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'études

Présenté pour l'obtention du Diplôme de

MASTER ACADEMIQUE

Domaine : **Mathématique et Informatique**
Filière :**Informatique**
Spécialité :**Systèmes Distribués et Intelligence Artificielle**

Présenté par :
ABAIDI Mouna
DADDA Siham

Thème :

Une approche effective pour la composition des services web

Soutenue le 15-06-2022 Devant le jury :

M.Farida Retima :	MCB	- Encadreur
M.Abdelkader Laouid :	MCA	- Présidente
M.Ismail Kertiou :	MCB	- Rapporteur

Année Universitaire : 2021/2022

Dédicace

“

JE exprime nos sentiments de profonde gratitude à mon Dieu le maitre de la vie, des temps et des circonstances, pour je avoir alloué de sa grâce inestimable et de je avoir donné la force, le courage et la patiente pour mener à terme ce travail,

À mes chers parents, Que nulle dédicace ne puisse exprimer ce que je leurs dois, pour leur bienveillance, leur affection et leur soutien Trésors de bonté, de générosité et de tendresse, en témoignage de mon profond amour et ma grande reconnaissance « Que Dieu vous garde »,

JE saisis l'occasion pour remercier mes frères : Larouci, Souad, Warda,Okacha, Samra, Maamer, Abou houraira , Mahammed bachir particulièrement Souad et mon professeur F.brahim ,

AUSSI , je remercie mes amis préférés : Sara, Roukia, mebrouka, saida,meriem Bennadji et tous mes amis du Collège,

ENFIN , que tous ceux qui, de loin ou de près, ont participé à la réalisation de ce travail trouvent ici l'expression de ma sincère gratitude.

”

- Mouna

Dédicace

“

MERCI a Dieu qui m'a donné la force et la capacité d'accomplir ce travail.... AVEC l'expression de ma reconnaissance, je dédie ce modeste travail à ceux qui, quels que soient les termes embrassés, je n'arriverais jamais à leur exprimer mon amour sincère……,

A mon cher père qui doit ma vie, ma réussite et tout mon respect…,

A mon mère la femme qui a souffert sans me laisser souffrir, qui n'a jamais dit non àmes exigences et qui n'a épargné aucun effort pour me rendre heureuse…,

A mes chères sœurs Soumeia Souhaila Fatima Khadidja et mon adorable petite sœur Dounia pour leurs encouragements permanents, et leur soutien moral, que Dieu les protège et leurs offre la chance et le bonheur……,

A mes chers frères Ibrahim Aymen pour leur chaleureusement supporté et encouragé tout au long de mon parcours…….

”

- Siham

Remerciements

Au terme de ce travail, nous remercions Dieu le tout puissant qui nous avons donné la force et la volonté d'achever ce travail. La première personne qui me vient à l'esprit et que nous tenons à remercier profondément est **Dr. Retima Farida**, Encadreur de mémoire, pour l'énorme soutien scientifique et moral qu'elle a su nous adresser pendant cette période. Nous la remercions aussi pour nous avoir toujours encouragées à aller de l'avant et pour nous avoir témoigné sa confiance totale. Nous ne pouvons qu'être infiniment reconnaissantes envers notre parent pour leur soutien indescriptible, leur patience, leur confiance et leurs nombreux sacrifices. Nous leur dédions avec plaisir ce travail, et qu'ils sachent que nous sommes conscients de ce que nous leur dois. Un merci très respectueux à nos amis dont la présence nous a permis de mener ce travail à bien. Finalement, nous sommes particulièrement agréables d'exprimer ici nous reconnaissance envers tous ceux qui ont contribué de près ou de loin, à la réalisation de ce travail.

Résumé

Avec la croissance exponentielle en nombre et en fonctionnalités des services web sur Internet, l'utilisateur est face à la difficulté de choisir entre un nombre important de services retournés par sa requête. Cependant, ces services diffèrent dans leurs qualités de services (QoS). Donc, il est devenu essentiel de proposer une solution qui permet au client de choisir les meilleurs services qu'il désire de manière efficace. De plus, il est difficile de satisfaire l'exigence complexe d'un utilisateur par juste un seul service web individuel ou atomique, ce qui a conduit les concepteurs à composer les services web existants.

L'objectif de ce travail consiste à proposer une solution effective pour la composition des services web en fonction des préférences de satisfaction du consommateur. Pour résoudre ce problème : nous avons proposé un MapReduce Skyline pour accélérer le filtrage des services web et le processus s'applique en parallèle pour traiter un grand nombre des services web.

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

Abstract

With the exponential growth in number and functionality of web services on the Internet, the user is faced with the difficulty of choosing between a large numbers of services returned by his request. However, these services differ in their qualities of service (QoS). Therefore, it has become essential to provide a solution that enables the customer to choose the best services in an efficient manner. Moreover, it is difficult to satisfy a user's complex requirement by just a single individual web service, which led designers to compose existing web services.

The objective of this work is to propose an effective solution for the composition of web services according to the consumer's satisfaction preferences. To solve this problem, we propose the following contribution : we proposed a MapReduce Skyline to speed up the filtering of web services and the process applies in parallel to process a large number of web services.

Keywords : Service Oriented Architecture (SOA), web service, selection of web services, composition of web services, QoS, MapReduce, Skyline.

ملخص

مع النمو الهائل في عدد ووظائف خدمات الويب على الإنترنت، يواجه المستخدم صعوبة في الاختيار بين عدد كبير من الخدمات التي يتم إرجاعها بناءً على طلبه. ومع ذلك، تختلف هذه الخدمات في جودة الخدمة .(QoS) لذلك، أصبح من الضروري توفير حل يمكّن العميل من اختيار أفضل الخدمات التي يريدها بطريقة فعالة. علاوة على ذلك، من الصعب تلبية متطلبات المستخدم المعقدة من خلال خدمة ويب فردية فقط ، مما أدى إلى المصممين لإنشاء خدمات الويب مركبة.

الهدف من هذا العمل هو اقتراح حل فعال لتكوين خدمات الويب وفقًا لتفضيلات المستهلك. لحل هذه المشكلة، نقترح المساهمة التالية: اقترحنا Skyline MapReduce لتسريع تصفية خدمات الويب ويتم تطبيق العملية بالتوازي لمعالجة عدد كبير من خدمات الويب.

كلمات مفتاحية : البنية الموجهة للخدمة ،(SOA) خدمة الويب، اختيار خدمات الويب، تركيب خدمات الويب، جودة الخدمة، MapReduce ، Skyline.

Table des matières

Table des figures

Introduction générale

Contexte

Le développement des systèmes d'information distribués a connu un succès retentissant ces derniers temps. Les systèmes d'information sont composés d'un ensemble de fonctions dont chacune est appelée service qui peut interagir avec son environnement à travers l'échange de messages.

Cependant, l'évolution successive des exigences des entreprises, la valeur réelle d'un service deviendra insatisfaisante, d'où la nécessité d'une nouvelle approche pour la composition de services afin d'apporter une valeur ajoutée. Dans ce contexte, nous parlons alors de la composition de services. Un service composé assemble divers services et coordonne les interactions au cours de leurs exécutions pour utiliser leurs fonctions et réaliser des tâches plus complexes.

Les services web est une technologie qui repose sur l'architecture SOA permettant à des applications de dialoguer (distribution de l'information) à distance via internet. L'approche SOA (Services Oriented Architecture) est un style architectural qui permet la découverte, l'intégration et l'utilisation des services web, permettant aux développeurs d'applications de résoudre plusieurs défis informatiques.

Dans cette architecture le fournisseur de service (serveur) publie ses services avec toutes ses caractéristiques dans un annuaire (registre). Le consommateur (client) accède à l'annuaire pour rechercher les services dont il a besoin en envoyant des requêtes.

Etant donné qu'il y a plusieurs services publiés sur internet, beaucoup d'entre eux, ne peuvent seuls satisfaire les besoins d'un client. Considérons, par exemple, le service d'organisation d'un voyage. Ce service peut se décomposer en plusieurs autres services, tels que :

- Réservation de billets d'avion ;
- Réservation de chambres d'hôtel ;
- Location de voiture ;
- Payement des différentes réservations ...etc.

Non seulement un seul de ces services ne permet pas d'organiser un voyage, mais seule une combinaison permet d'atteindre ce but. Il est possible évidemment d'effectuer cette

combinaison manuellement, c'est-à-dire que le client choisit séparément chaque service, et organisera leurs exécutions, en prenant en compte toutes les contraintes de temps et de précédences entre chaque service. Mais si la quantité de services proposés est considérable, atteindre le but d'organisation d'un voyage peut exiger beaucoup de temps et d'efforts de la part du client.

La composition des services web a pour objectif de déterminer une combinaison de services en fonction d'une requête d'un client. Du côté du client, cette composition semblera un unique service. La composition sera transparente au client, même si cette composition sera la combinaison de plusieurs services web.

Par ailleurs, une composition peut être composée de services atomiques ou composés. Les services atomiques sont caractérisés pour avoir une unique fonctionnalité. D'un autre côté, les services composés peuvent regrouper plusieurs fonctionnalités, c'est-à-dire plusieurs services atomiques. En conséquence, une composition peut être soit composée de services composés, soit de services atomiques mélangés.

Nous nous intéressons dans ce mémoire à la phase de rechercher d'un service composite basée sur la QoS dépend du profil QoS du service web, qui répondent aux besoins de l'utilisateur parmi les services web disponibles.

Problématique

Les recherches faites sur les services web par rapport à SOA couvrent des problèmes intéressants : la sélection des services Web, la composition des services web … etc. Notre étude est basée sur la notion de composition des services web qui est devenue un problème majeur à résoudre.

Le problème dans la composition des services web est comment distinguer entre les services qui répondent aux exigences de l'utilisateur à partir de plusieurs services sur la base des besoins fonctionnels et/ou non-fonctionnels. Les besoins non fonctionnels des services web sont généralement exprimés à l'aide des critères de qualité de service (QoS).

La qualité de service (QoS) [14]

Contribution

L'objectif de ce travail consiste à proposer une approche effective pour la composition des services web. Notre principale contribution consiste à rechercher d'un service composite à l'aide d'un MapReduce Framework. Cette solution permet accélérer le calcul de filtrage des services web disponibles selon le besoin de l'utilisateur de manière parallèle et distribué.

Plan du mémoire

Ce mémoire est composé de quatre chapitres qui sont organisé comme suit :

Chapitre 1 : Dans le premier chapitre, nous avons présenté le concept de service web ainsi que son fonctionnement, architecture SOA, et les principaux standards utilisés dans les services web.

Chapitre 2 : Dans le deuxième chapitre, nous avons présenté les critères de qualité de service web (QOS), et les techniques de sélection de service web avec quelque travaux connexes.

Chapitre 3 : Dans le troisième chapitre, nous avons présenté un ensemble des concepts utilisées dans notre solution proposé : MapReduce, calcul Skyline et la relation de dominance, puis nous avons expliqué l'idée proposée (la composition des services web de manière parallèle et distribuée sous hadoop).

Chapitre 4 : le dernier chapitre permet de présenter l'environnement matériel sur lequel notre système a été développé, les langages de programmation et les outils utilisés avec une étude de cas pour simuler la solution proposée.

Chapitre 1

Les services web

1.1 Introduction

Avec l'essor du web qui a eu lieu dans les dernières années, le besoin de permettre qu'une application client invoque un service d'une application serveur en utilisant internet, a surgi. Ce même besoin a été l'origine de ce qu'on appelle communément les services web. En tenant en compte que les services web permettent de connecter des applications différentes, l'utilité de cette technologie devient évidente et importante.

Dans ce chapitre, nous décrivons les services web en mettant en évidence la définition,les caractéristiques et l'architecture SOA (Service-Oriented-Architecture),ainsi les technologies des services web. Et nous terminant notre chapitre par les avantages de cette technologie.

1.2 Définition de services web

Plusieurs définitions ont été proposées dans la littérature. Parmi ces définitions :

Les services web sont la nouvelle vague des applications web. Ce sont des applications modulaires, auto-contenues et auto-descriptives qui peuvent être publiées, localisées et invoquées depuis le web. Les services web effectuent des actions allant de simples requêtes à des processus métiers complexes. Une fois qu'un service web est déployé, d'autres applications (y compris des services web) peuvent le découvrir et l'invoquer. [19].

1.3 Les caractéristiques d'un service web

La technologie des services Web repose essentiellement sur une représentation standard des données (interfaces, messageries) au moyen du langage XML. Cette technologie est devenue la base de l'informatique distribuée sur l'internet et offre beaucoup d'opportunités au développeur Web.

Un service Web possède les caractéristiques suivantes :

- Il est accessible via le réseau.
- Il dispose d'une interface publique (ensemble d'opérations) décrite en XML.
- Ses descriptions (fonctionnalités, comment l'invoquer et où le trouver ?) sont stockées dans un annuaire.
- Il communique en utilisant des messages XML. ces messages sont transportés par des protocoles Internet (généralement HTTP, mais rien n'empêche d'utiliser d'autres protocoles de transfert tels : SMTP, FTP, BEEP.etc) .
- L'intégration d'application en implémentant des services web produit des systèmes faiblement couplés : le demandeur du service ne connaît pas forcément le fournisseur. Ce dernier peut disparaître sans perturber l'application cliente qui trouvera un autre fournisseur en cherchant dans l'annuaire. [7]

1.4 Les technologies des services web

XML, SOAP (Simple Object Access Protocol), WSDL (Web Services Description Language) et UDDI (Universal Description, Discovery and Intégration) sont les technologies dominantes des services web est montré dans la figure 1.1 comme suit :

Fig. 1.1 : Les technologies des services web.

1.4.1 XML (eXtensible Markup Language)

XML est une famille de technologies développées au sein du W3C (World Wide Web Consortium). XML est née de la tentative de mettre SGML sur le Web. La première spécification de XML se concentre sur les données, contrairement à HTML qui focalise sur la présentation. XML permet donc de transformer Internet d'un univers d'information et de présentation de sites Web statiques à un univers Web programmable et dynamique.

XML est largement utilisé, Il est indépendant des plates-formes informatiques. Il est lisible par l'humain mais est destiné à être lu par la machine. Il est aussi flexible en ce sens que vous pouvez définir d'autres langages à partir d'XML, avec facilité d'utilisation.

1.4.2 SOAP (Simple Object Access Protocol)

SOAP est un protocole de la famille XML servant à l'échange d'informations dans un environnement distribué et décentralisé. Il est considéré comme la technologie la plus importante des services web. Le standard SOAP a été proposé au W3C par Microsoft, IBM, Lotus, DevelopMentor et Userland.

Le standard SOAP définit trois éléments composant d'un message : l'enveloppe, l'en-tête du message et le corps du message. L'enveloppe définit le cadre pour décrire ce qui est dans le message et comment le traiter. Les règles d'encodages sont placées dans l'en-tête et servent à exprimer et définir le mécanisme de représentation des données. Le corps du message permet de transmettre les requêtes et les réponses entre les systèmes.

Le modèle d'échange de message en SOAP est indiqué dans la figure 1.2.

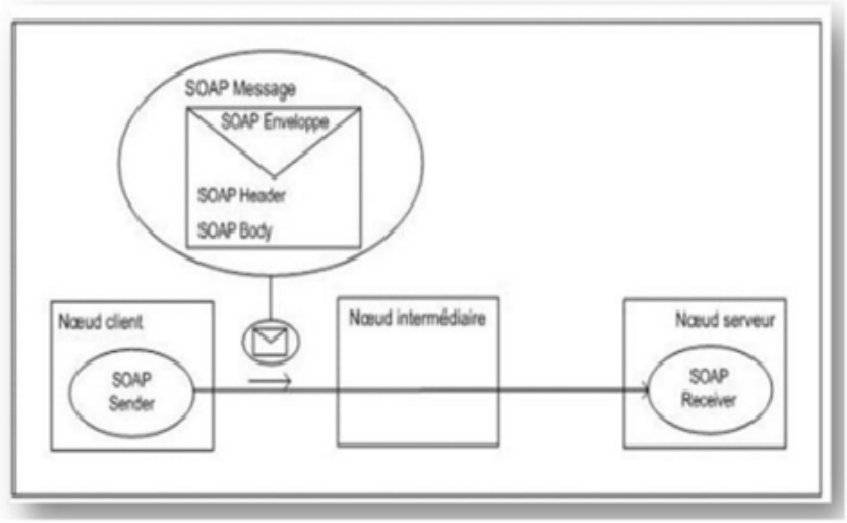


FIG. 1.2 : Modèle d'échange de message en SOAP.

1.4.3 WSDL (Web Services Description Language)

WSDL est un langage permettant de décrire les services web et les données attendues par ces services. L'utilité de WSDL est donc de décrire et publier le format et les protocoles d'un service web de manière homogène par l'utilisation de XML. Cela permettra au requérant et à l'émetteur d'un service de comprendre les données qui seront échangées.

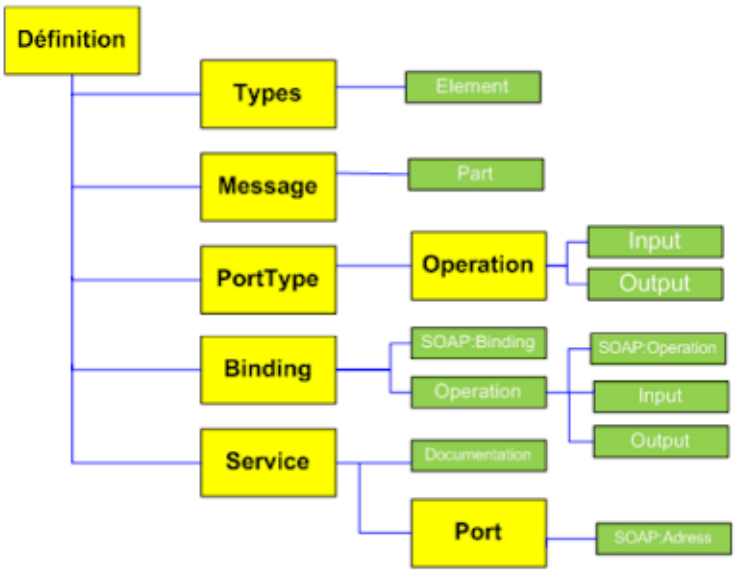


FIG. 1.3 : Structure d'un document WSDL.

La figure 1.3 représente les composants du WSDL :

- **Data types** : est l'élément qui définit les types de données utilisées dans les messages echangés par le service web.

- **Message** : spécifie les types d'opérations supportées par le service web.

  Il permet d'incorporer une séquence de messages corrélés sans avoir à spécifier les caractéristiques du flux de données.

- **Port Type** : est un groupement logique ou une collection d'opérations supportées par un ou plusieurs protocoles de transport, il est analogique à une définition d'un objet contenant un ensemble de méthodes. Au niveau concret, le service web est défini grâce aux éléments Bindings et Service et Port. Ces deux derniers décrivent des informations liées à un usage contextuel du service web. On y trouve l'adresse du fournisseur implémentant le service, et le service qui est représenté par les adresses des fournisseurs.

- **Bindings** : Décrit la façon dont un type de port est mis en oeuvre pour un protocole particulier (HTTP par exemple), et un mode d'invocation (SOAP par exemple). Cette description est faite par un ensemble donné d'opérations abstraites. Pour un type de port, on peut avoir plusieurs liaisons, pour différencier le mode d'invocation ou de transport de différentes opérations.

- **Port** : Spécifie une adresse URL qui correspond à l'implémentation du service web par un fournisseur et identifie une ou plusieurs liaisons aux protocoles de transport pour un Port Type donné.

- **Service** : Spécifie l'adresse complète du service web, et permet à un point d'accès d'une application distante de choisir à exposer de multiples catégories d'opérations pour divers types d'interactions.

  En résumé le document WSDL est indispensable au déploiement de services web et décrit deux documents essentiels : un document pour l'interface du service web et un autre pour son implémentation. La publication et la recherche d'un service web au sein de l'annuaire UDDI se font via ces deux types de documents WSDL.

1.4.4 UDDI (Universal Description, Discovery and Integration)

UDDI définit les mécanismes permettant de répertorier des services web. Ce standard régit donc l'information relative à la publication, la découverte et l'utilisation d'un service web. En d'autres mots, UDDI détermine comment nous devons organiser l'information concernant les services web qu'elle offre à la communauté afin de permettre à cette communauté d'y avoir accès. En fait, UDDI définit un registre des services web sous un format XML. Ce registre peut être public, privé ou partagé.

Tout comme WSDL, UDDI est une création du trio IBM, Microsoft et Ariba. [4]

1.5 Architecture Orientée Service (SOA)

1.5.1 Définition

Une architecture orientée services est essentiellement un ensemble de services. Ces services communiquent entre eux. La communication peut impliquer soit un simple transfert de données, soit deux services ou plus coordonnant certaines activités. Certains moyens de connecter les services les uns aux autres sont nécessaires.

La figure 1.4 illustre une architecture de base orientée services : Il montre un consommateur de services à droite envoyant un message de demande de service à un fournisseur de services à gauche. Le fournisseur de services renvoie un message de réponse au consommateur de services. Les connexions de demande et de réponse ultérieure sont définies d'une manière qui est compréhensible à la fois pour le consommateur de services et le fournisseur de services. La façon dont ces connexions sont définies est expliquée dans les services web. Un fournisseur de services peut également être un consommateur de services . [11]

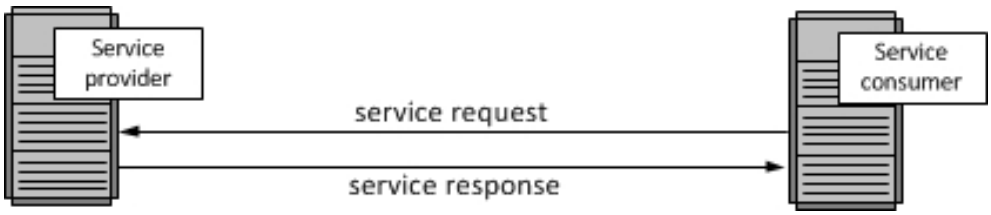


Fig. 1.4 : Service-Oriented Architecture Basics.

1.5.2 Caractéristiques de SOA

Les caractéristiques qui favorisent le SOA, par rapport aux autres solutions distribuées telles que le CORBA et le DCOM sont particulièrement :

- Le couplage lâche entre les services par échange de message qui permet de minimiser les dépendances des services, ce qui implique une meilleure flexibilité de la solution.
- L'interopérabilité : qui est l'un des enjeux de l'architecture SOA, assure la gestion de l'hétérogénéité des plateformes, des langages de programmation se basant sur l'infrastructure Internet.
- La réutilisation des services et la facilité d'intégration.
- La mise à l'échelle est rendue possible grâce à la découverte, la localisation et l'invocation dynamique d'autres services à partir du registre.
- Une solution moins coûteuse vue l'utilisation des standards et des protocoles internet fortement répandus et peu coûteux tel que XML et HTTP.[6]

1.5.3 Les acteurs de SOA

La figure 1.5 représente le modèle fonctionnel de l'architecture SOA qui comprend trois acteurs :

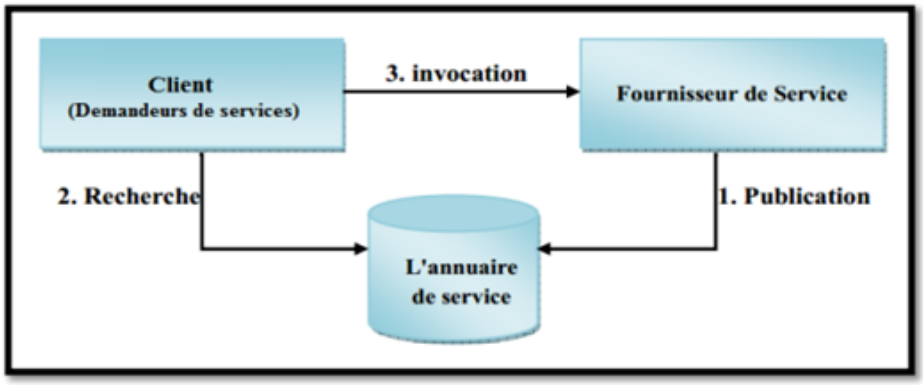


FIG. 1.5 : Modèle fonctionnel de l'architecture SOA.

- Le fournisseur du service : il désigne l'entité propriétaire du service. D'un point de vue technique, un fournisseur est constitué par la plate-forme d'accueil du service.
- Le client ou demandeur du service : C'est le consommateur de service. D'un point de vue technique, le service client est constitué de l'application qui va rechercher et invoquer un service.
- L'annuaire de service : Correspond à un registre de descriptions de services offrant des facilités de publication de services à l'intention des fournisseurs ; et des facilités de recherche de services à l'intention des clients. En d'autres termes, l'annuaire joue le rôle d'intermédiaire entre les clients et les fournisseurs de services.[10]

1.6 Architecture des services web

L'architecture des services web est un cadre conceptuel, nous discutons dans cette section deux types d'architectures. La première est l'architecture dite de référence et traditionnellement utilisée pour les services web isolés. La seconde est l'architecture qui est plus complète et est fréquemment utilisé lors de composition de services web. Elle est appelée architecture étendue ou encore en pile.

1.6.1 Architecture de référence

L'architecture de référence des services web (Figure 1.6) s'articule autour des trois rôles suivants :

- fournisseur de service : correspond au propriétaire du service.

- Le client : correspond au demandeur de service.
- L'annuaire des services : correspond à un registre de descriptions de services web.

Les interactions de base entre ces trois rôles incluent les opérations de publication, de recherche et de liens d'opérations. Pour garantir l'interopérabilité de ces trois opérations, des propositions de standards ont été élaborées pour chaque type d'interaction. Nous citons, notamment les standards émergents suivants :

- SOAP définit un protocole de transmission de messages basé sur XML.
- WSDL Introduit une grammaire commune pour la description des services.
- UDDI fournit l'infrastructure de base pour la publication et la découverte des services.

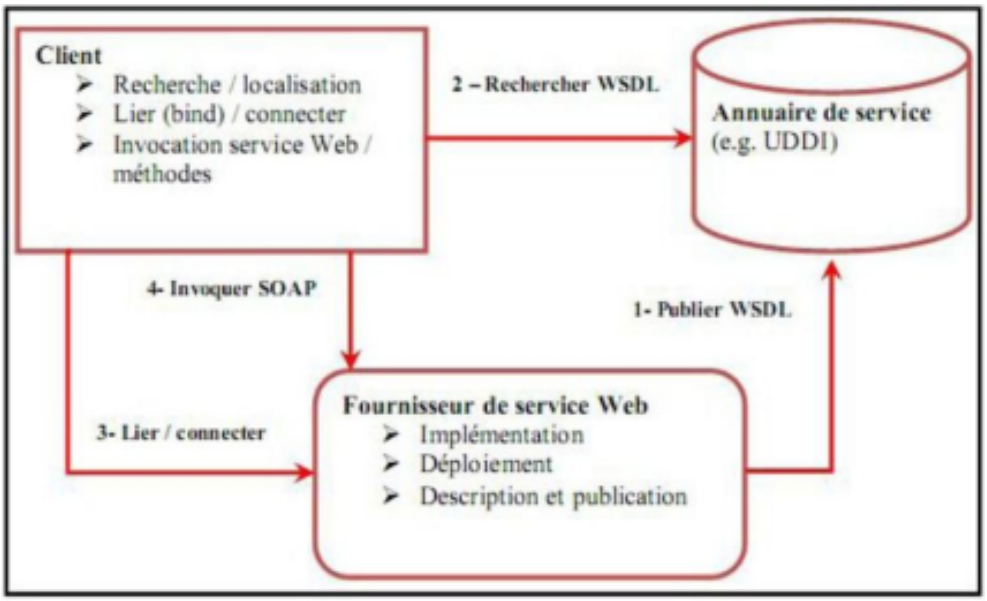


Fig. 1.6 : Architecture de référence des services Web.

Dans la figure 1.6, nous distinguons les échanges des messages établis par SOAP, la description de service par WSDL et la recherche de services web via le registre UDDI

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 via des protocoles standards. Donc il faut introduire une nouvelle architecture suffisante et complète.

1.6.2 Architecture étendue

Le nom de pile des services web est venu d'une architecture étendue est constituée de plusieurs couches se superposant les unes sur les autres, La figure 1.7 décrit un exemple d'une telle pile. La pile est constituée de plusieurs couches, chaque couche s'appuyant sur un standard particulier.

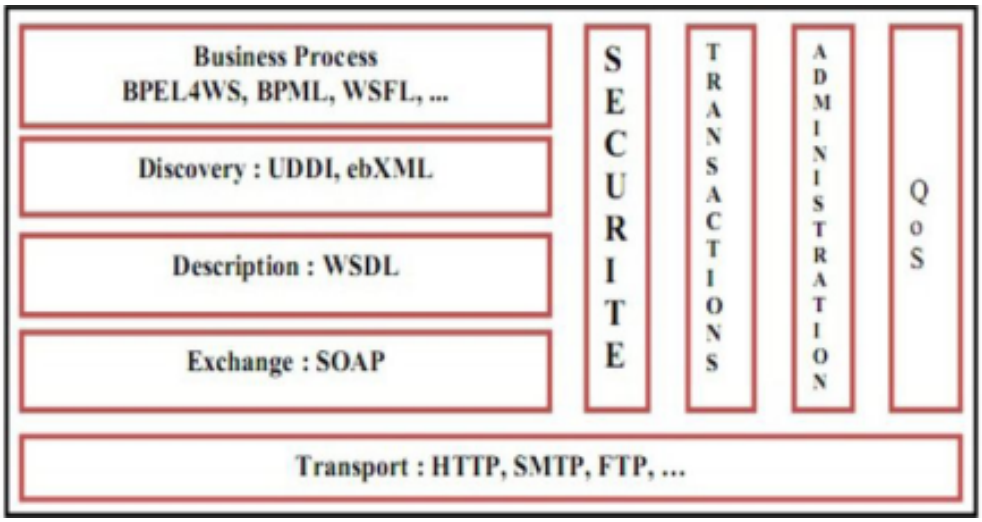


FIG. 1.7 : Architecture en Pile des services Web.

Nous apportons une explication de la mise en relief des trois types de couches (Voir Figure 1.7) :**L'infrastructure de base (Discovery, Discription, Exchange)** : ce sont les fondements techniques établis par l'architecture de référence. Nous distinguons les échanges des messages établis par SOAP, la description de service par WSDL et la recherche de services Web que les organisations souhaitent utiliser via le registre UDDI ; **Couches transversales (Security, Transactions, Administration, QoS)** : Ce sont ces couches qui rendent viable l'utilisation effective des services Web dans le monde industriel ;**La 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.

1.7 Les avantages des services Web

- Les services web fournissent l'interopérabilité entre divers logiciels fonctionnant sur diverses plates-formes.
- Les services web utilisent des standards et protocoles ouverts.
- Les protocoles et les formats de données sont au format texte, ce qui est permet de faciliter la compréhension du fonctionnement global des échanges.
- Basés sur le protocole HTTP, les services Web peuvent fonctionner au travers de nombreux pare-feu sans nécessiter des changements sur les règles de filtrage.
- Les outils de développement, s'appuyant sur ces standards, permettent la création automatique de programmes utilisant les services web existants.

1.8 Conclusion

Dans ce chapitre nous avons décrit les principaux concepts de service web. Nous avons commencé par présenter la définition et les caractéristiques de cette technologie. Ensuite nous avons expliquées l'architecture SOA. Dans le prochain chapitre, nous présenterons les différentes méthodes utilisées pour la sélection des services web.

Chapitre 2

La sélection de service à base de QoS

2.1 Introduction

La réponse à la requête d'utilisateur permet de choisir un service web parmi un ensemble des services web découverts qui peut satisfaire ses besoins.

Il est essentiel que cette sélection soit adaptée aux préférences de l'utilisateur en raison du fait qu'un utilisateur peut exiger de haute qualité tandis que l'autre peut exiger moins de coût.

Si la réponse à la requête d'utilisateur exige la sélection d'un seul service web non composite, alors la sélection est très simple. Pendant que 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 composants qui répondent mieux aux besoins des utilisateurs.

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. Ce dernier est considéré comme le critère non fonctionnel plus important pour la sélection du service web.

Nous montrons le processus de découverte, la sélection et la composition des services web dans la Figure 2.1

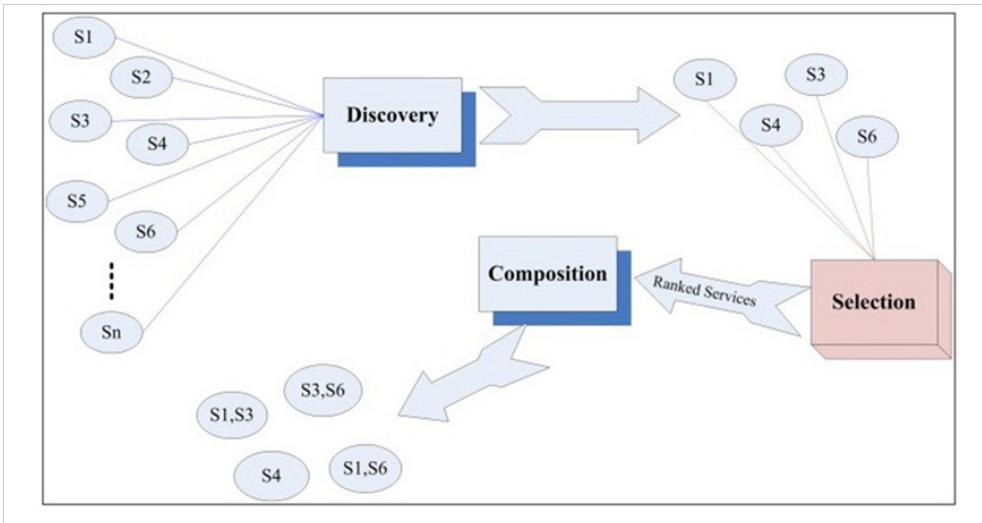


Fig. 2.1 : Découverte, la sélection et la composition des services web.

Dans ce chapitre nous définissons le terme de qualité de service web (QoS), et nous citons les techniques de sélection de service web avec quelques travaux connexes.

2.2 La Qualité de Service

2.2.1 Définition de Qualité de Service

Les services web sont composés d'attributs fonctionnels et non fonctionnels. Les attributs non fonctionnels sont appelés qualité de service (QoS). Ces attributs sont des

caractéristiques d'un service qui lui confère l'aptitude à satisfaire les 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 pour sélectionner parmi plusieurs services web découverts ceux qui respectent les contraintes de temps imposées.[17] [1]

2.2.2 Critères de QoS

Plusieurs critères de qualité de service sont décrits dans différents travaux. Parmi ces critères, nous définissons les plus populaires comme suit :

- **Disponibilité** : la probabilité que les ressources, les services sont disponibles pour les parties autorisées en tout temps pour l'utilisation.
- **Fiabilité** : La capacité d'un service à exécuter les fonctions requises dans des conditions spécifiées pendant une période donnée.
- **Temps de réponse** : la durée entre l'envoi d'une demande par un utilisateur de service et la réception d'une réponse.
- **Accessibilité** : Les services web devraient avoir une grande accessibilité. L'accessibilité représente, que le service web soit capable de répondre à la demande du client. Haute accessibilité peut être obtenue, par exemple en construisant des systèmes évolutifs.
- **Coût d'exécution** : Le coût à dépenser pour exécuter un service web. Ce coût peut être fourni par le fournisseur de service.
- **Latence** : c'est la mesure de retard prévu en secondes entre le moment où l'exécution commence et le moment où l'exécution finit.
- **Réputation** : la réputation d'un service est une mesure de sa fidélité. Elle dépend principalement à des expériences des utilisateurs. Les différents utilisateurs peuvent avoir le même service mais avec différents avis. La valeur de la réputation d'un service web est définie comme le rang moyen indiqué par les utilisateurs.[20]
- **Robustesse** : représente le degré auquel un service web peut fonctionner correctement.
- **Exactitude** : Les services web doivent être fournis avec une grande précision. La précision est définie comme le taux d'erreur généré par le service web. Le nombre d'erreurs que le service génère sur un intervalle de temps doit être minimisé.
- **Interopérabilité** : Les services web doivent être interopérables entre les différents environnements. Pour mettre en œuvre les services, les développeurs ne pensent à aucun langage de programmation ni à aucun système dans lequel les services ont été développés.

- **La sécurité** : Les services web devraient être fournis de manière sécurisée. Avec l'augmentation de l'utilisation des services web qui sont diffusés sur l'internet public, il existe une préoccupation croissante en matière de sécurité. Plusieurs méthodes pour protéger ces services comme : authentification, confidentialité, cryptage, non-répudiation . [8]

2.3 Technique de sélection de service web

Plusieurs travaux ont été proposés pour résoudre le problème de sélection des services web. on distingue 03 grandes classes (la sélection mono-objective, multi-objective, mono et multi-objective) comme l'illustrée la figure 2.2

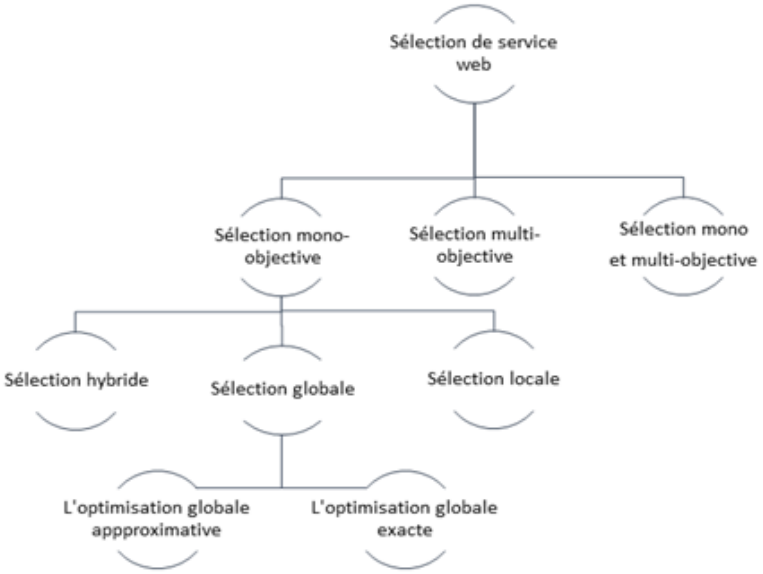


FIG. 2.2 : Les approches de sélection de services web à base de QoS.

2.3.1 Sélection mono-objective

Cette approche suppose que les m valeurs de qualité de service sont agrégés en un seul score. On considère une seule fonction objective qui associe des poids aux différents attributs de QoS. Elle est divisée en 03 sous classes : La sélection locale, globale et hybride (globale et locale).

Sélection Globale

Cette approche explore un espace de recherche dont les nœuds sont des compositions complètes (c.-à-d. contenant toutes les tâches), on distingue deux sous classes d'optimi-

sation globale : exacte et approximative.

- **Optimisation globale exacte** Elle se base sur la programmation entière ou la programmation par contrainte, ou les énumérations exhaustives. Ces méthodes donnent des résultats optimaux mais elles ont un temps d'exécution exponentiel.
- **Optimisation globale approximative** Cette catégorie consiste à explorer une partie de l'espace de recherche, en adoptant des recherches heuristiques ou des meta-heuristiques.

  Les recherches meta-heuristiques faites par **Dreo et al, 2003** sont des algorithmes d'optimisation génériques, qui adoptent une recherche locale ou globale, elles permettent de donner des résultats quasi-optimaux, tout en ayant un temps d'exécution abordable. Par opposition, les recherches heuristiques ne peuvent être généralisées pour le reste des problèmes.[12]

Sélection Locale

Elle consiste à élire un seul service de chaque classe en utilisant une fonction objective (indépendamment des autres classes). Ensuite elle compose les n résultats qui correspondent aux n classes. Ce type de méthodes ne prend pas en charge les contraintes globales (ex : le coût global de la composition), ce qui la rend obsolète pour les problèmes réels.

Alrifai et al, 2008 adaptent la sélection locale pour gérer les contraintes globales. Pour cela ils divisent les contraintes globales, en contraintes locales en se basant sur la distribution statistiques des valeurs de QoS.[3]

2.3.2 Sélection Hybride

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

2.3.3 Sélection Multi-Objectives

Les problèmes d'optimisation sont des problèmes multi-objectifs. alors on n'a pas une seule solution optimale mais un ensemble de solutions de compromis.

Plusieurs travaux, ont été proposés pour la recherche multi objectives des services skylines, la majorité utilise les meta -heuristiques telles que l'algorithme NSGAII (Non dominated Sorting Genetic Algorithm II).

Dans **Claro et al, 2005** les auteurs considèrent 04 fonctions objectives [9] :

- La Minimisation du coût
- La Minimisation de la durée,

- La maximisation du chiffre d'affaire,
- La maximisation de la réputation.

2.3.4 Sélection Mono et Multi-objective

Alrifai et al., 2010 proposent des approches combinant la sélection multi-objective (Skylines) et mono-objective. Enparticulier, ils proposent 03 algorithmes à base de skylines.[2]

Le premier est nommé « exact skylines ». Il extrait les services skylines de chaque classe et applique une optimisation globale à base de MIP(la programmation en nombre entiers ou Mixed Intege Programming) sur ces résultats.

Nous notons que l'extraction des skylines permet l'élagage de l'espace de recherche, et par la suite, elle assureungain de temps considérable lors de l'optimisation globale.

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 l'algorithme k-means. Chaque cluster possède un service représentant qui a la plus grande affinité.[13]

2.4 Étude comparative

Le tableau 2.1 présente notre étude comparative entre quelques travaux existants basés 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, sécurité, méthode ou modèle utilisé pour représenter les données, et l'efficacité :

Technique de sélection de services web												
	Multi objective						Mono objective			Mono et multi objective		
Critères	[Zhai et al, 2009]	[Maros, 2003]	[Dreo et al, 2003]	[Akbar et al, 2006]	[Alrifai et al, 2012]	[Alrifai et Risse, 2009]	[K.Deband Meyarivan, 2002]	[E. Zitizler, 2005]	[Claroeta 1,2005]	[Alrifai et al, 2010]	[Alrifai et Risse, 2009]	
Temps de réponse	Il prend en considération 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		
Qualité de services	Oui						Oui			Oui		
Technique utilisé pour la sélection services web	-Un algorithme à base de MIP -la technique de programmation entière -Recherches heuristique						Le méta heuristique (l'algorithme NSGAII)			Algorithme Skyline et l'algorithme de clustering		
Préférences d'utilisateur	Oui						Oui			Oui		
Scalabilité	Oui						Oui			Oui		
La sécurité	Non						Non			Non		
Méthode ou modèle utilisé pour représenter les données	Un modèle de graphe						Un modèle de graphe			Modèle hiérarchique des skylines		
Efficacité	Oui						Oui			Oui		

FIG. 2.3 : Etude 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 trois travaux dans l'approche mono objective, six travaux dans l'approche multi objective, et deux travaux dans l'approche mono et multi objective. Le critère du temps d'exécution est pris en compte dans la plupart de ces travaux. Presque tous les travaux étudiés dans cette étude prennent en considération : la qualité de service et la préférence d'utilisateur.

À partir de cette étude comparative de différentes approches de sélection de services web ; nous déduisons que le temps d'exécution est manquée dans la plupart des solutions proposée. Ainsi, le nombre croissant des services web nécessitent une méthode de traitement efficace. À partir de ces deux dernières observations, nous présentons dans notre travail une solution parallèle pour la composition de services web basée sur MapReduce.

la catégorie de notre proposition est sélection hybride.

2.5 Conclusion

Dans ce chapitre, nous avons présenté les propriétés non fonctionnelles des services web. Également, nous avons donné un aperçu des approches existantes dans la littérature pour la sélection de services web àbase de QoS.

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

Chapitre 3

Conception

3.1 Introduction

Avec la croissance exponentielle en nombre et en fonctionnalités des services web sur Internet, l'utilisateur est face à la difficulté de choisir entre un nombre important de services retournés par sa requête. Cependant, ces services diffèrent dans leurs qualités de services (QoS).Choisir les meilleurs services souhaités par le client joue un rôle efficace dans la création d'un service composite.

De plus, il est difficile de satisfaire l'exigence complexe d'un utilisateur par juste un seul service web individuel ou atomique, ce qui a conduit les concepteurs à composer les services web existants.

Ce chapitre consiste à proposer une solution effective pour la composition des services web en fonction des préférences de satisfaction du consommateur.

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 du calcul skyline pour choisir les meilleurs services.

3.2 MapReduce

3.2.1 Définition MapReduce

MapReduce est un modèle de programmation qui permet d'effectuer un traitement parallèle et distribué sur d'énormes ensembles de données aux seins d'un cluster de nœuds. Il est lancé par Google en 2004. Il est conçu pour la scalabilité et la tolérance aux pannes. Un tel modèle est adopté via la mise en œuvre de Hadoop. 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. [21]

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 cle /valeur. Elle va regrouper les données en les associant à des cles, choisies de telle sorte que les couples cle/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 cles distinctes produite par l'opération MAP. Au terme de l'opération REDUCE, on aura un résultat pour chacune des cles distinctes. Ici, on attribuera à chacune des machines du cluster une des cles uniques produites par MAP, en lui donnant la

liste des valeurs associées à la cle. Chacune des machines effectuera alors l'opération REDUCE pour cette cle. La fonction Reduce s'écrit de la manière suivante : Reduce (clé2, List (valeur2)) → List (valeur2) .[16]

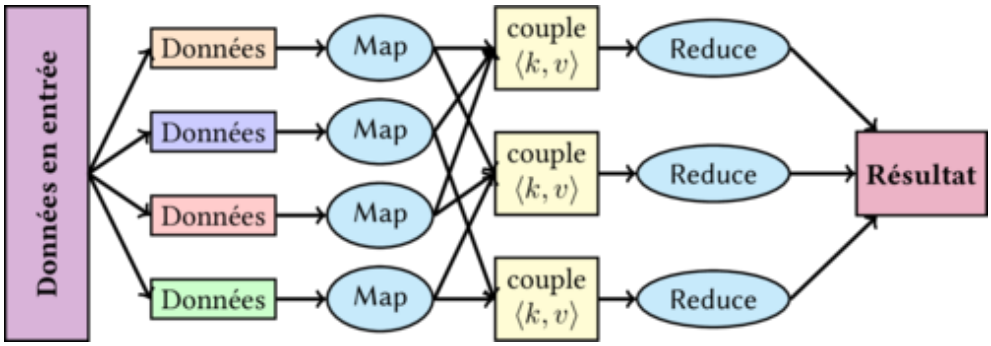


FIG. 3.1 : MapReduce.

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 :
Deer Bear River
Car River Car
Deer Car Bear.

Dans cet exemple, la clé d'entrée correspond au numéro de ligne dans le fichier et tous les mots sont comptabilisés. Le résultat de la fonction Map est donné ci-dessous. 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 fourniespar le Framework Hadoop, donc, il n'a pas à les implémenter. Après l'exécution des fonctions shuffle et sort l'appel de la fonction Reduce, le résultat de l'exemple est le suivant :

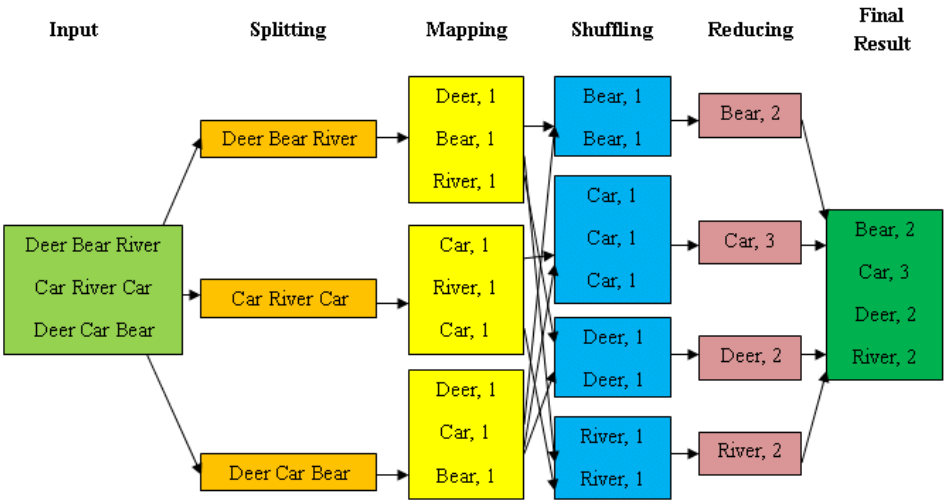


FIG. 3.2 : Exemple d'un programme mapreduce.

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 sélectionner les meilleurs services. 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 Définition de calcul Skyline

Le traitement Skyline dans des environnements informatiques distribués et parallèles a été proposé par de nombreux travaux. 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.

Le calcul Skyline a attiré beaucoup d'attention au cours des dernières années. Il a été largement appliqué dans les environnements distribués et mobiles. 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. [22]

3.2.3 Définition de la 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 (1) sur chaque dimension di $\in D$, pi $>=$ qi ; et (2) sur au moins une

dimension $dj \in D, \mathrm{pj} > \mathrm{qj}$, où P représente l'ensemble de données sur l'espace de dimensions D.

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 .[26]

Exemple

Dans cet exemple, on suppose qu'un utilisateur s'intéresse au service web de haute fiabilité et disponibilité, 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 deux services web.

	Fiabilité	Disponibilité	Temps de réponse	Cout
Sw1	0.3	0.1	0.6	0.8
Sw2	0.4	0.7	0.3	0.5

Tab. 3.1 : Montre les valeurs de critères de QoS de deux services web.

Le service web Sw2 domine le service web Sw1 pour les critères de QoS « fiabilité, disponibilité et coût ».

Le service web Sw1 domine le service web Sw2 pour le critère de QoS « temps de réponse ».

Donc : le service web Sw2 domine le service web Sw1 et le service Sw1 est éliminé.

3.2.4 Algorithme BNL

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.

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 ← supprimer).
2. **p :** domine n'importe quel point dans la liste (p ← inséré, tous les points dominés par p ← supprimer).
3. **p :** ni dominé, ni domine n'importe quel point dans la liste (p ← inséré, 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.

3.3 Présentation de l'approche proposée

3.3.1 L'architecture globale

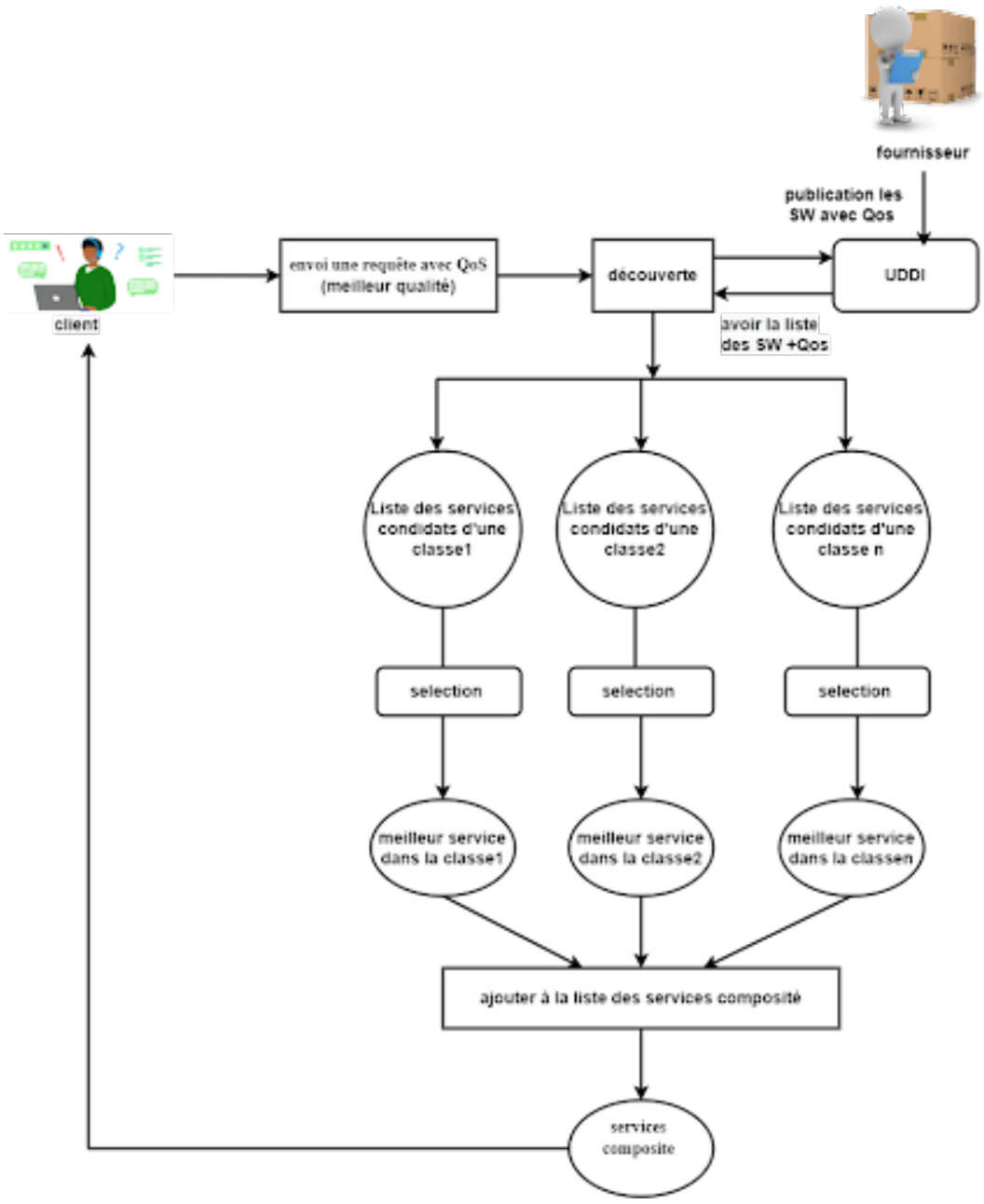


FIG. 3.3 : Architecture de l'approche proposée.

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 classés en plusieurs listes des services candidats. Pour chaque liste des services

candidats, on a sélectionné le meilleur service pour obtenir à la fin un service composite pour satisfaire le besoin du client.

Dans l'étape de sélection : à 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 la fonction Combiner en utilisant l'algorithme BNL de manière parallèle. Ce type de filtrage vise à garder seulement les services web qui ont des meilleures qualités.

Les services web générés par tous les serveurs esclaves sont fusionnés et intégrés par le Reducer pour faire le dernier filtrage.

La liste des services web obtenus par ce filtrage sont des services web incomparables et qui possèdent des meilleures qualités.

3.3.2 L'architecture détaillée

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

Client :

souhaite d'avoir différents services 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 les différentes listes des services web disponibles avec leurs propriétés non fonctionnelles (QoS) à partir du registre UDDI.
3. Passer ces listes au composant de sélection pour choisir les services composites.

Registre UDDI :

il a pour but de permettre d'automatiser les communications entre fournisseurs et clients et de gérer les métadonnées des services (QoS). 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).

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 réalisé 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

L'opération sélection indiqué dans la figure 3.3 est détaillé dans ce schéma ci-dessous :

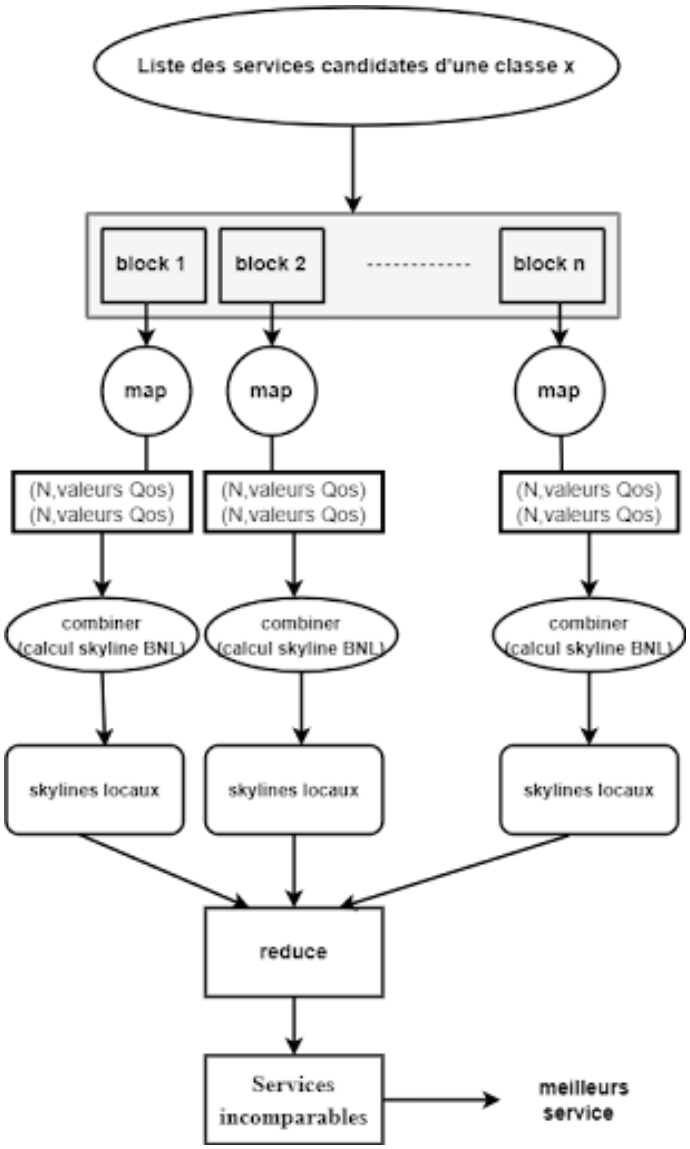


FIG. 3.4 : L'étape de sélection.

Map :

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) → (num service/les valeurs de QoS)

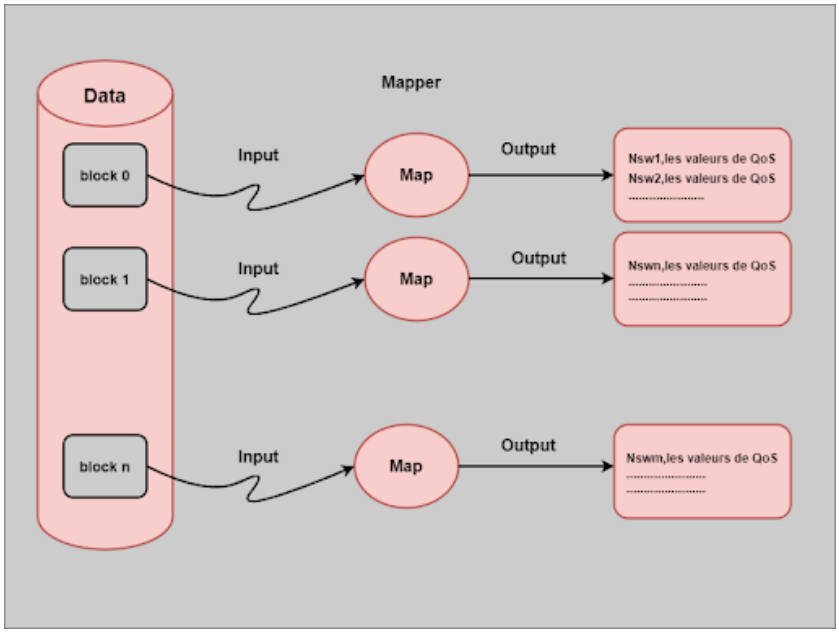


FIG. 3.5 : Fonctionnement de Mapper.

Combiner :

- 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.

La comparaison est basée sur les valeurs de qualité de service en termes de préférence d'un client.

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.

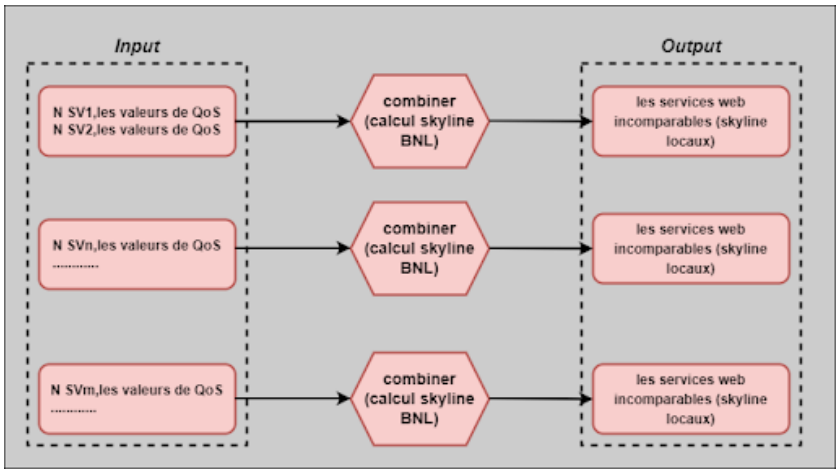


FIG. 3.6 : Fonctionnement de Combiner.

Reducer :

- Les 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.
- La sortie, on obtient un ensemble de services web final qui sont incomparables et qui possèdent des meilleures qualités.

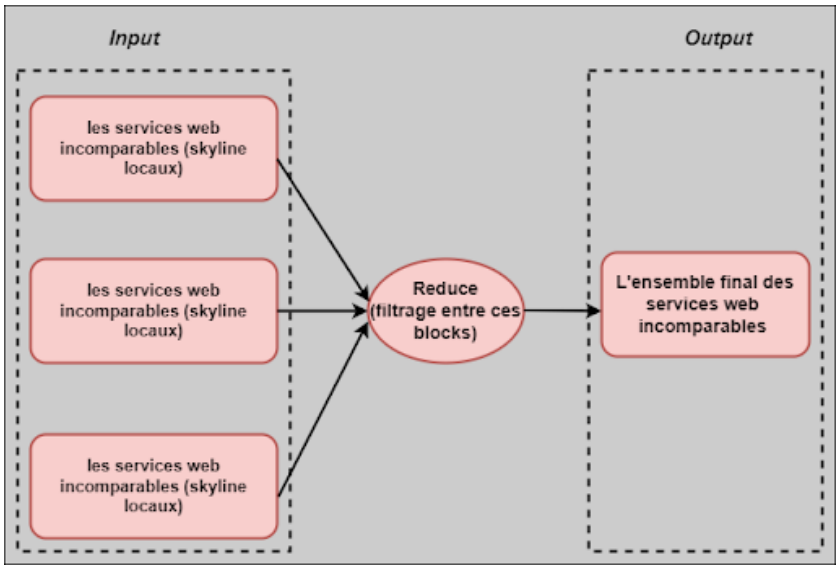


FIG. 3.7 : Fonctionnement de Reducer.

3.4 Modélisation

3.4.1 Digramme de séquence de sélection d'un service web d'une classe x

Une fois que nous disposons d'un registre UDDI dont lequel il y a eu des services web publiés par des fournisseurs, 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 l'opération de recherche, il y a eu une liste des services web similaires qui est retournée à ce module avec différents critères de QoS. À ce moment là, l'opération de MapReduce commence : il envoi des N blocks au module Map. Ce module envoyé les services web sous forme des paires (Num Service, valeurs QoS) au Combiner. Ce dernier effectue un processus de filtrage basé sur BNL pour minimiser le nombre des services web donné au Reducer. Finalement, le Reducer sélectionne le meilleur service web.

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 :

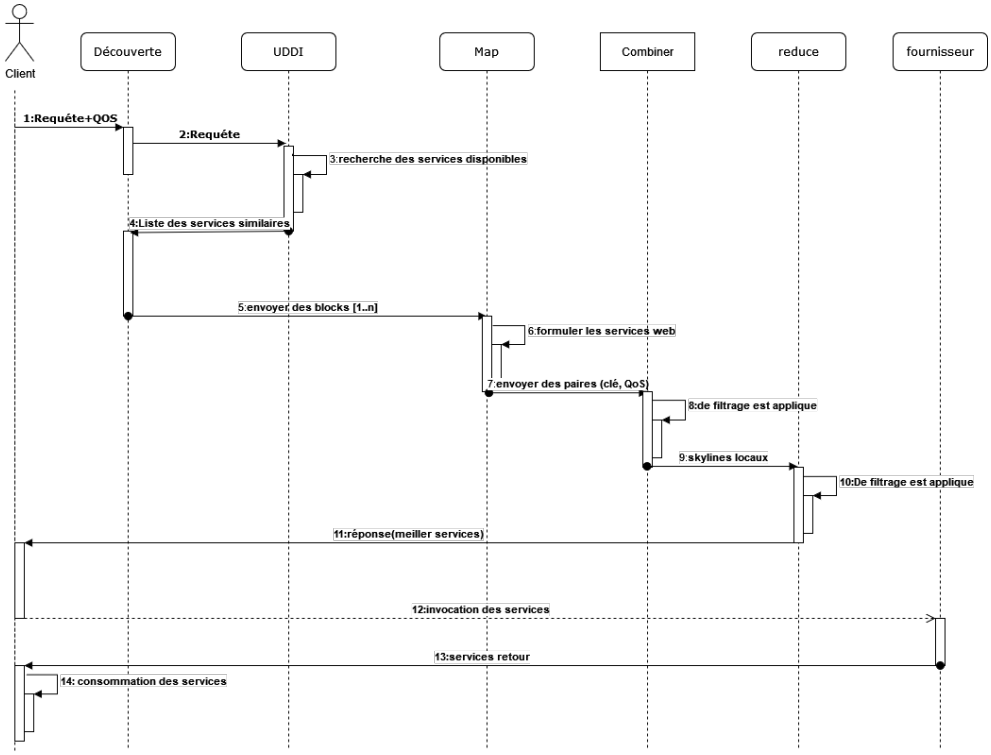


FIG. 3.8 : Diagramme de séquence de sélection de services web.

3.5 Algorithmes proposés de MapReduce

3.5.1 Algorithme de Map

Algorithm 1 Map

Input : la liste des N blocks des services candidats d'une classe X

Output : la liste des NewBlock i sous forme paires (num sw, QoS)

```
1 foreach block i ∈ ( 1..N ) do
2 |  NewBlock i ← Map ( block i ) /* Mapper */
3 end
```

3.5.2 Algorithme de Combiner

Algorithm 2 Combiner

Input : la liste des NewBlock i sous forme paires (num sw, QoS)

Output : skylines locaux des services

```
4 foreach Newblock i ∈ ( 1..N ) do
5 |  LSkyBlock i ← BNL ( Newblock i ) /* Combiner → obtenir les skylines
  |     locaux pour chaque block */
6 end
```

3.5.3 Algorithme de Reducer

Algorithm 3 Reducer

Input : skylines locaux des services

Output : le meilleur service dans la classe X

```
7 GSkylines ← BNL ( LSkyBlock i , .... LSkyBlock N ) /* Reducer */
8  Afficher(GSkylines) /* choisir le meilleur service dans la classe X */
```

3.6 Conclusion

Dans ce chapitre nous avons présenté une méthode de sélection des services web composites on se basant sur des critères de qualités de service (QoS). Nous avons proposé une méthode hybride composée de MapReduce et le calcul Skyline. Notre proposition vise à accélérer le calcul de filtrage des services web disponibles selon le besoin de l'utilisateur pour obtenir des services web composites avec une bonne qualité.

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

Chapitre 4

Implémentation et analyses

4.1 Introduction

Dans ce chapitre nous présentons 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 I5 doté de windows 10 (64bits) qui est décrit dans la figure suivante :

Fig. 4.1 : Environnement logiciel utilisé.

4.3 Langages Python

Python est le langage de programmation le plus utilisé dans le domaine du machine learning, du big data et data science. Le langage de programmation Python est un langage de programmation open source créé par le programmeur Guido van Rossum en 1991. Il tire son nom de l'émission Monty Python's Flying Circus. Depuis quelques années ce langage de programmation s'est hissé parmi les plus utilisés dans le domaine du développement de logiciels, de gestion d'infrastructure et d'analyse de données. Il s'agit d'un élément moteur de l'explosion du Big Data.

4.4 Hadoop (High-availability distributed object-oriented platform)

4.4.1 Définition

Hadoop est un projet open source Apache Software Foundation (ASF) écrit en Java qui fournit une infrastructure rentable et évolutive pour le traitement distribué de grands ensembles de données sur des clusters de marchandises.

Hadoop a été ”inspiré” par Google File System GFS et l'environnement informatique distribué MapReduce de Google.

Initialement, il a commencé comme un projet de moteur de recherche Nutch distribué et nommé par le développeur Doug Cutting d'après l'éléphant jouet de son fils (White 2009). Il a été utilisé avec succès pour traiter des problèmes hautement distribuables sur une grande quantité d'ensembles de données en utilisant des serveurs couramment disponibles dans un très grand cluster. Le projet Hadoop actuel se compose de trois modules principaux, à savoir le système de fichiers distribué appelé Hadoop Distributed File System (HDFS), le moteur MapReduce et YetAnother Resource Negotiator (YARN).[15]

4.4.2 Caractéristiques essentielles d'Hadoop

Hadoop est une plate-forme logicielle de stockage et d'analyse de données, dont on peut lister les propriétés suivantes :

- La plate-forme possède un système de fichiers distribué très facilement extensible. Hadoop gère seul la distribution et le stockage des données sur ses différents nœuds. Pour augmenter la capacité de stockage il suffit d'ajouter des nœuds de données dans la plate-forme.
- Les codes des traitements sont routés jusqu'aux données. Cette stratégie est la plus efficace pour de grosses volumétries de données stockées sur des machines standard reliées par des réseaux standard. Les nœuds de données se transforment donc en nœuds de calculs, et par conséquent, augmenter le nombre de nœuds de données pour accroître la capacité de stockage et donc augmente la capacité de traitement.
- Des mécanismes de tolérance aux pannes sont intégrés à la plate-forme. Hadoop étant conçu pour fonctionner sur du matériel standard (bon marché). Des pannes fréquentes sont supposées inéluctables, et les données sont répliquées sur plusieurs nœuds afin d'être toujours accessibles. Quand un réplicat disparaît (suite à une panne), ses copies sont à nouveau répliquées pour maintenir un bon taux de réplication. De même, les tâches de traitements exécutées sur les nœuds de données sont monitorées et relancées sur le nœud d'un autre réplicat si une panne survient. L'utilisateur n'a pas à se soucier de la tolérance aux pannes.
- Un paradigme de programmation Map-Reduce est intégré à la plate-forme. Ce paradigme convient à la récupération et au filtrage de données stockées dans l'ensemble des nœuds.

4.4.3 Principes et pile logicielle d'Hadoop

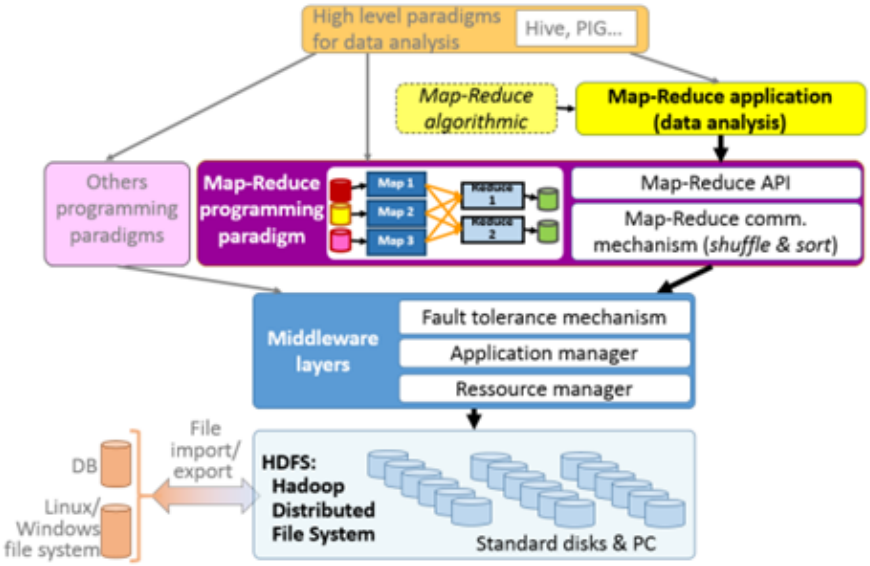


FIG. 4.2 : Pile logicielle simplifiée de l'écosystème Hadoop.

La figure1 montre la pile logicielle simplifiée d'Hadoop et d'une partie de son écosystème.

A la base se trouve un système de fichiers distribué appelé HDFS (Hadoop Distributed File System), sur lequel s'appuie un ensemble de composants permettant d'identifier puis de choisir les ressources sur lesquelles déployer les processus d'une application, de gérer des applications distribuées, et de réagir en cas défaillance pour être tolérant aux pannes. Cet ensemble de composants et de fonctionnalités forment un couche de middleware quasi-invisible à l'utilisateur et qui a beaucoup évolué entre les versions 1 et 2 d'Hadoop (la version 2 est aussi appelée YARN : YetAnother Resource Negociator).

Le monitoring des ressources a été renforcé, et la supervision des applications Map-Reduce en cours d'exécution a cessé d'être un goulot d'étranglement qui limitait le passage à l'échelle .

L'utilisateur pouvant se contenter de développer la classe Java d'une tâche Map et celle d'une tâche Reduce. Le mécanisme de redistribution des sorties des tâches Map vers les entrées des tâches Reduce, qui est un composant clé du Map-Reduce d'Hadoop, est également fourni par la pile logicielle et reste une boite noire pour l'utilisateur.

Il est ainsi possible de développer facilement des applications d'analyse de données basées sur une algorithmique Map-Reduce , avec peu de connaissances d'informatique distribuée.

Toutefois, Hadoop peut supporter d'autres types de paradigmes de programmation distribuée, dont les tâches seront déployées en se basant toujours sur le gestionnaire de ressources et le gestionnaire d'application du milieu de pile. Enfin, l'écosystème d'Hadoop est très riche, et on compte notamment des applications de plus haut niveau permettant par exemple de traiter des données dans un formalisme proche de SQL (comme dans une

base de données relationnelle), et des outils permettant d'importer des données extérieures dans le système de fichiers distribué d'Hadoop ou d'exporter des données d'Hadoop vers l'extérieur (en bas à gauche de la figure 1).[24]

4.5 Docker

FIG. 4.3 : Docker.

Docker est un projet open source (Apache 2.0) écrit en GO et hébergé sur GitHub :https ://github.com/docker(https ://github.com/docker).

Initialement il est 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 run C).
- 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.[18]

4.6 Visual Studio

Visual Studio Code est un éditeur de code open-source développé par Microsoft, et il est disponible pour Windows, macOS et Linux. Il est livré avec un support intégré pour un très grand nombre de langages (tels que C++, C#, Java, Python, PHP, Go) grâce à des extensions. Il supporte l'auto-complétion, la coloration syntaxique, le débogage, et les commandes git . [25]

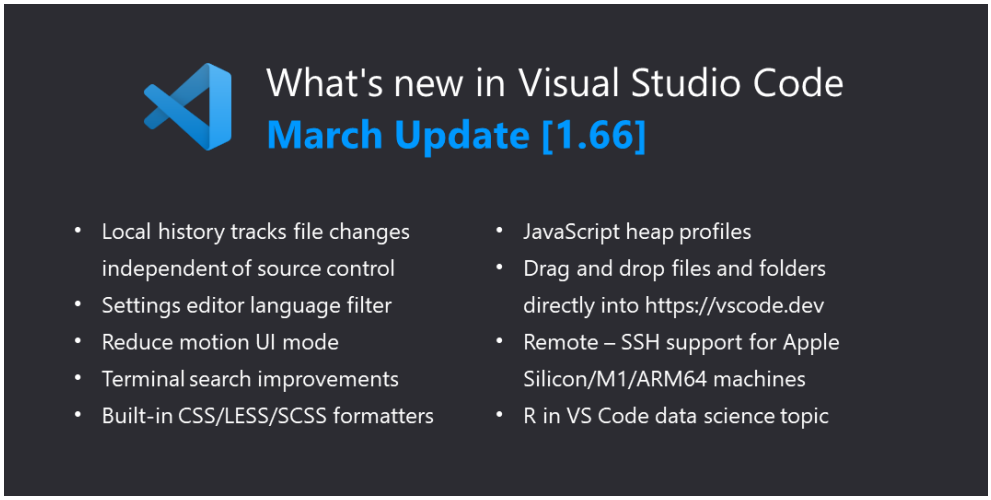


Fig. 4.4 : Visual Studio

4.7 XAMPP

XAMPP est un synonyme de multi-plateforme (X), Apache (A), MySQL (M), PHP (P) et Perl (P). C'est une distribution Apache libres simple et légère qui permet aux développeurs de créer facilement un serveur web local à des fins de test et un serveur FTP.

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.[23]

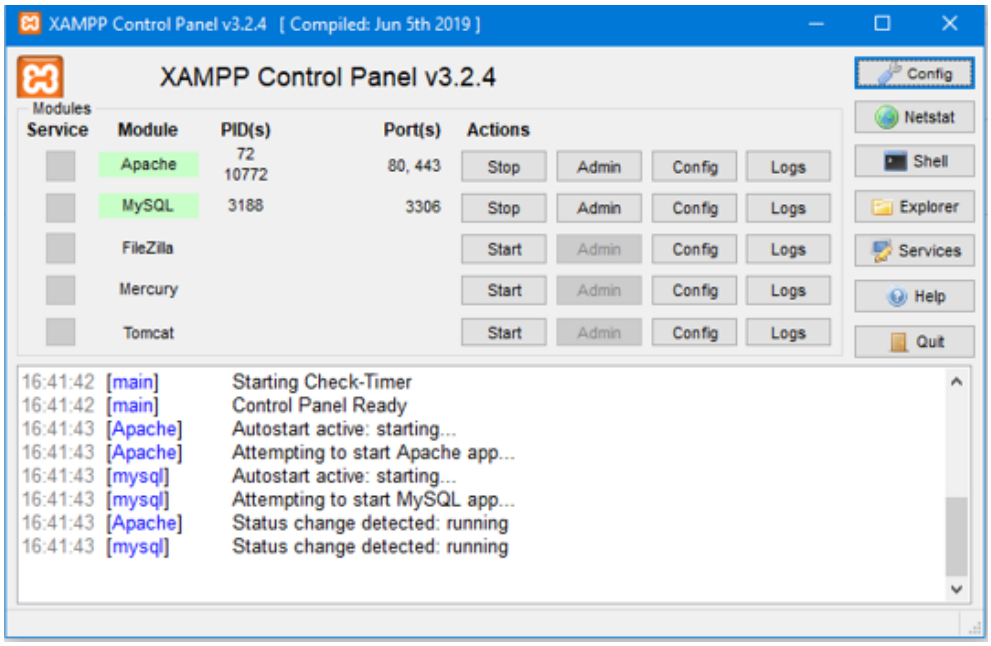


Fig. 4.5 : Xampp

4.8 Analyse de scénario

Nous expliquons notre travail en présentant un scénario simple pour la réservation d'un voyage : L'utilisateur veut faire les réservations nécessaires pour voyager. Pour répondre à cette requête, l'utilisateur doit réserver les éléments suivants :

- Réserver un avion
- Réserver une voiture pour aller à l'hôtel
- Réserver dans un hôtel

En revanche, plusieurs services web qui fournissent ces besoins sont disponibles avec des différentes qualités (QoS). L'objectif de ce scénario est de fournir aux utilisateurs un service composite répondant à leurs besoins.

Dans notre simulation, la requête de l'utilisateur est de rechercher des services web qui offre ces besoins (réservation d'un avion, réservation d'une voiture, réservation dans un hôtel). Ces services doivent avoir une disponibilité et une fiabilité élevée, un temps de réponse et un coût de service plus faibles possibles.

Pour répondre à la requête de l'utilisateur, notre solution proposée est basé sur MapReduce Skyline sur Hadoop pour choisir un meilleur service composite.

4.9 Présentation des interfaces graphiques

4.9.1 Interface pour les services de vol

Cette interface permet d'afficher tous les services web disponibles pour la réservation de vol avec leurs qualités : temps de réponse, le coût, fiabilité, et la disponibilité.

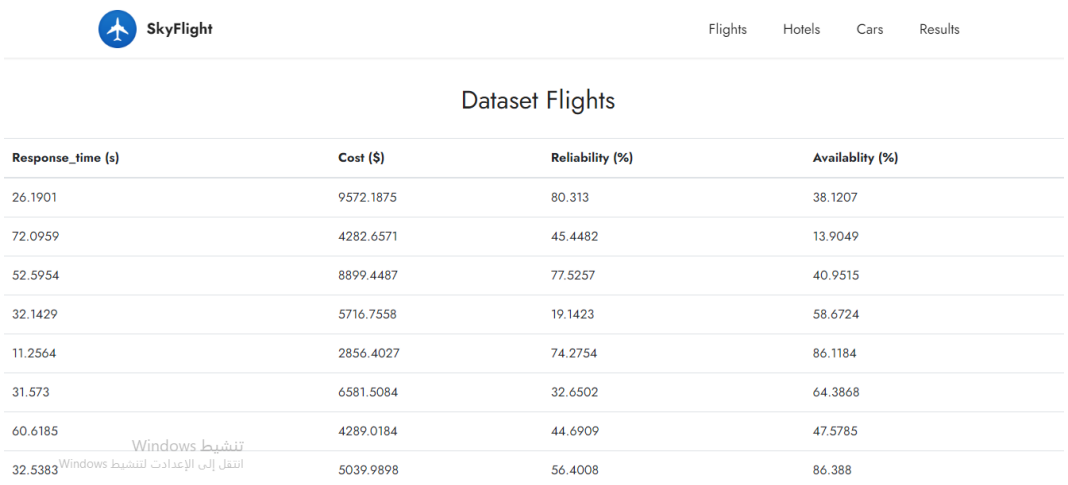


FIG. 4.6 : Interface pour les services de vol.

4.9.2 Interface pour les services d'hôtel

Cette interface permet d'afficher tous les services web disponibles pour la réservation dans un hôtel avec leurs qualités : temps de réponse, le coût, fiabilité, et la disponibilité .

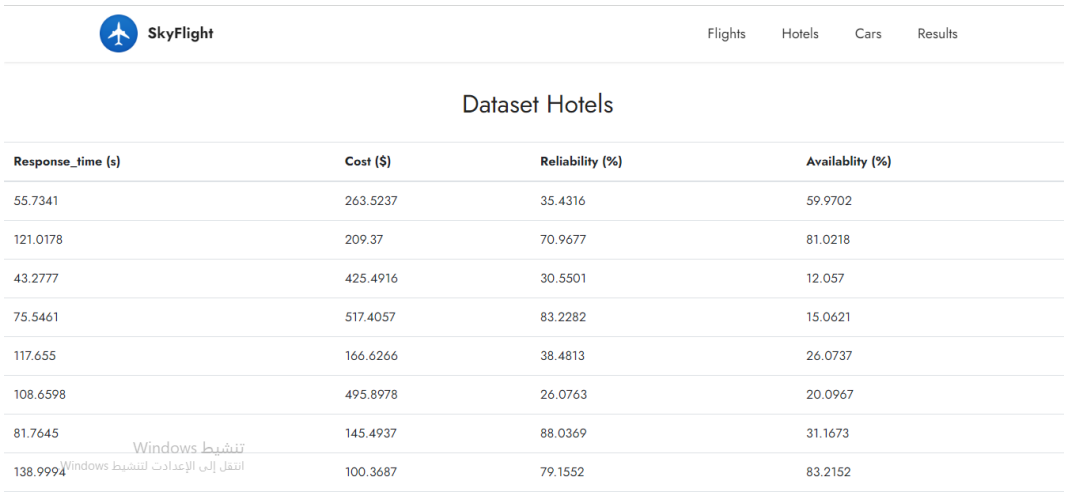
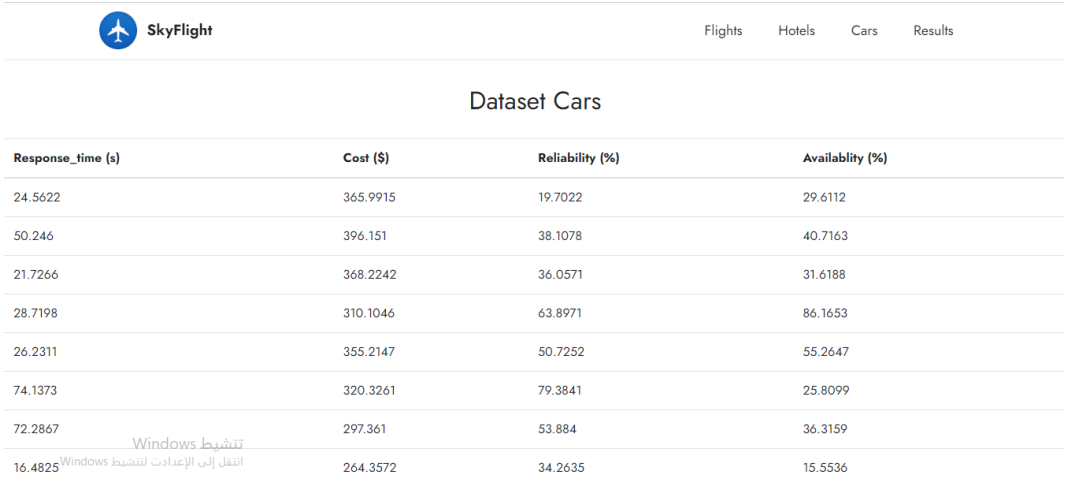


FIG. 4.7 : Interface pour les services d'hôtel.

4.9.3 Interface pour les services de voiture

Cette interface permet d'afficher tous les services web disponibles pour la réservation d'une voiture avec leurs qualités : temps de réponse, le coût, fiabilité, et la disponibilité.



Response_time (s)	Cost ($)	Reliability (%)	Availablity (%)
24.5622	365.9915	19.7022	29.6112
50.246	396.151	38.1078	40.7163
21.7266	368.2242	36.0571	31.6188
28.7198	310.1046	63.8971	86.1653
26.2311	355.2147	50.7252	55.2647
74.1373	320.3261	79.3841	25.8099
72.2867	297.361	53.884	36.3159
16.4825	264.3572	34.2635	15.5536

FIG. 4.8 : Interface pour les services d'une voiture.

4.9.4 La génération aléatoire des services web

Cette interface permet d'afficher une partie du code liée à la génération aléatoire des services avec leurs valeurs de qualités.

```python
import random
import csv

def generate_data():

  def flights():

      response_time = round(random.uniform(10, 80),4)
      cost = round(random.uniform(100,10000),4)
      reliability = round(random.uniform(10, 90),4)
      availablity = round(random.uniform(10, 90),4)

      return   response_time, cost,reliability, availablity
  header = ['response_time', 'cost', 'reliability', 'availablity']

  with open('flights.csv', 'w', encoding='UTF8') as f:
      writer = csv.writer(f)
```

FIG. 4.9 : la génération aléatoire des services.

4.9.5 La version d'Hadoop

Cette fenêtre permet de donner une vue de la version d'hadoop installée sur l'application Docker, sur laquelle nous allons travailler et tester notre application.

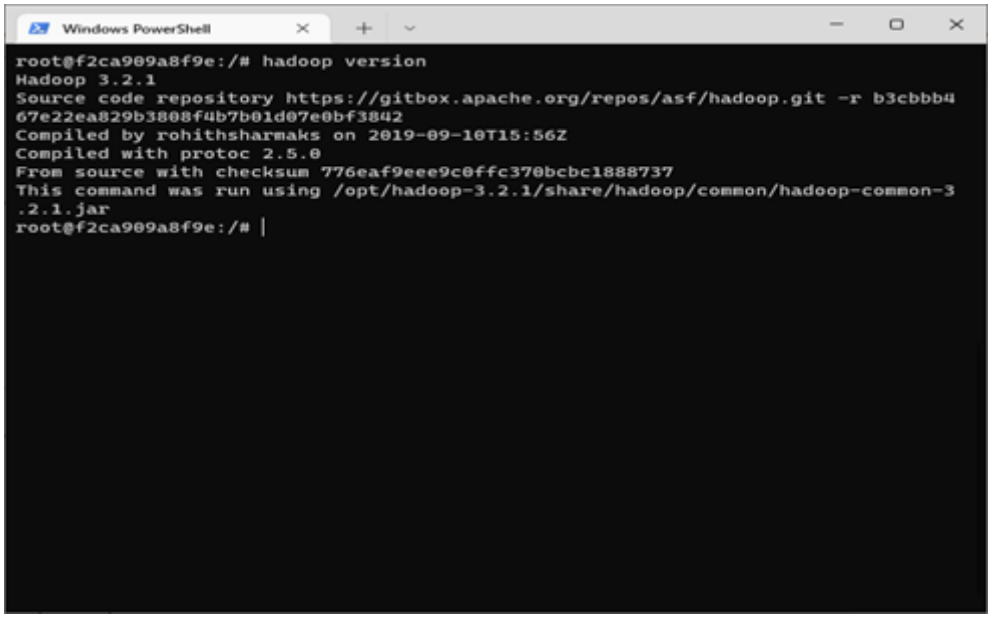


Fig. 4.10 : la version d'Hadoop.

4.9.6 L'interface principale de l'application

L'interface principale de l'application contient un ensemble des tâches liées au contrôle de l'application et à l'exécution de codes et il montre également les résultats obtenus.

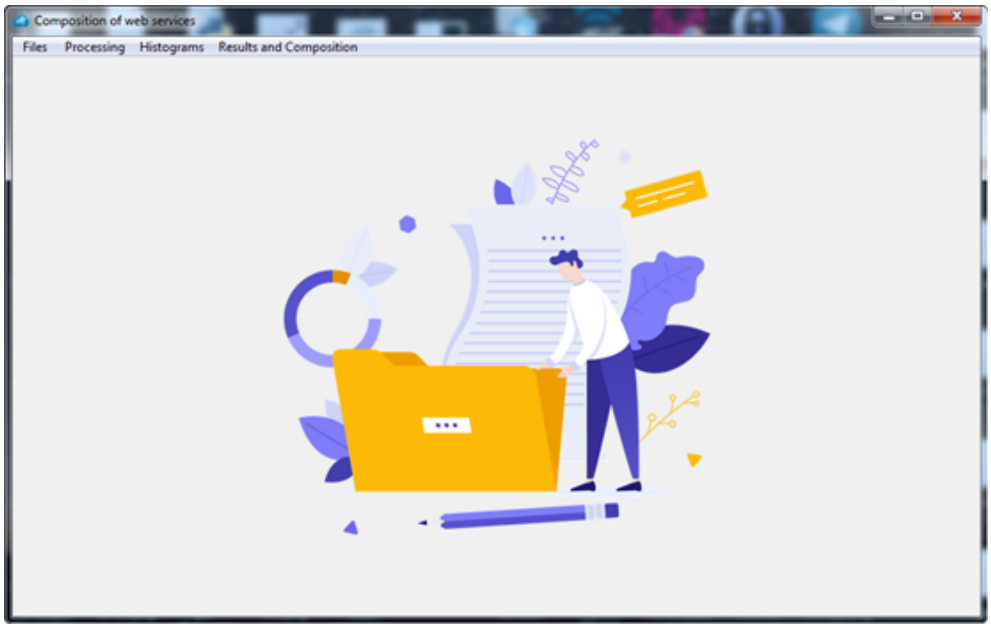


Fig. 4.11 : L'interface principale de l'application.

4.9.7 Affichage du numéro du meilleur service de réservation de vol

Affichage du numéro du meilleur service de réservation de vol après l'exécution de processus de traitement par MapReduce Skyline.

Fig. 4.12 : Affichage du numéro du meilleur service de réservation de vol.

4.9.8 Affichage du numéro du meilleur service de réservation dans un hôtel

Affichage du numéro du meilleur service de réservation dans un hôtel après l'exécution de processus de traitement par MapReduce Skyline.

Fig. 4.13 : Affichage du numéro du meilleur service de réservation dans un hôtel.

4.9.9 Affichage du numéro du meilleur service de réservation d'une voiture

Affichage du numéro du meilleur service de réservation d'une voiture après l'exécution de processus de traitement par MapReduce Skyline.

FIG. 4.14 : Affichage du numéro du meilleur service de réservation d'une voiture .

4.9.10 Présentation du meilleur service composite

Après l'exécution de processus de sélection d'un meilleur service pour chaque classe, nous obtenons un meilleur service composite en termes de vol, hôtel et de voiture.

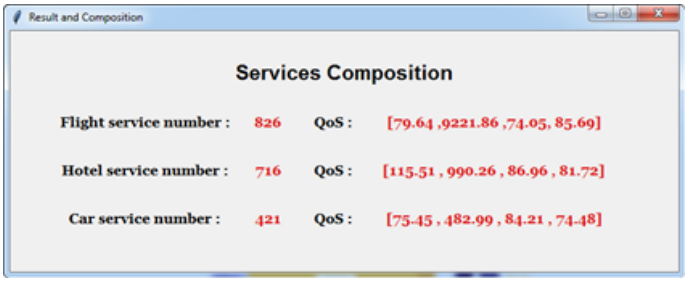


FIG. 4.15 : Présentation du meilleur service composite.

4.9.11 Interface Docker

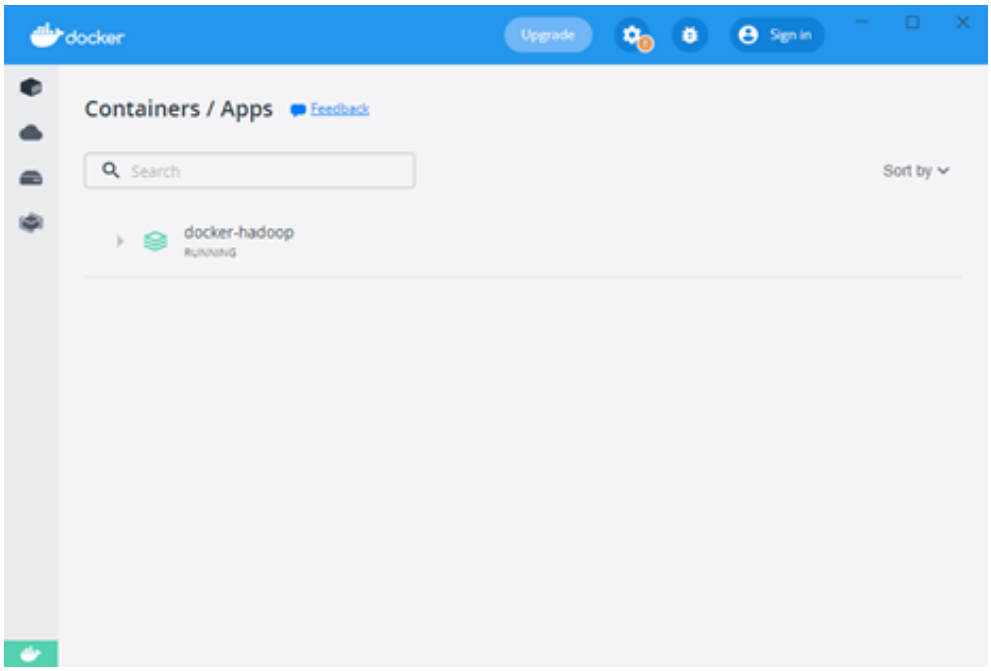


FIG. 4.16 : Interface Docker.

Ici, cette fenêtre permet d'afficher l'installation d'hadoop.

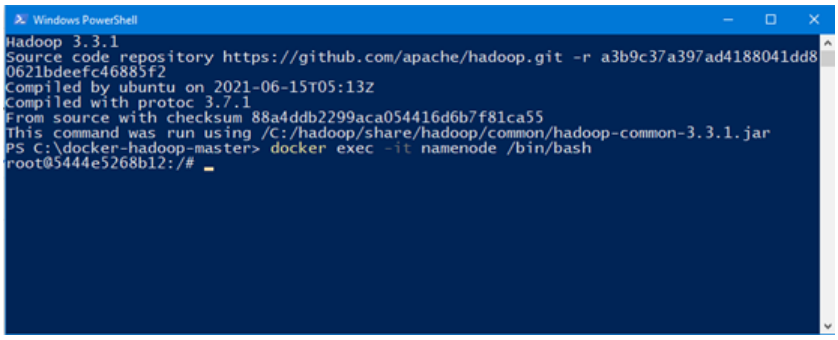


FIG. 4.17 : la version Hadoop installée sur le Docker .

Cette image montre la version Hadoop installée sur le Docker pour la version 3.3.1. Nous créons un répertoire dans HDFS avec le même nom et copions tout le contenu dans le répertoire 'input' créé. Puis, ils sont distribués à 3 Data Nodes comme l'indiqué les trois fenêtres suivantes :

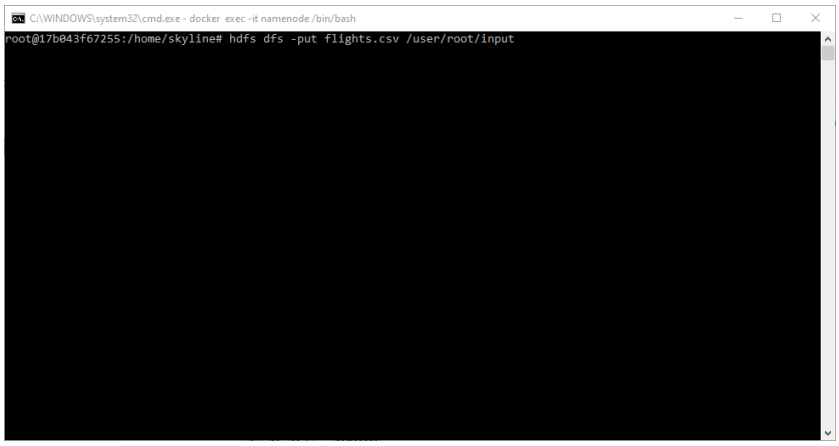


FIG. 4.18 : Copy dadaset flight in hdfs input.

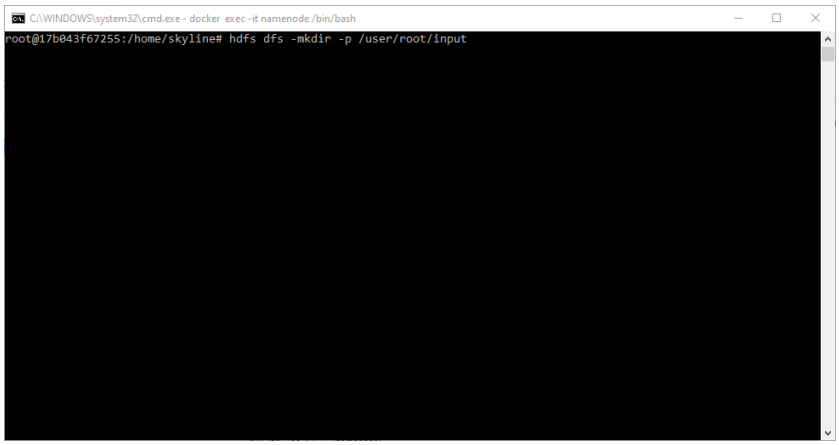


FIG. 4.19 : Create input .

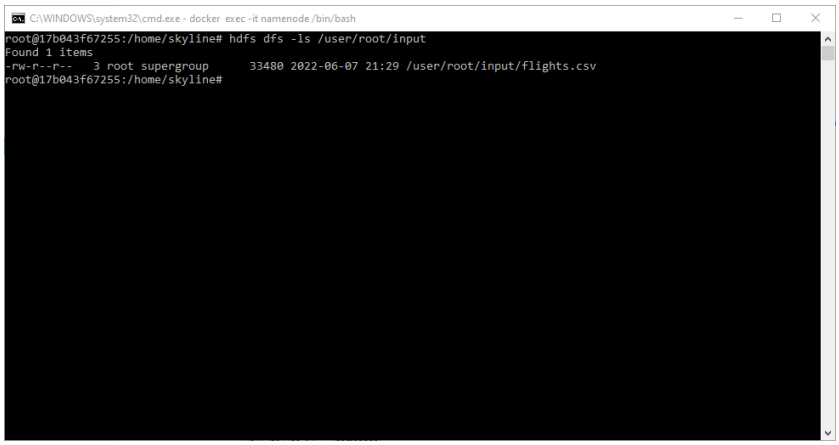


FIG. 4.20 : Copy dataset ls.

4.9.12 Le résultat de MapReduce

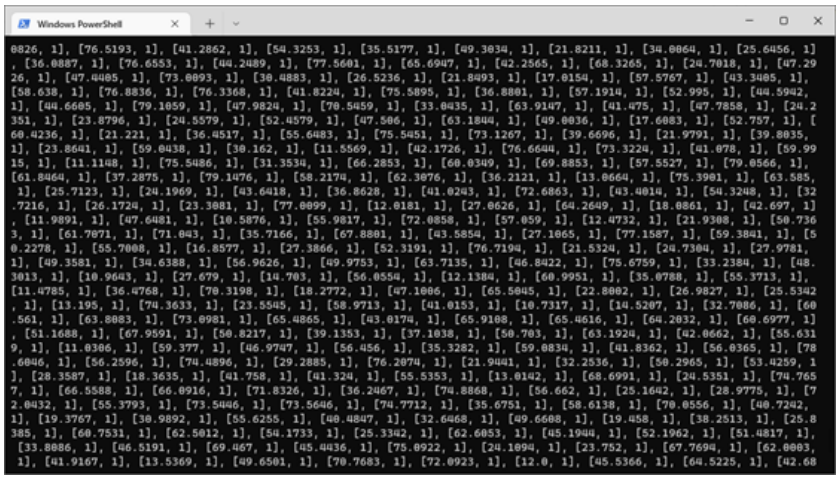


Fig. 4.21 : Le résultat de MapReduce.

Le résultat d'implémentation sous Hadoop, et les résultats montrent les pairs sous forme (clé-valeur).

Le processus de MapReduce pour la réservation de vol

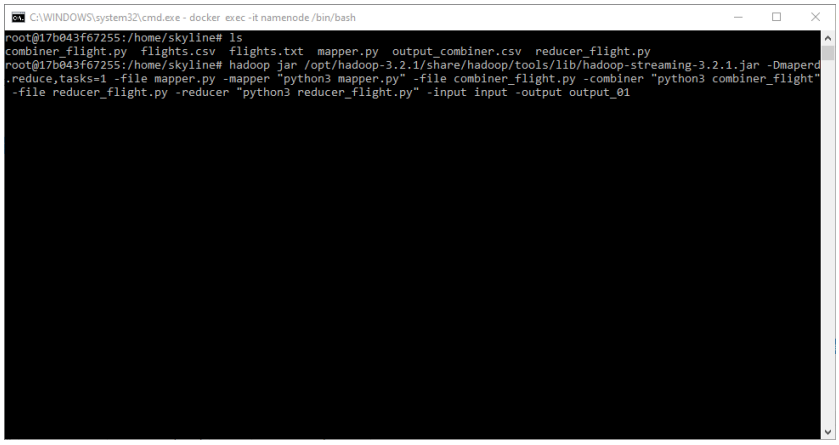


Fig. 4.22 : Le processus de MapReduce pour la réservation de vol.

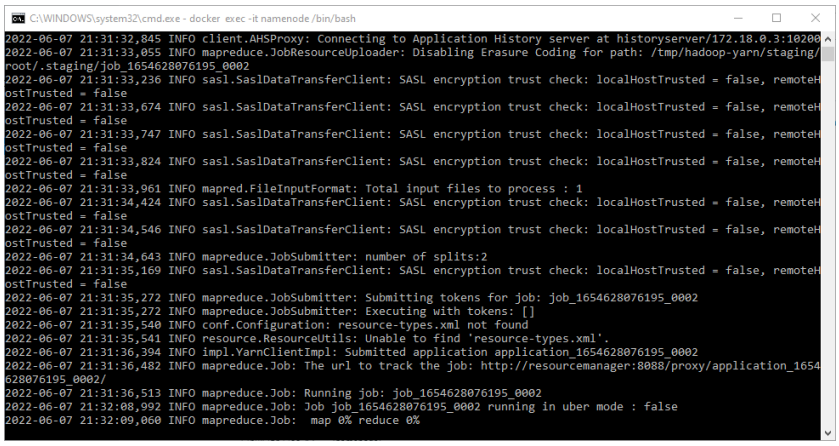


Fig. 4.23 : le processus de MapReduce sous Hadoop.

Ces deux images montrent le processus de MapReduce sous Hadoop pour traiter ces données.

4.9.13 Le résultat de Combiner

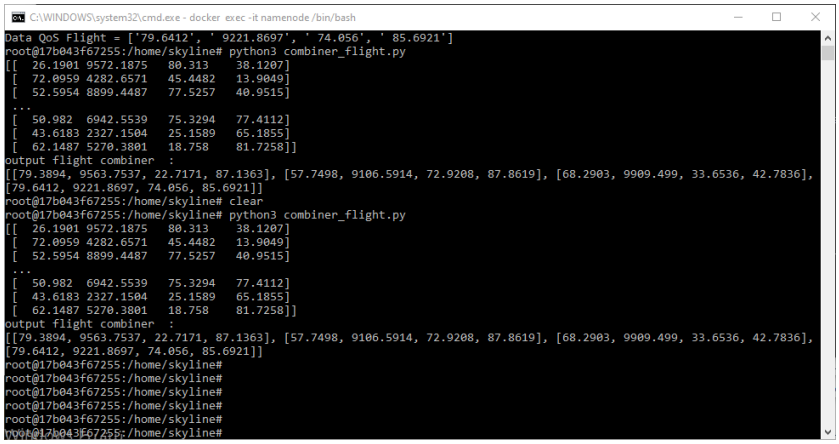


Fig. 4.24 : Le résultat de Combiner.

Cette fenêtre montre le résultat de la fonction Combiner sous Hadoop, et les résultats apparaissent sous forme des groupes de services avec la meilleure qualité, non comparables au niveau local.

4.9.14 Le résultat de Reduce

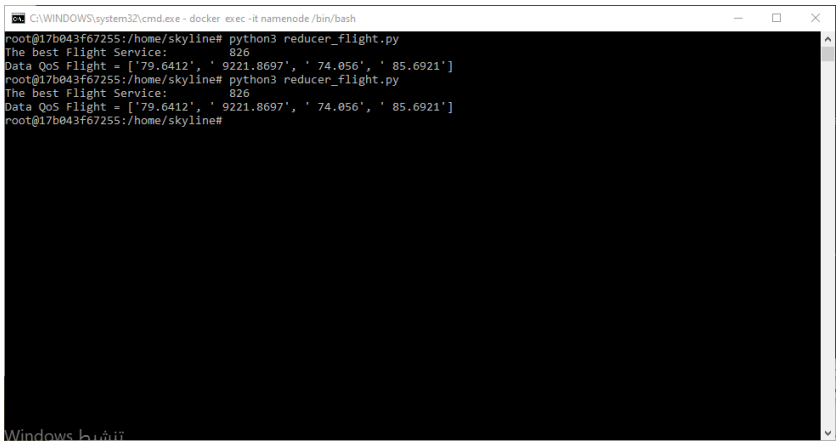


FIG. 4.25 : Le résultat de Reduce.

Cette fenêtre montre le résultat de la fonction Reduce sous Hadoop, et les résultats apparaissent sous forme des groupes de services avec la meilleure qualité, non comparables au niveau global.

4.10 Conclusion

Ce chapitre a été consacré à la présentation du l'environnement de développement en termes de différents outils et les technologies utilisés. Nous avons simulé la solution proposée pour résoudre le problème de composition de services web. Comme nous l'avons montré dans ce chapitre l'efficacité de notre proposition pour minimiser le temps de découverte d'un service composite de manière distribué et parallèles pour gérer des grand nombre des services web disponibles.

Conclusion et perspectives

Conclusion générale

Le principe de l'approche service web est de transformer le web en un dispositif distribué pour échanger des messages, où les services web peuvent interagir d'une manière intelligente. Actuellement, de nombreux services web, avec des fonctionnalités similaires sont fournis par des fournisseurs concurrents, et de ce fait les utilisateurs ont besoins d'approches efficaces pour la sélection des services.

La sélection des services web est l'une des problématiques les plus importantes dans le processus de composition. Au cours de ce mémoire, nous nous sommes intéressés au problème de sélection des services web dans une composition sur la base des besoins non fonctionnels (QoS). L'objectif de ce mémoire est de proposer des solutions qui aident à choisir les services de meilleures qualités pour participer dans le processus de composition.

L'idée principale de ce mémoire est consacré à la Framework MapReduce Skyline pour accélérer le calcul et introduire un parallélisme pour traiter de grandes quantités des services web disponibles. Par conséquent, cette solution sélectionne les services web optimales pour construire un service composite en réduisant le temps et le coût de calcul.

Nous avons aussi validé et mis en œuvre nos propositions à travers le développement sous Hadoop. Dans ce travail, on n'a pas utilisé de vrai service web et on a supposé qu'on a un grand nombre des services web avec leurs QoS. Les expérimentations réalisées à travers Hadoop nous ont permis de constater que la composition des services basée sur la QoS retourne des services qui ont des meilleures qualités et de manière très rapide.

Perspectives

Afin de compléter le travail présenté dans ce mémoire, on peut envisager quelques améliorations et travaux futurs :

- L'intégration de plusieurs services composite.
- La prise en charge de la réparation des services quand un service est inaccessible ou en panne, le système doit pouvoir le remplacer par un autre service au moment de l'exécution.
- Développer une méthode de sélection efficace pour l'ensemble filtré obtenue par la fonction Reduce.
- Expérimenter notre approche avec une large dataset des services web.
- Ajouter d'autres critères pour donner plus de choix aux clients.

Bibliographie

[1] Aouiche Ahmed et Zaidi Faycel. “Sélection des services Web dans les systèmes distribués : étude comparative”. In : (2021).

[2] Mohammad Alrifai, Dimitrios Skoutas et Thomas Risse. “Selecting skyline services for QoS-based web service composition”. In : *Proceedings of the 19th international conference on World wide web*. 2010, p. 11-20.

[3] Mohammad Alrifai et al. “A scalable approach for qos-based web service selection”. In : *International conference on service-oriented computing*. Springer. 2008, p. 190-199.

[4] Gilbert Babin et Michel Leblanc. “Les Web Services et leur impact sur le commerce B2B”. In : *CIRANO, CIRANO Burgundy Reports* (1er jan. 2003).

[5] Stephan Borzsony, Donald Kossmann et Konrad Stocker. “The skyline operator”. In : *Proceedings 17th international conference on data engineering*. IEEE. 2001, p. 421-430.

[6] Ibtissem BOUCHAKOUR. *Bouchakour Ibtissem Reconfiguration dynamique d'un service web composé basée sur les Qds*. 2018.

[7] Ibtissem Bouchakour et Rohallah Benaboud. “Reconfiguration dynamique d'un service web composé basée sur les gds”. In : (2018).

[8] Université Mohamed Chérif, Messadia Souk Ahras et Mohamed Chérif Messadia. “Selection par la composition des services web basée sur QOS avec la logique floue”. In : (), p. 61.

[9] Daniela Barreiro Claro, Patrick Albers et Jin-Kao Hao. “Selecting Web Services for Optimal Composition.” In : *SDWP@ ICWS*. 2005.

[10] HAZAR DERRADJI. *Une Approche Effective Pour La Sélection Des Services Web Composites*. 2019.

[11] Barry Douglas K. *Service-Oriented Architecture (SOA) Definition*.

[12] Johann Dréo et al. *Métaheuristiques pour l'optimisation difficile*. Eyrolles, 2003.

[13] M Feham et B Atmani. “Composition et interopération des services web sémantiques”. In : (), p. 223.

[14] Boudjelaba HAKIM. “Sélection des web services sémantiques..” Thèse de doct. Université de Béjaia-Abderrahmane Mira, 2012.

[15] Ibrahim Abaker Targio Hashem et al. “MapReduce : Review and open challenges”. In : *Scientometrics* 109.1 (oct. 2016), p. 389-422. issn : 0138-9130, 1588-2861. doi : 10.1007/s11192-016-1945-y.

[16] *implantation_du_modele_mapreduce_dans_lenvironnement_distribue_hadoop_distribution_cloudera*.

[17] Katawut KAEWBANJONG et Sarun INTAKOSUM. "QoS Attributes of Web Services : A Systematic Review and Classification". In : *Journal of Advanced Management Science* (2015), p. 194-202. ISSN : 21680787. DOI : 10.12720/joams.3.3.194-202.

[18] A LAHOUZE. *Introduction à Docker*. 2018.

[19] Mrissa MICHAEL. *M ediation S emantique Orient ee Contexte pour la Composition de Services Web*. 15 nov. 2007.

[20] Mohy-eddine MOULAY KHATIR OUSSAMA et Redouane HENAOUI. *Sélection de Service Web à base d'algorithme Mimétique*. 2013.

[21] Kiran RAVI. *MapReduce*. 28 mar. 2022.

[22] Farida RETIMA. "Gestion de contexte pour l'Internet des objets". Thèse de doct. Université Mohamed Khider Biskra, 2019.

[23] Fabrice S et Développeur web FULL. *Utiliser XAMPP pour créer un serveur de test local*. 30 nov. 2017.

[24] *Technologie de l'écosystème d'Hadoop*.

[25] *Visual Studio Code*.

[26] Akrivi VLACHOU, Christos DOULKERIDIS et Yannis KOTIDIS. "Angle-based space partitioning for efficient parallel skyline computation". In : *Proceedings of the 2008 ACM SIGMOD international conference on Management of data - SIGMOD '08*. the 2008 ACM SIGMOD international conference. Vancouver, Canada : ACM Press, 2008, p. 227. ISBN : 978-1-60558-102-6. DOI : 10.1145/1376616.1376642.