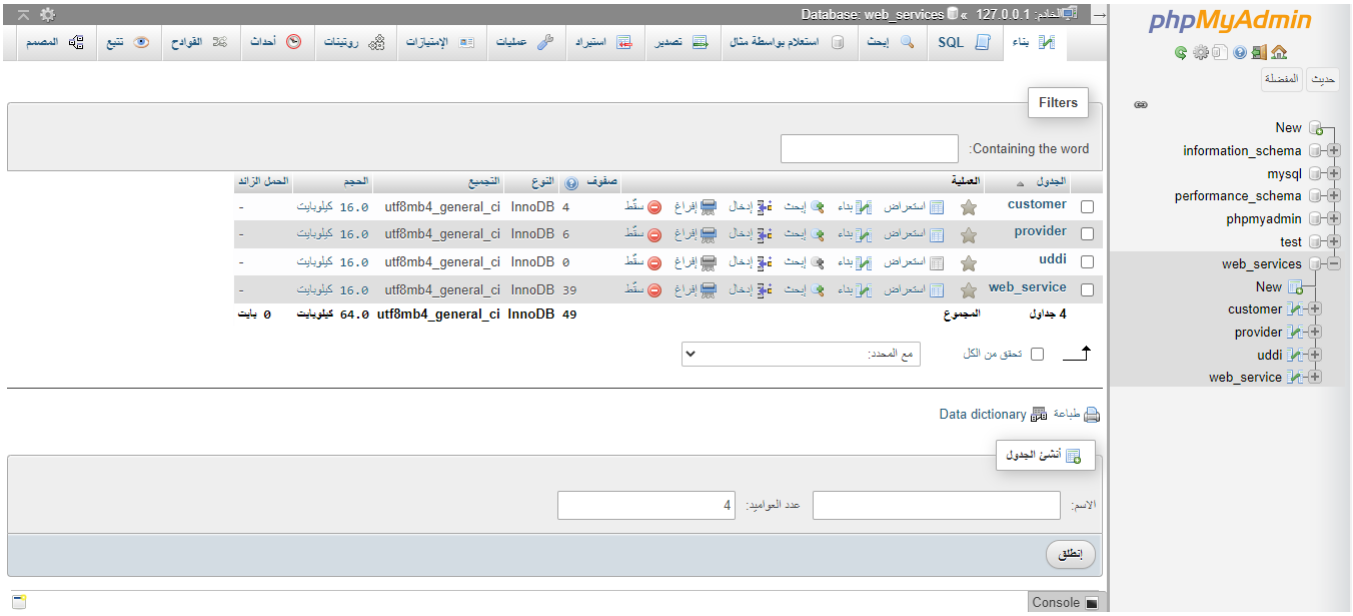

FIGURE 4.6 – Interface Serveur phpMyAdmin.

4.7.4: Base de données :

Pour implémenter notre prototype, nous avons créé une base de données de tables :

— Client

— Fournisseur

— Service-web

— UDDI

La principale table est la table "service-web". ele contient les services web et leurs qualités de services.

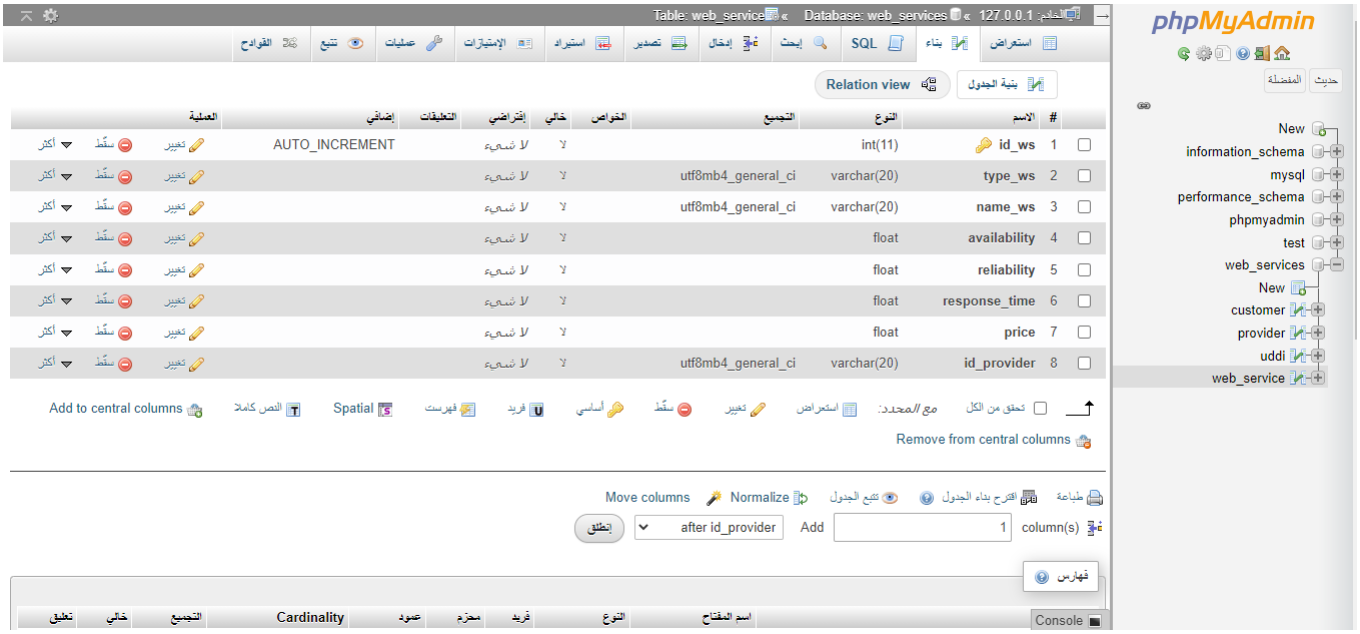

FIGURE 4.7 – Table "Service-web".

Analyse de scénario

Nous illustrons notre travail en utilisant un scénario très simple. les utilisateurs exploitent généralement des services web pour connaître des informations nécessaires (information X par exemple : information de météo ou d'autre type d'information). en revanche, plusieurs services web qui fournissent l'information X sont disponibles. l'objectif de cet scénario est de fournir aux utilisateurs les informations X nécessaire selon leurs besoins en termes de variables linguistiques . dans notre simulation, la requête de l'utilisateur est de rechercher un service web qui fournit l'information X désiré. ce service devrait avoir une grande disponibilité et fiabilité, et un temps de réponse et un cout de service le plus petit possible.

4.8: Présentation des Interfaces Graphiques

4.8.1: Interface d'accueil

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

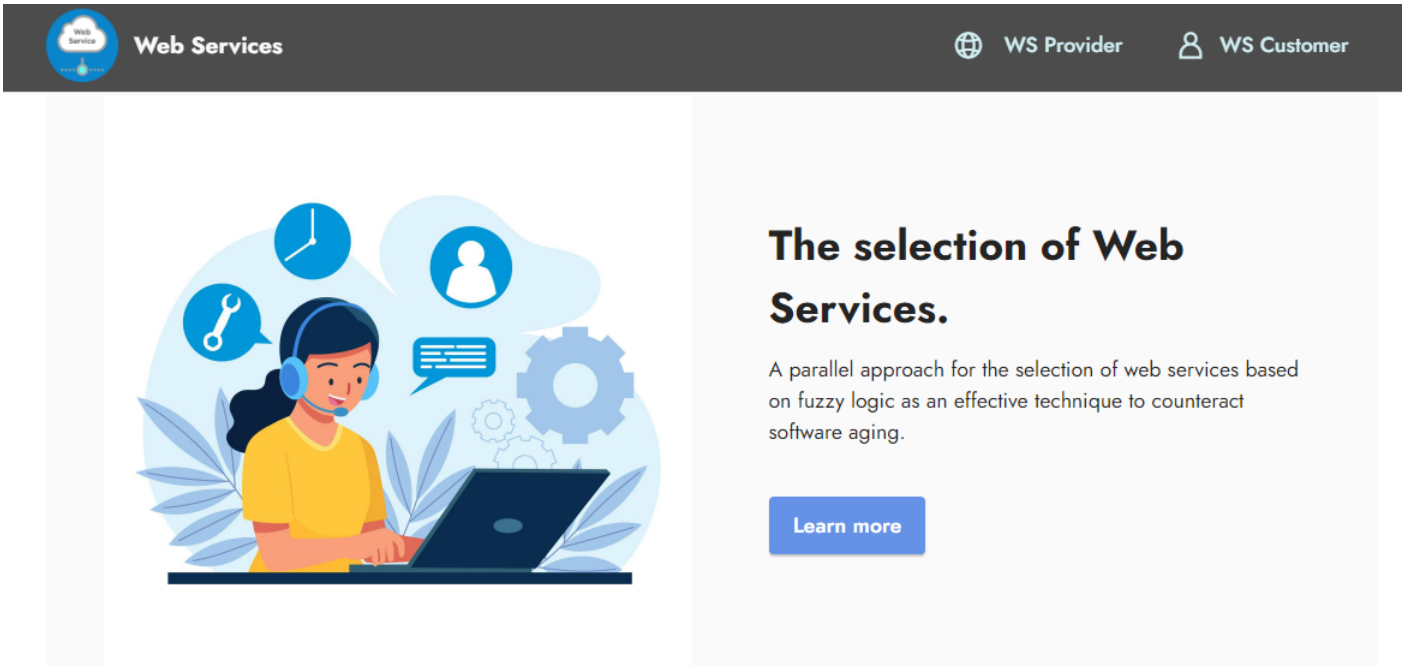

FIGURE 4.8 – Interface principale de l'application.

Interfaces principales

Notre système contient plusieurs interfaces qui traitent les différents cas d'utilisation des services web comme suit :

4.8.2: Interface d'Inscription du Fournisseur

L'utilisateur doit préciser son statut avant de s'inscrire. nous montrons l'interface de formulaire d'inscription du fournisseur comme le montre la figure 4.9. dans la partie formulaire d'inscription de fournisseur, on trouve une partie qui concerne les informations personnelles du fournisseur.

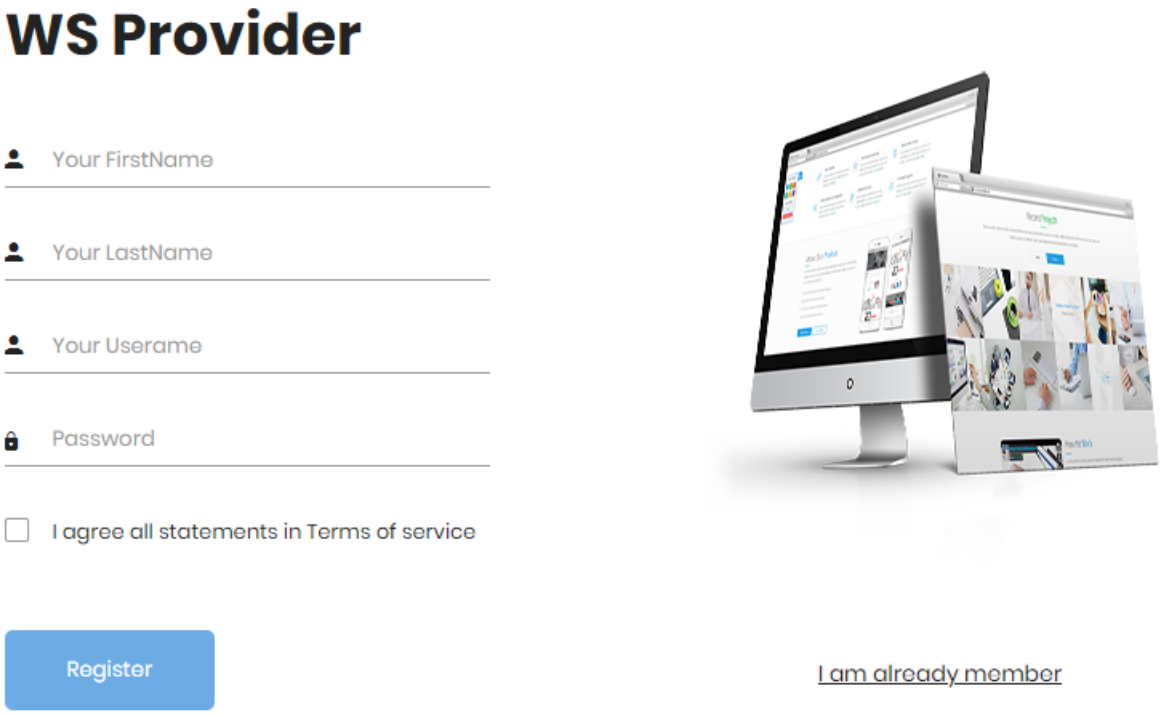

FIGURE 4.9 – Interface d'inscription du fournisseur.

4.8.3: Interface Login du fournisseur

Pour ajouter un nouveau service web à notre base de données (table UDDI) par un fournisseur de service, il faut d'abord authentifier l'identité de ce fournisseur.

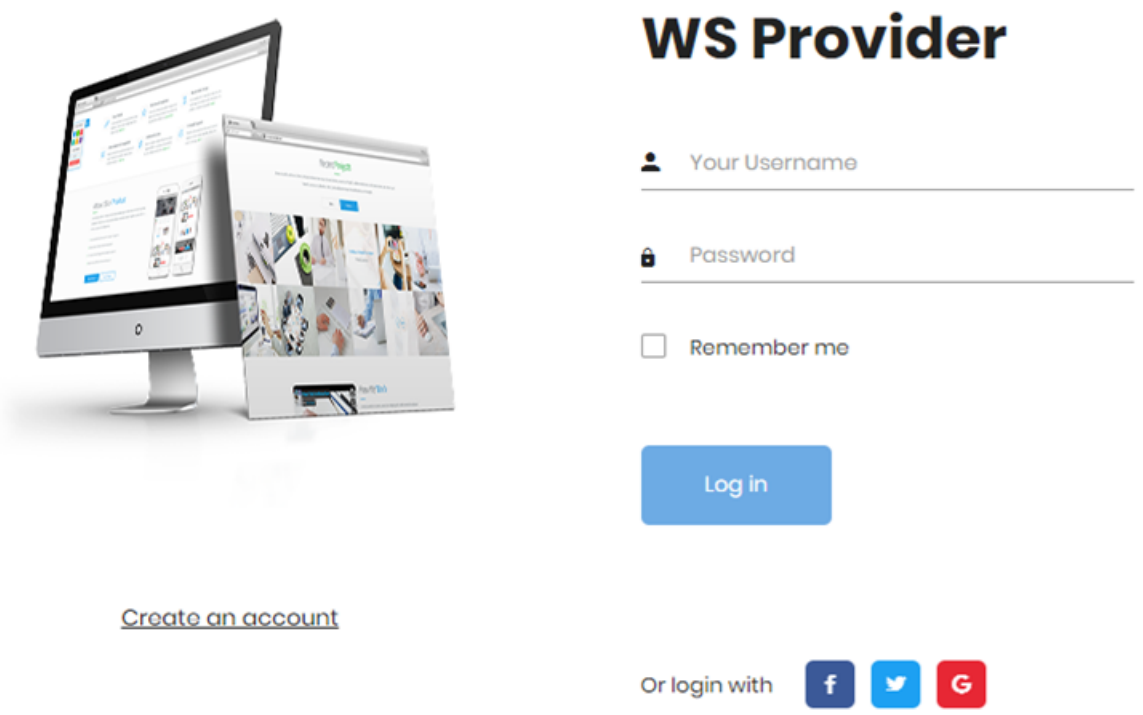

FIGURE 4.10 – Interface Login du fournisseur

4.8.4: Interface d'espace Fournisseur

Une fois un fournisseur s'inscrit, il sera rediriger vers son espace personnel qui est «Espace Fournisseur». à partir de cet espace, le fournisseur pourra publier les descriptions de son service. figure 4.11 montre l'interface de l'espace du fournisseur et ses différentes fonctionnalités.

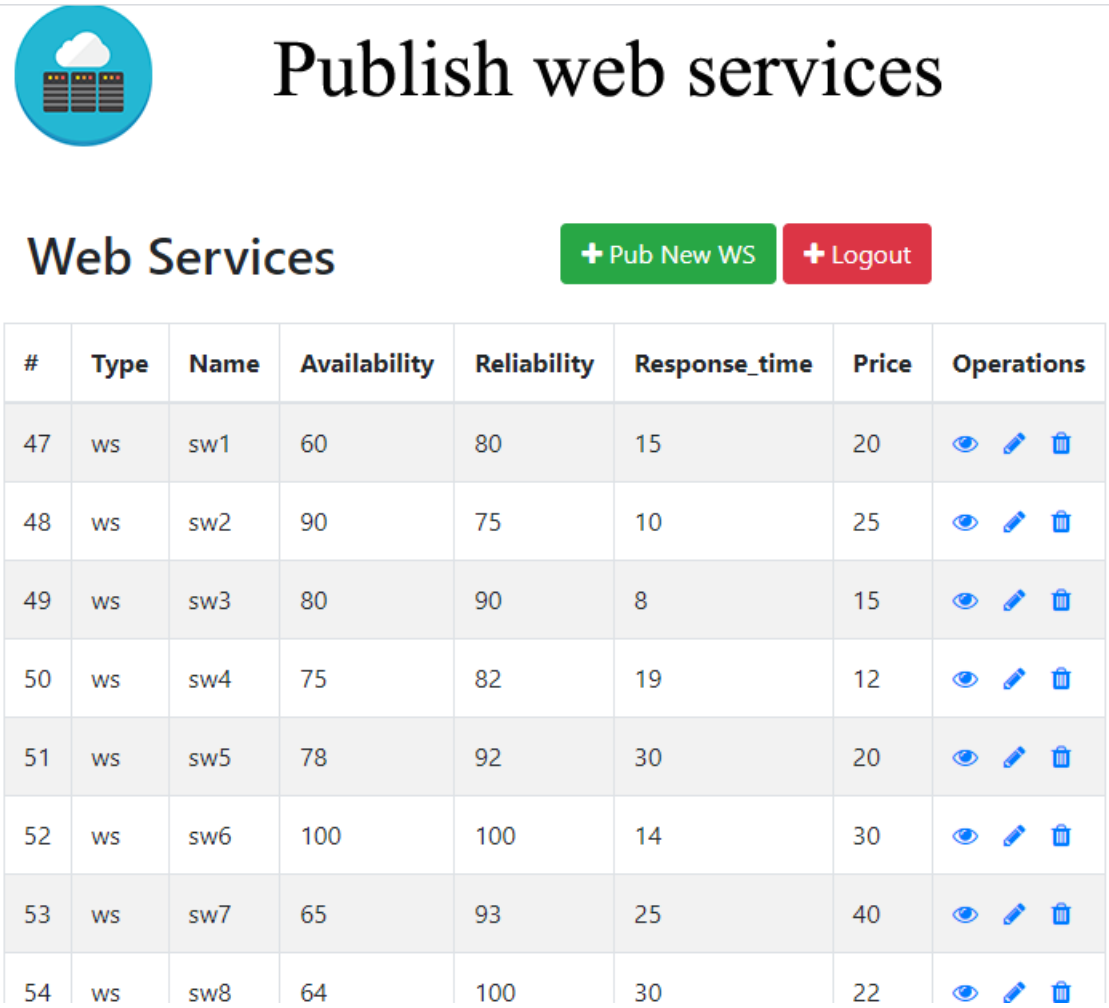

FIGURE 4.11 – Interface du fournisseur

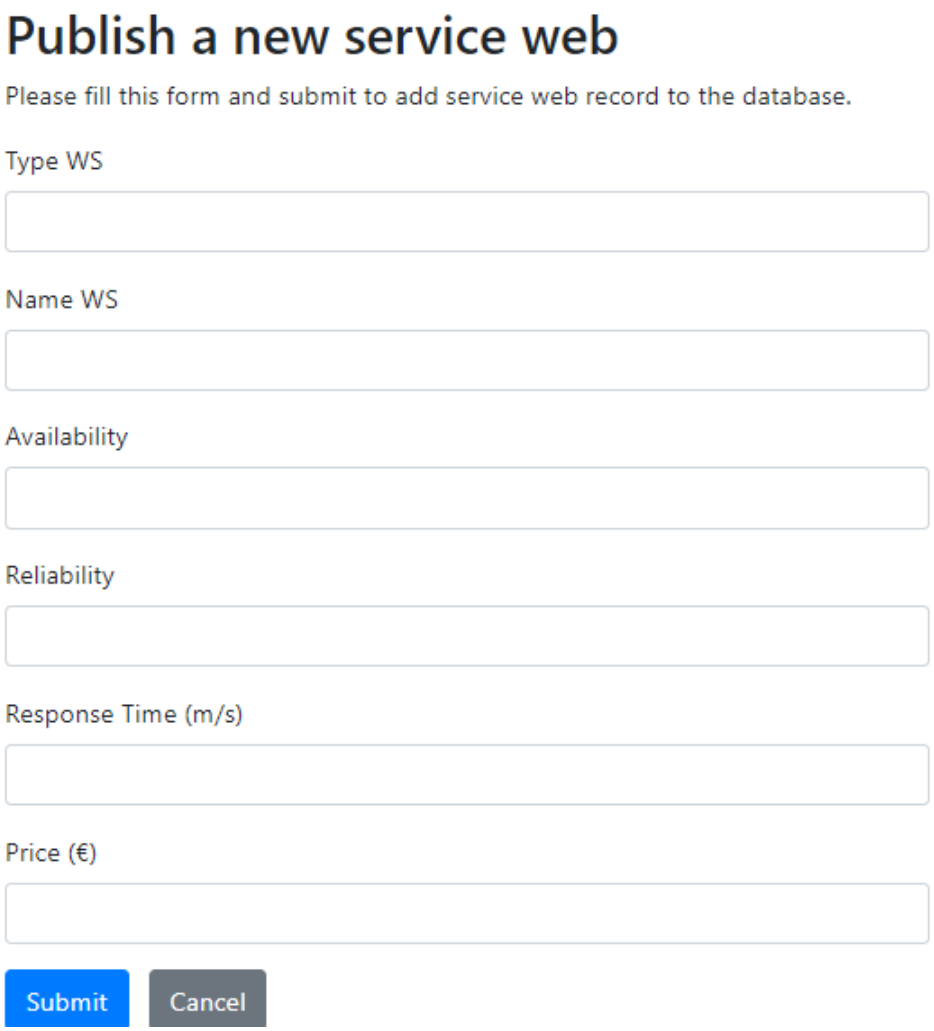

FIGURE 4.12 – Interface Fournisseur pour ajouter d'un nouveau service.

4.8.5: Interface d'Inscription du Client

Un client doit aussi s'inscrire afin de pouvoir réaliser ses tâches. la figure 4.9 montre l'interface d'inscription du client.

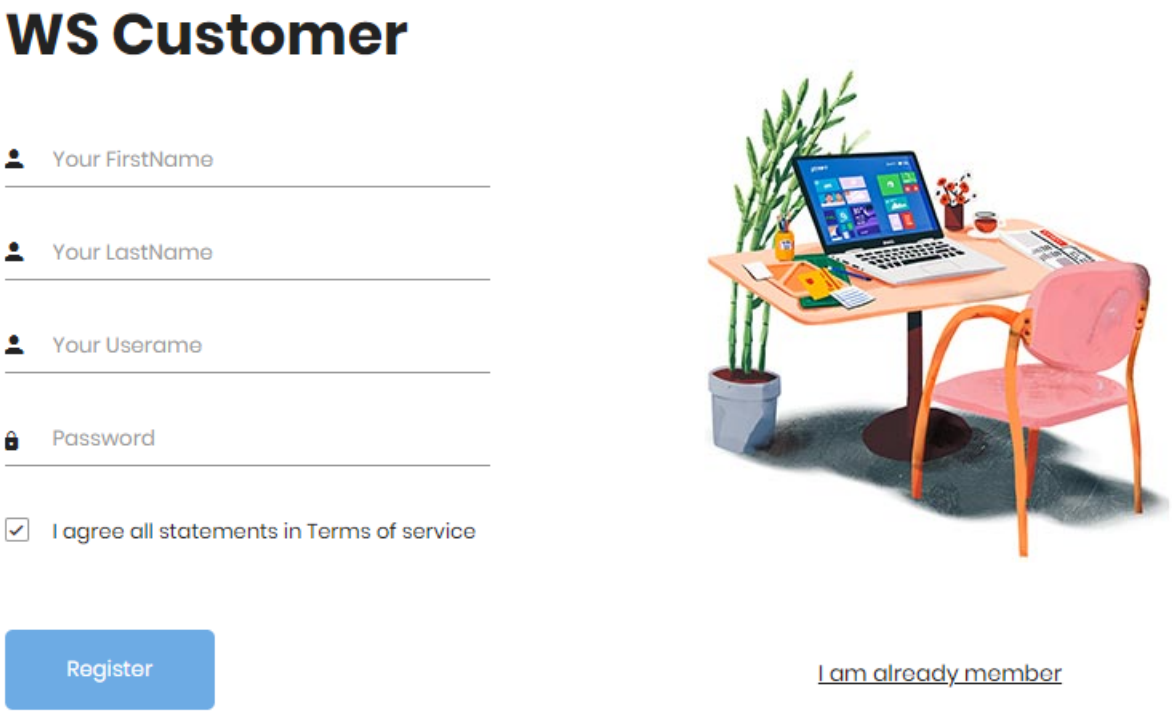

FIGURE 4.13 – Interface d'inscription du client.

4.8.6: Interface Login du Client

Un client qui demande de consommer un service web, il faut d'abord s'authentifier son identité.

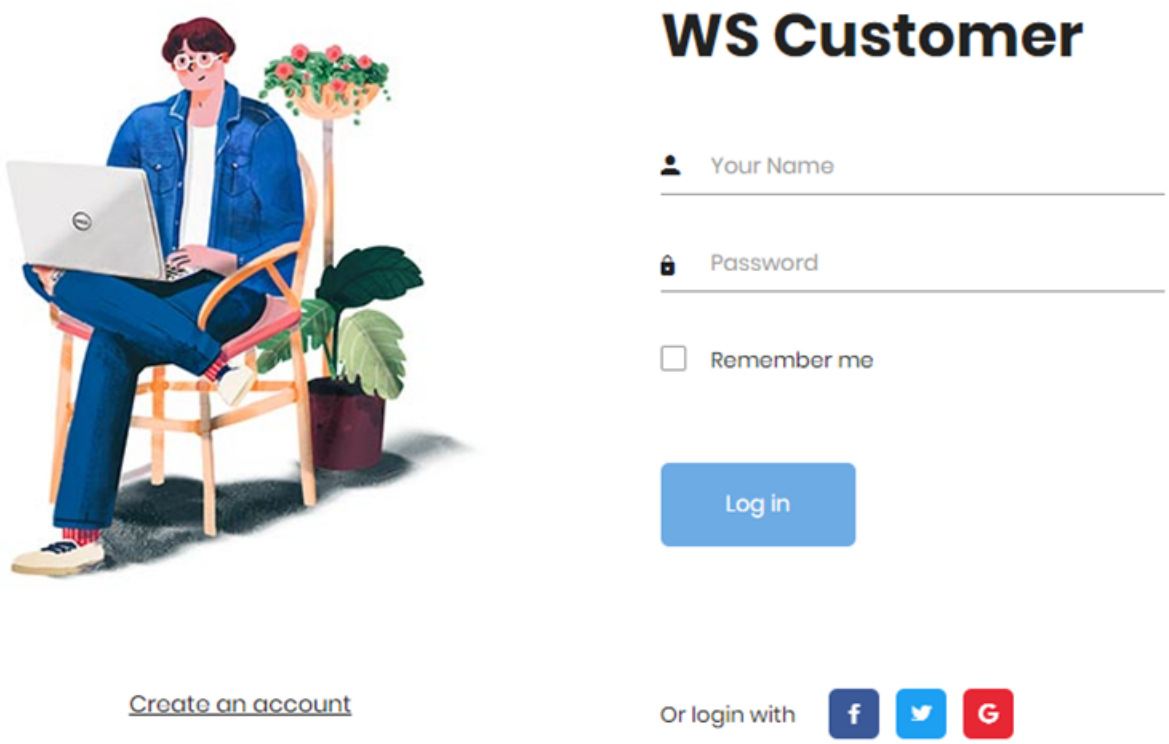

FIGURE 4.14 – Interface Login du client.

4.8.7: Interface d'Espace Client

Une fois le client s'inscrit, il sera redirigé vers «l'espace client» qui lui permet de sélectionner un service satisfaisant ses besoins à travers l'interface montré dans la figure 4.15.

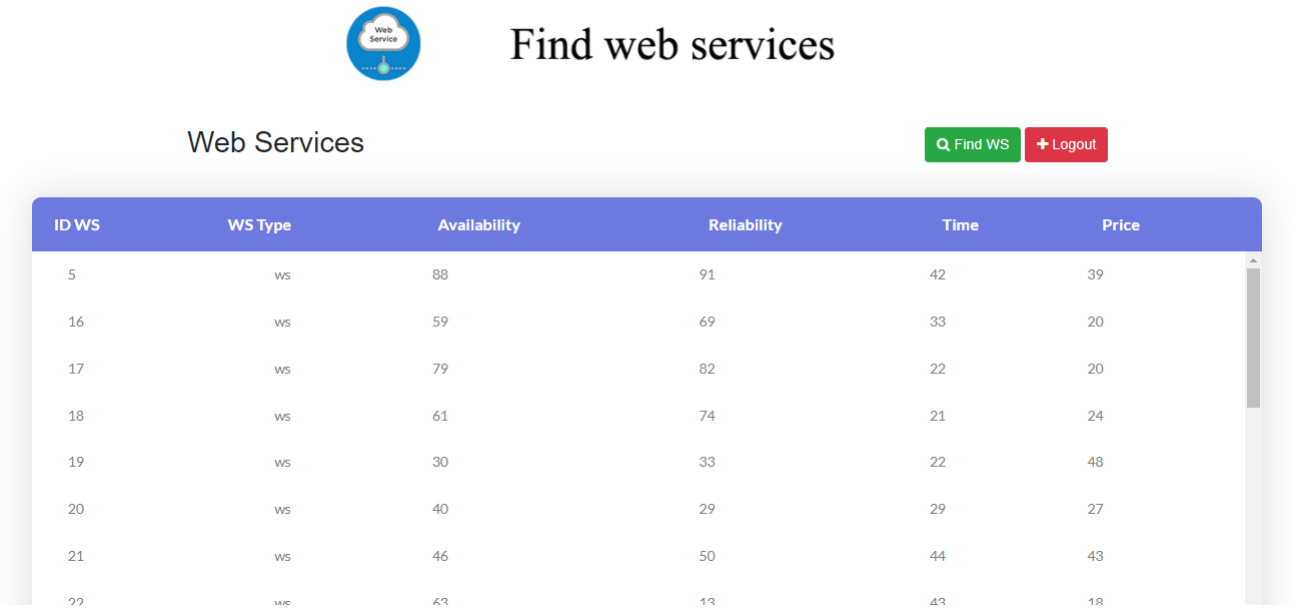

FIGURE 4.15 – Interface du Client.

FIGURE 4.16 – Interface d'espace client et affichage des services similaires.

4.9: Mise en œuvre et expérimentation

Dans cette partie, nous voyons les résultats de la mise en œuvre de notre solution proposé.le dossier sur lequel nous allons travailler s'appelle work.sa source est root. ici nous copions le fichier input.txt qui contient tous les services web .

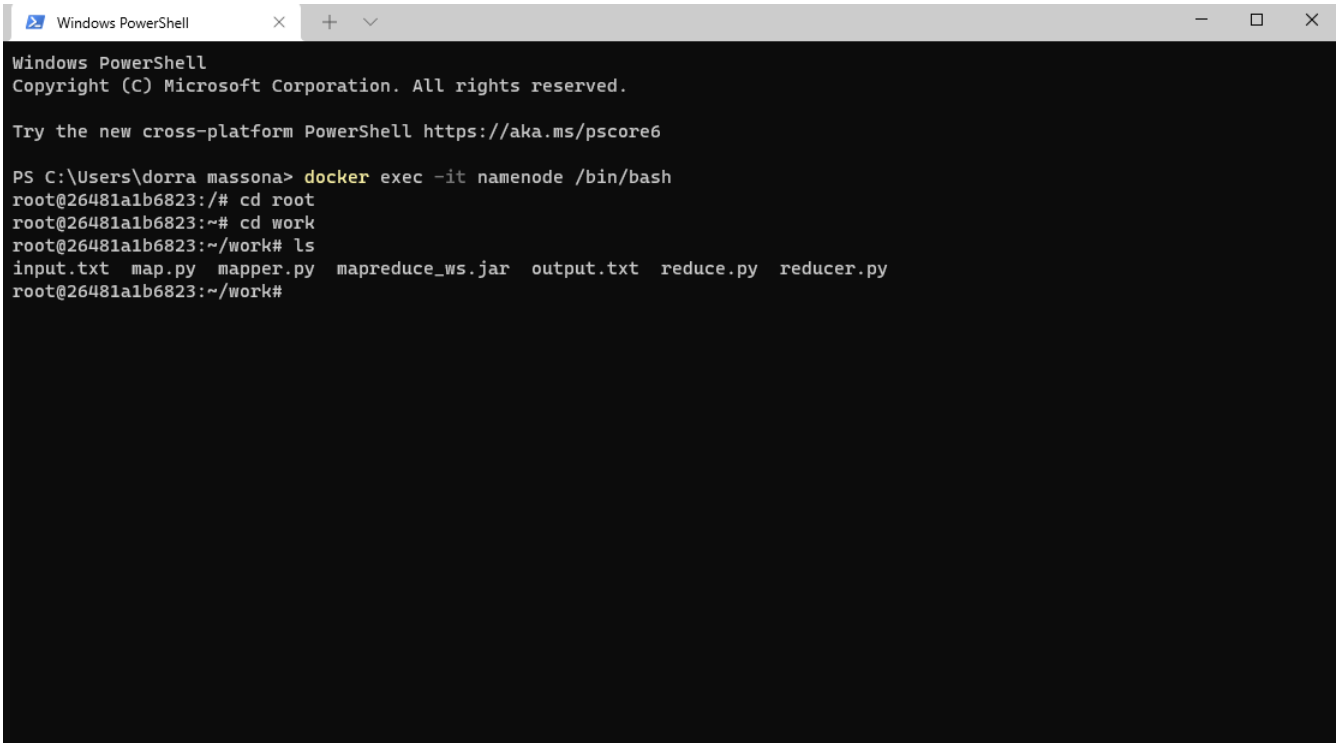

FIGURE 4.17 – Interface d'exécution

4.9.1: Gérer l'exécution du code MapReduce

Map : il lit le fichier d'entrée et renvoie les couples (clé /valeurs).

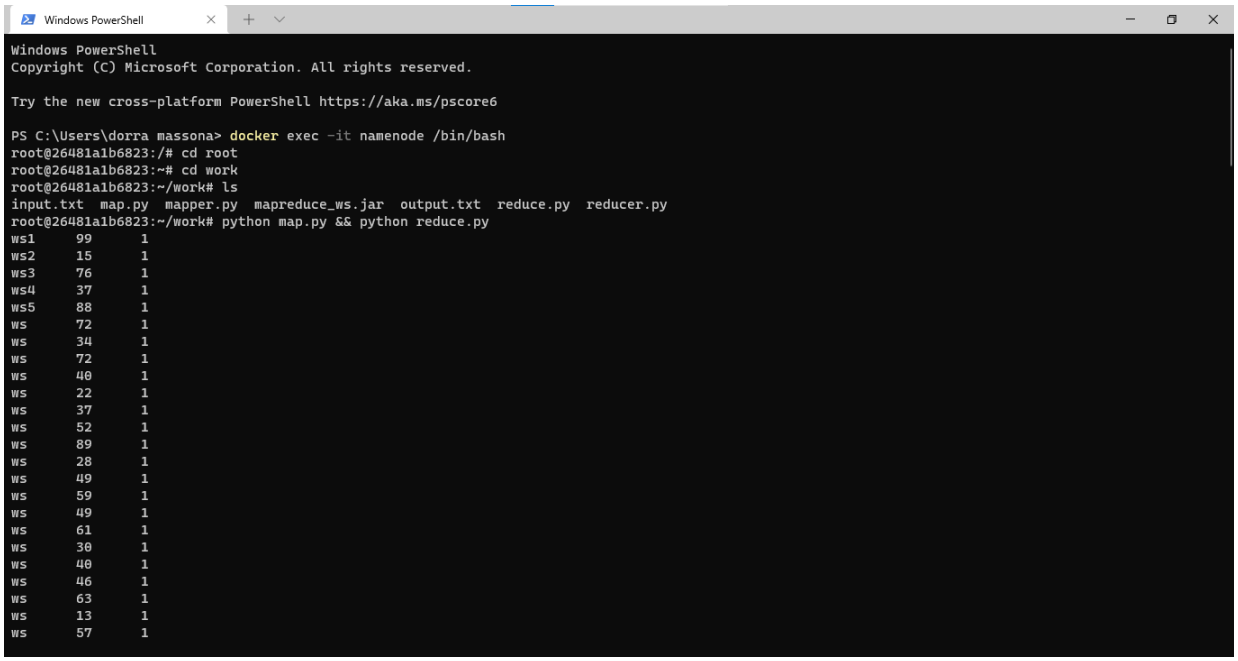

FIGURE 4.18 – Résultat sous forme Map.

Reduce : le processus de « Reduce » est commencé comme l'indiqué la figure 4.19.Et enfin, on obtient le résultat final (quatre services web incomparables).

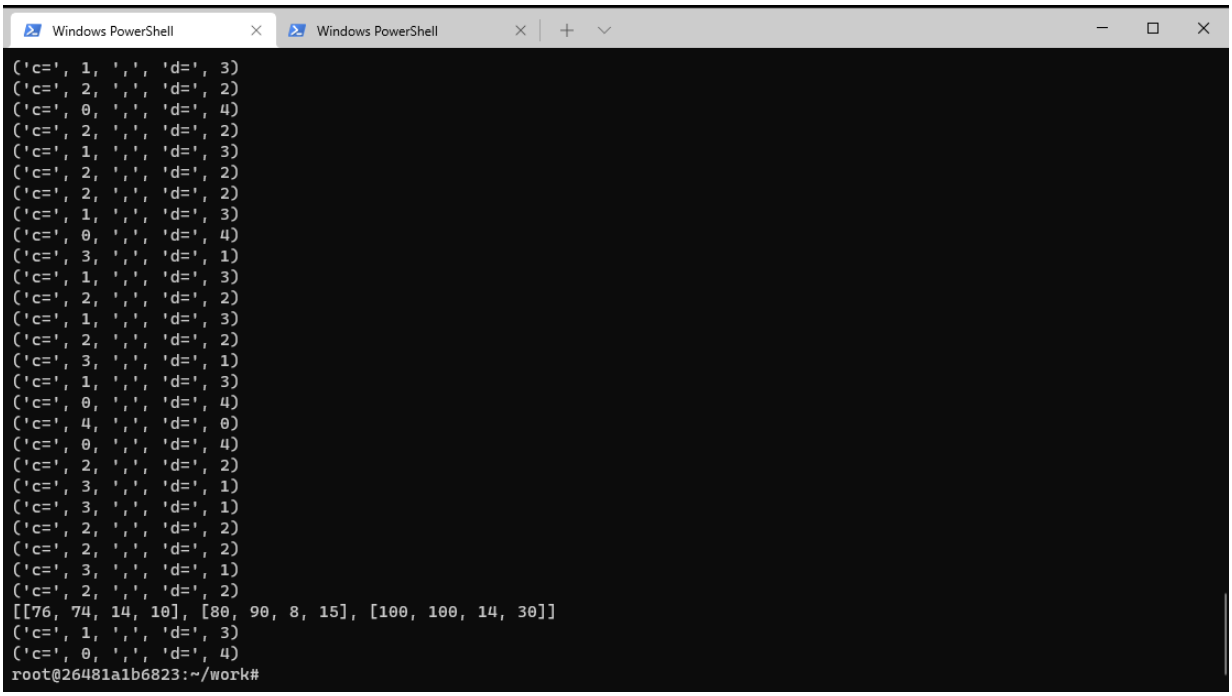

FIGURE 4.19 – Résultat sous forme Reduce.

4.9.2: Exécuter la logique floue

Nous copions le fichier de sortie de MapReduce pour appliquer la logique floue.

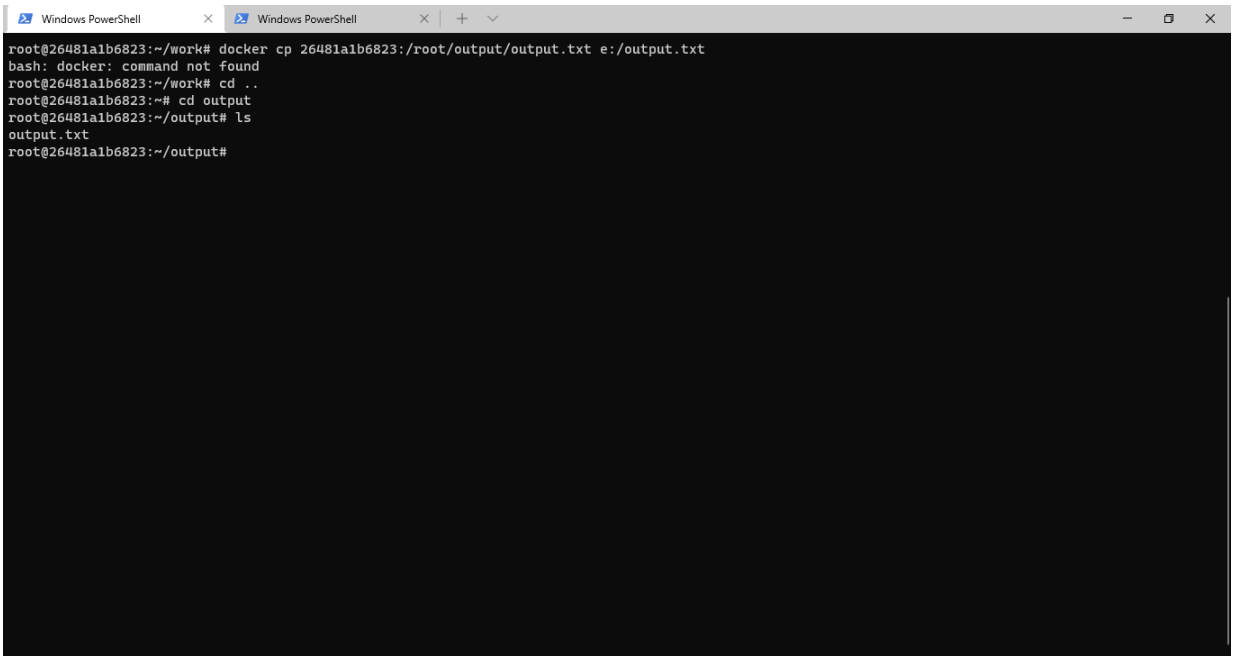

FIGURE 4.20 – Fichier de sortie .

Logique floue Le résultat de la logique floue est indiqué dans la figure 4.21. notez que le premier service web est la meilleure et la plus proche de la demande du client.

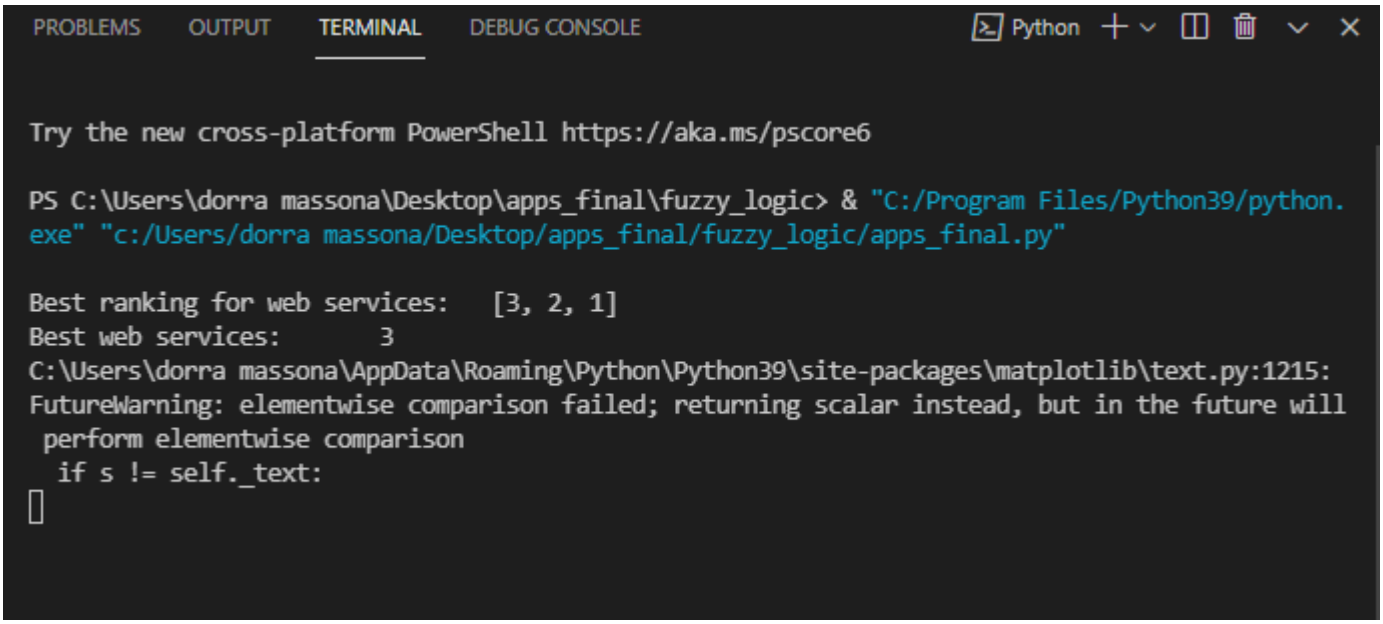

FIGURE 4.21 – Résultat sous forme logique floue.

FIGURE 4.22 – Interface affichant le meilleur service.

4.10: Conclusion

Ce chapitre a été consacré à la présentation du langage de programmation ainsi que l'environnement de développement. nous avons aussi présentés une description de notre application : les différentes interfaces et l'implémentation de l'approche proposée pour résoudre le problème de sélection de services web. dans ce chapitre, nous avons démontré l'efficacité de notre proposition pour la sélection des services web de manière distribué et parallèles pour gérer des grand nombre des services web.

.: Conclusion Générale et Perspectives

Conclusion

Les services web sont des technologies émergentes 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.
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. au cours de ce mémoire, nous nous sommes intéressés au problème de sélection des services web sur la base des besoins non fonctionnels (QoS).
Nous avons proposé une nouvelle approche qui sert à la découverte des services web, en s'appuyant sur la technique de Mapreduce et la logique floue. cette solution permet d'une part d'accélérer le calcul et introduire le parallélisme pour filtrer des grandes quantités des services web en fonction des besoins des clients. et d'autre part, il vise à l'évaluation des valeurs incertaines de QoS mesurées, et aide à effectuer la sélection de service web la plus proche au besoin d'utilisateur.

- Le premier chapitre est réservé à d'état de l'art. nous avons présenté les principaux standards liés à la technologie des services web, et leur architectures.
- Le second chapitre a présenté un aperçu sur les efforts déployés pour résoudre le problème de sélection de services web à base de QoS. et nous avons proposé une étude comparative de quelques solutions basée sur quelques critères bien ciblés.
- Le troisième chapitre est consacré aux contributions, où nous avons décrit l'ensemble de nos propositions.
- dans le chapitre 4, nous avons simulé le fonctionnement de nos propositions. cette simulation commence par la présentation de requête de l'utilisateur jusqu'à la sélection de service web adéquate à la demande de l'application avec des captures d'écrans.

Perspectives

En plus de nos contributions dans ce mémoire, certaines défis doivent encore être examinés. en fait, tout au long du travail présenté dans ce mémoire, nous avons formulé un certain nombre d'hypothèses qui peuvent être discutées et traitées dans les activités de recherche futures. nous les énumérons brièvement dans ce qui suit :

- Permettre aux clients de choisir les critères de QoS.
- Ajouter d'autres critères pour donner plus de choix aux clients.
- Améliorer la méthode de filtrage de services Web selon le besoin de l'utilisateur .
- Tester notre prototype avec un Data Set réelles des services web.
- Expérimenter notre approche avec un large data set des services Web.

Bibliographie

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, et al. Tensorflow : Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv :1603.04467*, 2016. 7

[2] R. Abdellah and M. Imane. Composition des services web. 2016. ix, 2, 7, 8

[3] G. D. Abowd, A. K. Dey, P. J. Brown, N. Davies, M. Smith, and P. Steggles. Towards a better understanding of context and context-awareness. In *International symposium on handheld and ubiquitous computing*, pages 304–307. Springer, 1999. 7

[4] A. Alkamari. Composition de services web par appariement de signatures. 2008. 16

[5] S. Borzsony, D. Kossmann, and K. Stocker. The skyline operator. In *Proceedings 17th international conference on data engineering*, pages 421–430. IEEE, 2001. 33

[6] C. Cherifi. *Classification et Composition de Services Web : Une Perspective Réseaux Complexes*. PhD thesis, Université Pascal Paoli, 2011. ix, 14

[7] D. N. El Houda. Architectures orientées services (soa). 15

[8] P. F. Fuchs and P. Poulain. *Introduction à la programmation Python pour la biologie*. PhD thesis, Université de Paris, 2020. 51

[9] M.-L. Gavard-Perret, D. Gotteland, C. Haon, and A. Jolibert. Méthodologie de la recherche. *Editions Pearson Education France*, 2008. 42

[10] A. Gustavo, C. Fabio, K. Harumi, and M. Vijay. Web services : concepts, architectures and applications. 2004. ix, 10, 11

[11] F. Lécué. *Composition de services web : Une approche basée liens sémantiques*. PhD thesis, Ecole Nationale Supérieure des Mines de Saint-Etienne, 2008. 2

[12] M. Merzoug. *Découverte et sélection des services web*. PhD thesis, Université de Tlemcen-Abou Bekr Belkaid. iii, 2, 7

[13] A. D. Messeguem. *Analyse des données avec apache spark*. PhD thesis, 2019. 32

[14] M. K. O. Mohy-eddine and H. Redouane. Thème. 12

[15] F. Retima. *Gestion de contexte pour l'Internet des objets*. PhD thesis, Université Mohamed Khider Biskra, 2019. 33

[16] A. A. A. Slimane. *Méthode de sélection intentionnelle des services, en fonction des exigences non-fonctionnelles des agents métier*. PhD thesis, Université Panthéon-Sorbonne-Paris I, 2012. 25

[17] K. Zernadji and S. Zertal. La découverte auto-organisée des services web dans un environnement dynamique. 2017. 10