



RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE
MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA RECHERCHE SCIENTIFIQUE
Université Mohamed Khider – BISKRA
Faculté des Sciences Exactes, des Sciences de la Nature et de la Vie
Département d'informatique

Année : 2018
N° d'ordre : SIOD 14/M2/2018.

Mémoire présenté pour obtenir le diplôme de master académique en

Informatique

Option :SIOD

Titre :

*Une approche personnalisée pour la sélection de
Service Web en tenant compte des contraintes
Temps réel*

Présentée par : Okat Bilal

Date de soutenance: 24/06/2018

Composition du Jury

<i>Djerou Leila</i>	MCA	Président
Ben Seghir Nadia	MCB	Encadreur
Meklid Abdesslem	MAA	Examineur

Remerciement

Grand merci à Allah, le tout puissant qui m'a donné la force et le courage d'arriver jusque-là.

Je tiens à remercier profondément mon encadreur : Ben Seghir Nadia qui m'a encouragé à faire le maximum d'efforts dans ce travail ainsi que pour sa disponibilité.

Je remercie par la même occasion les membres de jury qui ont bien voulu accepter d'examiner et évaluer mon travail et participer à ma soutenance.

Je remercie aussi ma famille et tous les enseignants du département d'informatique pour leurs efforts afin de nous assurer la meilleure formation possible durant notre cycle d'étude.

Enfin, je remercie tous ceux qui ont contribué de près ou de loin à l'aboutissement de ce travail.

«Table des Matières »

« Introduction générale »

I. Introduction.....	1
II. Problématique :	1
III. Contribution :	2
IV. Plan du mémoire :	2

Chapitre 1 : « Les services web »

1. Introduction	4
2. Définition :	4
3. Architecture orientée service (SOA):	6
3.1 Définition de SOA :	6
3.2 Objectifs:	7
3.3 Avantages.....	7
4. Architecture des services web :	8
4.1 Architecture de reference:	8
4.1.1. Les acteurs :	9
4.2.1. Les opérations principales :	10
4.2 Architecture etendue :	11
4.3 Langages et protocoles utilisés par les services web :	12
4.3.1. Le langage XML (eXtensible Markup Language) :	12
4.3.2. La couche de Transport :	13
4.3.3. La couche de Communication (SOAP) :	13
4.3.4. La couche de Description (WSDL) :	14
4.3.5. La couche de Découverte et de Publication (UDDI) :	19
4.3.6. La couches transversales (Sécurité, Transactions, QoS) :	22
5. Les avantages et inconvénients des services Web :	23
5.1. Les Avantages des Services Web: [16,17].....	23
5.2. Les Inconvénients des Service Web[18] :	23
6. Quelques domaines d'application de services Web :	24
7. Conclusion.....	24

Chapitre 2 : « Qualité de service (QoS)»

1. Introduction :	25
2. Qualité de service :	25
2.1. Différents paramètres de qualité d'un service :	25
2.2. Critères de Qualité de Service :	26

«Table des Matières »

3.Modèles QoS existants :	27
3.1.Paramètre liée au Temps d'exécution :	27
3.2.Paramètre de QoS liée au transactions :	28
3.3.Paramètre de QoS liée à la sécurité :	28
3.4.Paramètre de (QoS) liée à la gestion de la configuration et coût :	29
4.La sélection des services web :	29
4.1.Propriétés fonctionnelles et non fonctionnelles dans les Web services :	30
4.1.1.Les propriétés fonctionnelles :	30
4.1.2.Les propriétés non fonctionnelles :	30
4.1.3.Sélection basée sur les besoins fonctionnels :	31
4.1.4.Sélection basée sur les besoins non fonctionnels :	31
5.conclusion :	32

Chapitre 3 : « Approches de sélection de service web »

1.Introduction aux Approches de sélection de services web :	32
2.La selection mono-objective:	32
A.La sélection globale :	32
i.L'optimisation globale exacte :	32
ii.L'optimisation globale approximative (méta-heuristiques) :	33
B.La sélection locale :	34
C.La sélection hybride :	35
iii.La sélection multi-objective :	36
iv.La sélection mono et multi-objective :	36
3. Conclusion	36

Chapitre 4 : « Conception du système »

Introduction:	38
La Méthode TOPSIS	38
Dominance	42
Analyse globale :	44
Besoin fonctionnel :	44
Diagramme Cas d'utilisation.....	44
Scenario Global	45
Conclusion:.....	53

«*Table des Matières*»

«Table des Matières »

Chapitre 5 : « Implémentation et étude de cas »

1. Introduction	54
2. Environnement de développement	54
2.1. Environnement matériel et logiciel :	54
2.2. Langage de programmation :	55
2.2.1. Le langage Java :	55
2.2.2. Le Langage de Modélisation Unifié :	56
2.2.3. Le langage XML :	56
2.3. Outils et technologies :	56
2.3.1. Eclipse :	57
2.3.2. Gestion de base de données (phpMyAdmin)	58
2.3.3. Visual Paradigm	59
3. Fonctionnalités :	59
3.1. Visualisation graphique des performances :	59
3.2. Sauvegarde et restauration des données	59
3.3. Transfert entre SGBD	59
4. Présentation des interfaces graphiques :	60
4.1. Interface d'accueil :	60
4.2. Les interfaces principales de notre prototype :	61
5. Conclusion :	68
« Conclusion générale »	
Conclusion général	69

faz

«Table des Figures »

Figure 1.1 : Architecture de base des services web.....	8
Figure 1.2 : Principe de Fonctionnement des Services Web	10
Figure 1.3 : Architecture en La pile Protocolaire simplifiée des services web.....	12
Figure 1.4 : Schéma d'un message SOAP.....	14
Figure 1.5 : Structure d'une description WSDL 1.1.....	15
Figure 1.6: Exemple d'un document WSDL.....	18
Figure 1.7 : Le contenu de l'annuaire UDDI[7].....	19
Figure 1. 8 : Entités composant un annuaire UDDI	20
Figure 1.9 : Les structures de données d'UDDI.....	21
Figure 2.1 : Le modèle QoS	27
Figure 2.2: Guide proposé par W3C pour les QoS et leurs métriques	31
Figure 4.1: Représentation d'une matrice de décision [31].....	39
Figure 4.2 : La méthode TOPSIS [29].....	42
Figure 4.3 : Cas d'utilisation	45
Figure 4.4 : Diagramme de séquence de l'ajout d'un service	46
Figure 4.5 : Diagramme d'activité de l'ajout d'un service.....	46
Figure 4.6 : Diagramme de séquence de la suppression d'un service.	47
Figure 4.7 : Diagramme d'activité de la suppression d'un service.	48
Figure 4.8: Diagramme de séquence de la modification du service.....	49
Figure 4.9 : Diagramme d'activité la modification du service.....	49
Figure 4.10: Diagramme de séquence la modification des paramètres de calcul. (poids).....	50
Figure 4.11 : Diagramme d'activité la modification des paramètres de calcul. (Poids).....	51
Figure 4.12 : Diagramme de séquence de lancement du calcul.....	52
Figure 4.13 : Diagramme d'activité de lancement du calcul.....	52
Figure 4.14 : Diagramme de classe.	53
Figure 5.1 – Environnement logiciel utilisé.....	54
Figure 5.2 – Interface principale de Eclipse	57
Figure 5.3 : Interface serveur phpMyAdmin.....	58
Figure 5.4 : Interface visual paradigm.....	58
Figure 5.5 : Interface principale de l'application.....	60
Figure 5.6 : Interface d'affichage des services	61
Figure 5.7 : Interface choix du bouton TOPSIS pour sélection les services.	61
Figure 5.8 : Résultat calculé par TOPSIS.....	62
Figure 5.9 : Interface (1) modification des poids.....	62
Figure 5.10 : Interface (2) modification des poids.....	63
Figure 5.11 : Résultat calculé par TOPSIS (Modification des poids).....	63

«Table des Figures »

Figure 5.12: Interface choix du bouton Dominance pour sélection les services.	64
Figure 5.13: Résultat calculé par Dominance.	64
Figure 5.14 : Interface pour gère les services web.	65
Figure 5.15 : Interface login pour gère les services.	65
Figure 5.16 : Interface pour l'ajout de nouveaux services à la base de données.	66
Figure 5.17: L'ajout d'un nouveau service à la base de données.	66
Figure 5.18 : modification des paramètres d'un service.	67
Figure 5.19 : supprimer un service dans la base de données.	67
Figure 5.20: le meilleur résultat est classé à partir de Dominance et TOPSIS.	68

Résumé

Vu le nombre croissant des services Web qui peuvent répondre à un même besoin fonctionnel, la sélection d'un service Web parmi plusieurs services similaires est devenue de plus en plus une tâche délicate pour les clients des Web. Une sélection doit être alors réalisée dans le but de déterminer quels sont les Web services pertinents qui satisferaient les besoins d'un utilisateur. Dans la littérature, divergents efforts ont été dépensés pour résoudre ce problème en se basant sur les propriétés non fonctionnelles des services Web. Dans ce papier, nous proposons un modèle pour la sélection des services Web en se basant sur des critères non fonctionnelles des QoS. Ainsi pour classer les services similaires, nous appuyons sur un algorithme TOPSIS qui classe les services web en fonction des exigences non-fonctionnelles. Afin de montrer la faisabilité de l'architecture proposée, nous avons développé un prototype de sélection de services web qui est une démonstration pour des services réels.

Mots clés : Service Web, Qualité de service (QoS), Sélection des services, TOPSIS, Dominance, Java.

Abstract

The increasing number of Web services that can meet the same functional need, selecting a Web service from several similar services has become increasingly a delicate task for Web clients. A selection must be made in order to determine which relevant Web services would satisfy a user's needs. In the literature, divergent efforts have been expended to solve this problem based on the non-functional properties of Web services. In this paper, we propose a model for the selection of Web services based on non-functional criteria of QoS. Therefore, to classify similar services, we are using a TOPSIS algorithm that classifies web services according to non-functional requirements. In order to show the feasibility of the proposed architecture, we have developed a prototype of web services selection, which is a demonstration for real services.

Keywords: Web service, Quality of service (QoS), TOPSIS, Selection of service, Dominance, Java.



Introduction générale

I.Contexte :

Les dernières décennies ont été marquées par le développement rapide des systèmes d'information distribués et tout particulièrement par la diffusion de l'accès à Internet. Cette évolution du monde informatique a entraîné le développement de nouveaux paradigmes d'interaction entre applications et la maturation des réseaux et protocoles. L'interaction de différentes applications offertes par des entreprises et organisations en réseau a toujours été une affaire complexe, tant que ces applications peuvent être écrites dans des langages de programmation différents et exécutées sur des plates-formes hétérogènes, la standardisation de certains aspects de cette interaction est nécessaire.

Les organisations de tous les spectres ont déjà déplacé leurs opérations principales au Web, ce qui a entraîné une croissance rapide des différentes applications Web et en particulier les services Web. Ces derniers ont l'avantage de faciliter l'intégration d'applications ou de systèmes d'information.

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

Le problème de la recherche dans les services Web a attiré l'attention des chercheurs au cours de la dernière décennie. La raison à cela est que la technologie évolue et que beaucoup de services sont disponibles, il devient important d'être en mesure de localiser les services qui répondent à nos besoins dans un grand nuage dense d'offres.

II.Problématique :

Avec l'intégration des services Web, la qualité de service (QoS) décrite par les Web services devient la principale préoccupation des prestataires de service et des consommateurs. Les fournisseurs doivent spécifier et garantir la QoS dans leurs Web services pour rester concurrentiels et pour réaliser un revenu plus élevé. De plus, les consommateurs veulent avoir une bonne exécution du service. Avec l'augmentation croissante du nombre de fournisseurs de services, il arrive très fréquemment que de nombreux services répondent à un même besoin fonctionnel, mais avec une QoS différente, une sélection doit être alors faite pour déterminer quels sont les services pertinents. Dans ce travail, nous nous intéressons au problème de

sélection de Web services. Nous considérons les caractéristiques de qualité de service comme un critère de choix lors de la sélection.

Étant donné que les services Web sont caractérisés par les QoS, et chaque WS est caractérisé par plusieurs critères de QoS telles que : Le Temps de réponse, La Documentations, La Sécurité, Le Coût, Précision et Fiabilité telle qu'on cherche à maximiser les valeurs des critères avec un sens de préférence croissant et à minimiser les critères avec une préférence décroissante généralement de type coût.

III.Contribution :

On propose dans ce travail un modèle pour la sélection des services web prise en compte de la qualité de service QoS de telle façon la phase de la recherche des services similaire serait donc simple et facile et pour faire l'évaluation de ses critères qui focalise sur l'utilisation de l'algorithme TOPSIS pour classifie les services web a base des critères de qualité QoS, en fonction des exigences non-fonctionnelles. Nous nous inspirons cette idée après l'étude de quelques travaux qui utilisent les méthodes qui aident à la décision multicritères comme une axe principale pour la résolution de certains problème dans le domaine des services Web pour avoir une sélection des meilleurs services Web parmi les services similaires.

ajoute beaucoup des avantages au cœur de services Web, tel que au plus tard en n'est pas besoin d'évaluer les critères de qualité de service QoS au moment d'exécution ou à la phase de recherche d'un service Web. Au contraire l'évaluation se fait au moment de publication des services Web dans l'annuaire (UDDI).

IV.Plan du mémoire :

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

- ❖ **Le premier chapitre « Services Web »** : Ce chapitre est consacré à l'étude des services Web et leurs caractéristiques pour mieux comprendre les concepts de base de cette technologie. Le chapitre se comporte plusieurs notions fondamentales : nous présentons dans la première partie les services Web, leur architecture, leur modèle d'interaction et leurs standards de base. Dans la deuxième partie, nous aborderons quelque exemple réel de services Web, enfin nous allons présenter quelques avantages et inconvénients et aussi les domaines d'application des services Web.

- ❖ **Le deuxième chapitre « Qualité de service Web »** : nous donnons quelques notions sur qualité de service (QdS ou QoS en anglais : Qualité of service) telles que leur définition et leurs paramètres. Ce chapitre montre le besoin d'utilisation d'un nouveau modèle pour la sélection des services web en se basant sur des propriétés non fonctionnelles avec l'intégration d'une méthode qui aide à la décision multicritère.

- ❖ **Le troisième chapitre « Approches de sélection de service web »** : est consacré à la représentation d'une approche de sélection des services web .

- ❖ **Le quatrième chapitre « Conception du système »** : Ce chapitre fournit une vue détaillée des besoins fonctionnels de notre application, scénarios des cas d'utilisations, diagrammes de séquence, et diagramme d'activité réalisés avec le langage de modélisation UML, puis un scénario global qui résume le fonctionnement de l'application, et enfin le diagramme de classe qui illustre les relations entre les différentes classes de l'application.

- ❖ **Le cinquième chapitre « Implémentation du système »** : Dans ce dernier chapitre, nous allons présenter l'environnement logiciel et matériel sur lequel le système sera réalisé et validé, les détails d'implémentation de notre application, propose une mise en œuvre pratique et une présentation des résultats expérimentaux.

- ❖ Ces cinq chapitres sont clôturés par une conclusion générale qui fait une synthèse sur les différents points suivis pour la réalisation de notre travail ainsi que les perspectives qui nous semblent intéressantes pour continuer ce travail.

Chapitre 1



Les Services Web

1. Introduction :

L'essor du Web a été marqué par le développement rapide des systèmes d'information distribués, notamment par la diffusion de l'accès à l'Internet car il a surgi le besoin d'assurer et simplifier la communication entre une application cliente qui invoque un service d'une application serveur en utilisant l'Internet. Ce besoin a été l'origine de ce qui se connaît comme "Service Web"

Les services Web sont devenus le facteur le plus significatif technologique par produit. De cela, l'utilité de cette technologie devient très évidente et importante. Malgré le succès de cette technologie, les services Web ont confronté le problème de l'interopérabilité, et pour s'en débarrasser, un ensemble de standards a été élaborée, ce qui nous invite à aborder les technologies liées telles que : eXtensible Markup Language(XML), Simple Object Accès Protocol(SOAP), Web Services Description Language (WSDL), et Universal Description Discovery and Intégration (UDDI).

L'une des questions de recherche des plus importantes dans le domaine des services Web est relative à la prise en compte des différents critères de la qualité (QoS, Quality of Service) de service dans la découverte et la sélection du meilleur service Web ayant une fonctionnalité donnée, afin de satisfaire au mieux le demandeur du service. En effet, on retrouve de plus en plus des services Web ayant une même fonctionnalité. Pour cela, plusieurs études proposent l'utilisation des différents critères non fonctionnels de qualité de services afin de les comparer et ainsi de choisir le meilleur parmi ceux disponibles [1].

Dans ce chapitre, nous présenterons un état de l'art sur les services Web dans laquelle nous concentrons sur : l'architecture orientée services et sa principale réalisation, les services Web est ses concepts et les critères des qualités de service.

2. Définition :

Depuis les dernières années, l'utilisation de Services web a connu une popularité grandissante. Ces services sont très utilisés notamment par les entreprises pour rendre accessible leurs métiers ou leurs données via le web. Les services web ont été proposés initialement par IBM et Microsoft, puis en partie standardisés par le W3C (le consortium du World Wide Web).Plusieurs définitions des services web ont été proposées dans la littérature :

D'un point de vue technique, un service Web est une entité logicielle offrant une ou plusieurs fonctionnalités allant des plus simples aux plus complexes. Ces entités sont publiées, découvertes et invoquées à travers le web grâce à l'utilisation d'internet comme infrastructure de communication ainsi que de formats de données standardisés comme XML [19].

Cette définition met en évidence uniquement le standard XML, donc n'importe quelle application orientée Internet qui garantit ces caractéristiques et qui utilise les technologies basées sur XML est aussi un Service web.

Selon IBM¹ :

« 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 ».

Les deux premières définitions affirment que les services web sont accessibles par d'autres à travers le web en utilisant des protocoles et des formats standards, mais elles ne mettent pas en évidence les technologies utilisées pour mettre en œuvre un service web.

Selon le W3C²:

« Un service web est un système logiciel identifié par une URI et conçu pour supporter l'interaction interopérable de machine à machine sur un réseau. Il possède une interface décrite dans un format exploitable par la machine, c.à.d. décrite en WSDL (web Services Description Language). D'autres systèmes interagissent avec le service web d'une façon prescrite par sa description en utilisant des messages SOAP (Simple Object Access Protocol), typiquement en utilisant HTTP (HyperText Transfer Protocol) avec une sérialisation XML en même temps que d'autres normes du Web ».

Cette définition met l'accent sur les standards de l'Internet et l'interface ouverte qui permet les invocations des services. Cependant, elle n'est pas encore suffisamment précise parce qu'elle ne mentionne pas la découverte de services web (UDDI).

¹ IBM Web services tutorial. Online: <http://www-106.ibm.com/developerworks/webservices/>

²www.w3c.org

Une définition plus raffinée des services web est fournie par le dictionnaire Webopedia³ qui définit un service web comme « une manière standardisée d'intégration des applications basées sur le Web en utilisant les standards ouverts XML, SOAP, WSDL, UDDI et les protocoles de transport de l'Internet. XML est utilisé pour représenter les données, SOAP pour transporter les données, WSDL pour décrire les services disponibles, et UDDI pour lister les fournisseurs de services et les services disponibles ».²

3. Architecture orientée service (SOA):

Plusieurs paradigmes de développement de logiciel ont été proposés pour faciliter la création, l'exposition, l'interconnexion et la réutilisation d'applications à base de services. C'est pour cette raison que l'informatique distribuée a dû passer par le modèle client-serveur, l'architecture à plusieurs niveaux (l'orienté objet, l'orienté composant), les standards basés COBRA (Common Object Request Broker Architecture) de l'OMG (Object Management Groupe), Java RMI (Remote Method Invocation) de Sun, DCOM (Distributed Component Object Model) de Microsoft pour enfin aboutir à l'architecture orientée service (SOA) qui est née pour répondre aux inconvénients des technologies orientées composants tels que : la complexité, le manque de compatibilité, en plus ils sont difficilement interopérables entre eux et la modification dans un composant conduit généralement à une défaillance du système. Comme, ils sont souvent utilisés en Intranet [2].

3.1. Définition de SOA :

L'architecture orientée service désignée par le vocable de SOA (« Service Oriented Architecture »), est un style architectural pour la conception, le développement, le déploiement et la gestion de systèmes logiciels distribués qui délivre des fonctionnalités d'application sous forme de service, soit à l'utilisateur final ou à d'autres services [2]. Cette infrastructure vise à permettre l'échange d'informations entre applications.

Ce modèle définit un système par un ensemble de composants logiciels distribués qui collaborent afin de réaliser une fonctionnalité globale préalablement établie [3]. Elle permet de répondre aux problèmes d'intégration dans les architectures logicielles en facilitant l'ajout et la suppression d'un module sans remettre en cause toute l'architecture.

²www.w3c.org

³<http://-www.webopedia.com>

En d'autre terme, cette architecture consiste à diviser un logiciel répondant à un problème, en un ensemble d'entités qui représente en un ensemble de fonctions basiques proposant des services applicatifs indépendants et interopérables. Chacune de ces entités peut utiliser les services proposés par d'autres entités. On obtient ainsi un réseau de services interagissant entre eux. En outre, SOA permet de construire des systèmes informatiques évolutifs et adaptables, en améliorant leur qualité et en simplifiant leur intégration dans l'infrastructure informatique de l'entreprise par recours aux services [4].

3.2 Objectifs:

- ❖ L'architecture SOA vise trois objectifs importants :
 - Identification des composants fonctionnels.
 - Définition des relations entre ces composants.
 - Etablissement d'un ensemble de contraintes sur chaque composant de manière à garantir les propriétés globales de l'architecture.

3.3 Avantages

- ❖ SOA possède plusieurs avantages, parmi eux [5], [6] :
 - SOA réside dans le fait que les applications réparties n'ont plus besoin d'un système de middleware réparti commun pour communiquer, mais seulement des protocoles de communication interopérables sur Internet.
 - SOA peut donc être représentée par une interconnexion de multiples points d'accès.
 - Le contrat de service du modèle des SOA s'inspire directement du modèle des contrats professionnels de service. Dans le cas des services web, ce contrat est un document WSDL.
 - Une grande tolérance aux pannes et une maintenance facilitée.
 - Une bonne modularité permettant de remplacer facilement un composant (service) par un autre puisque ça ne gêne pas le fonctionnement des autres services.
 - La possibilité de l'évolution et la réutilisabilité des composants.
 - La découverte des services disponibles, afin d'être sûr que les utilisateurs potentiels découvriront le service dont ils ont besoin.
 - Une réduction du coût de développement et d'intégration d'application.
 - La mise à l'échelle est rendue possible grâce à la découverte et à l'invocation de nouveaux services lors de l'exécution.

L'approche la plus répandue pour mettre en œuvre une architecture SOA est l'utilisation des services Web. Cette technologie vise à la transposition des SOA dans le cadre du Web.

4. Architecture des services web :

L'architecture des services Web est une instance de SOA. L'interopérabilité qu'impliquent les services Web doit être soutenue par une architecture stable et fiable. Deux types d'architecture existent pour les services Web : La première dite architecture de **référence**, elle contient trois couches principales. La seconde architecture est plus complète, elle utilise les couches standards de la première architecture en ajoutant au-dessus d'autres couches plus spécifiques. Elle est appelé architecture **étendue** ou encore en Pile.

4.1. Architecture de référence:

L'architecture classique de « référence » des Services Web se fonde sur le modèle SOA (Service Oriented Architecture) qui fait intervenir trois acteurs : un client, un fournisseur ainsi qu'un intermédiaire jouant le rôle d'annuaire (**Figure 1.1**). [7]

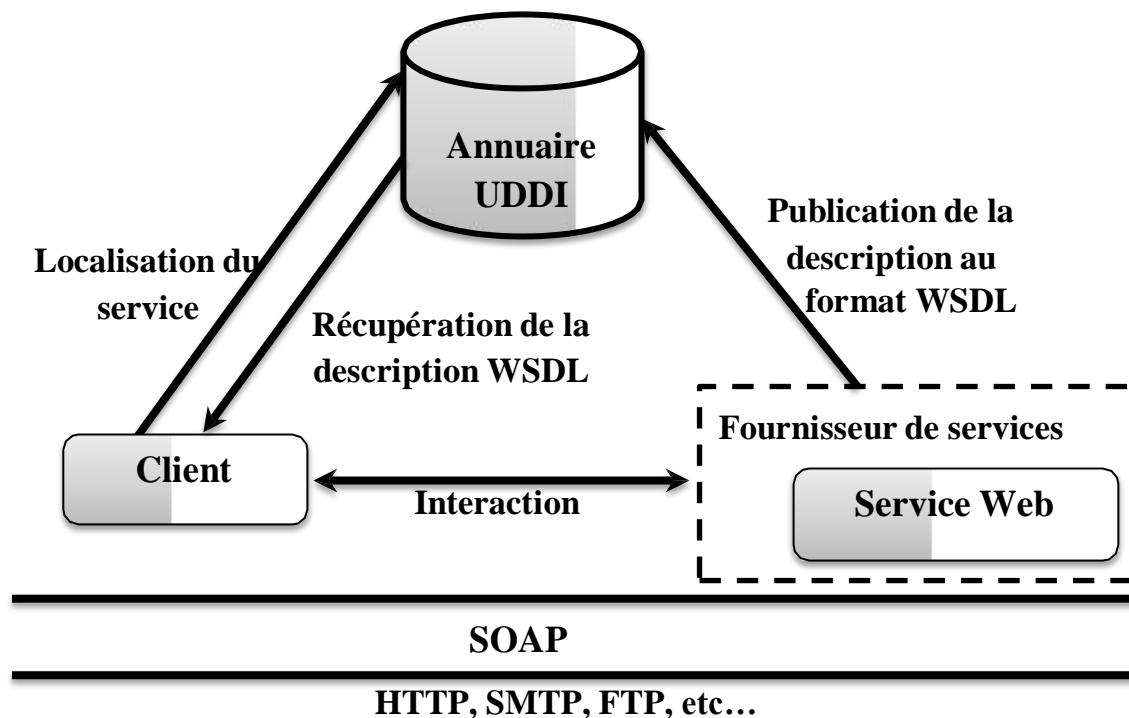


Figure 1.1 : Architecture de base des services web.

4.1.1. Les acteurs :

- 1) **Le Fournisseur du service** : qui publie son service en fournissant la description au format WSDL dans l'annuaire de services et réalise les opérations propres au service. C'est donc le propriétaire du service web. Il représente l'environnement [7] d'hébergement et d'exécution du service. Il est constitué de trois couches de bases :
 - ↳ **La couche de données** : contient les différentes bases de données utilisées par le service.
 - ↳ **La couche applicative** : c'est la plateforme de développement qui assure l'exécution du service web.
 - ↳ **La couche de description** : elle expose les fonctionnalités du service via un fichier WSDL.

- 2) **L'Annuaire**: qui, d'une part reçoit et enregistre les descriptions de services publiées par les fournisseurs, et d'autre part reçoit et répond aux recherches de services lancées par les clients. C'est donc un registre de description qui offre aux fournisseurs le moyen de publier et d'indexer leurs services web sur le réseau. Il permet également aux clients de rechercher ces services selon plusieurs critères. [7]

- 3) **Le Client** : qui obtient la description du service grâce à l'annuaire et utilise le service. Dans l'architecture des services web, le Client est défini comme le consommateur du service. Il peut accéder à ce dernier en échangeant avec le fournisseur des messages SOAP. Cet échange de messages se fait généralement à l'aide des protocoles Internet standards tels que HTTP, SMTP, etc. Techniquement, le Client peut être une simple application Windows ou web, comme il peut être un autre service web.

Par conséquent, un service web est toujours accompagné d'une description fournissant aux applications les informations nécessaires à son utilisation. Les services sont implantés par les fournisseurs qui mettent à disposition les descriptions de services sous forme de fichiers WSDL. Ces descriptions sont centralisées et stockées dans des annuaires. La notion d'annuaire est comparable aux annuaires téléphoniques que nous utilisons pour accéder à des personnes ou des services. Les applications clientes envoient des requêtes aux annuaires pour sélectionner les services, de la même manière que nous recherchons un numéro dans un annuaire téléphonique. Elles téléchargent ensuite les descriptions des services sélectionnés, et les invoquent directement. [7].

4.1.2. Les opérations principales :

Le fonctionnement des Services Web repose sur un modèle en phases, dont les trois phases fondamentales sont les suivantes : la publication d'un service, la découverte d'un service et l'interaction avec un service. Ce fonctionnement est normalisé à travers un certain nombre de standards : un protocole abstrait de description et de structuration des messages SOAP, une spécification XML qui permet la publication et la localisation des services dans les annuaires UDDI (Universal Discovery, Description and Integration) et un format de description des Services Web publiés dans les annuaires (WSDL), **figure1.2** [3].

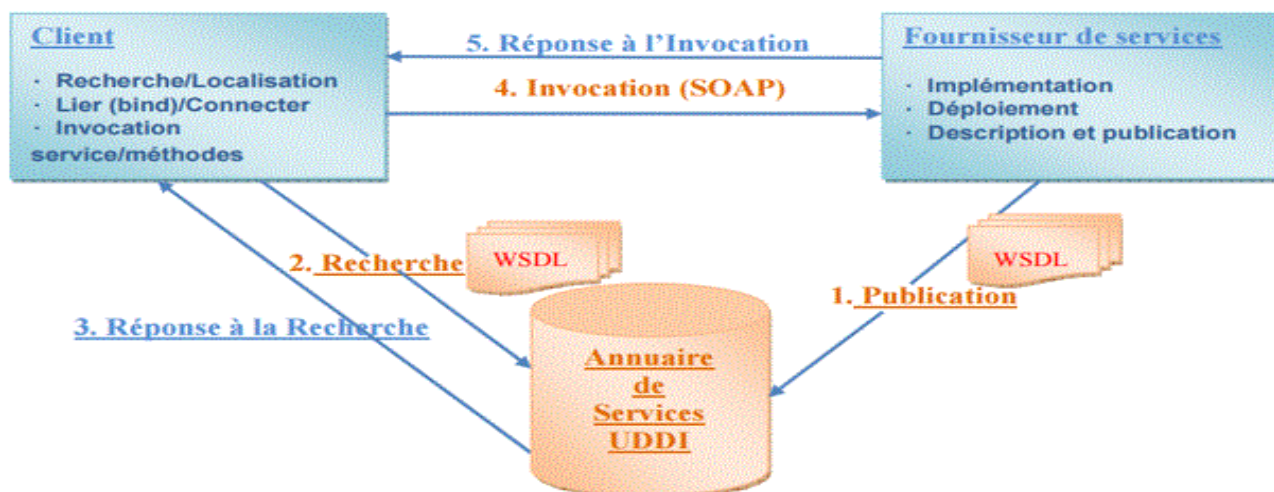


Figure 1.2 : Principe de Fonctionnement des Services Web .

Tous ces couches et standards sont détaillés dans les sections suivantes :

- ❖ **Publication** (Publier) : Une fois que le fournisseur a construit son Service Web, il a besoin de publier ce service afin que les autres (personnes, machines) soient capables de le découvrir puis de l'utiliser. Un annuaire UDDI est utilisé pour publier les Services Web sur un dépôt central. Afin d'être publié dans un annuaire UDDI, le Web Service a besoin d'être décrit. La description de ce service est effectuée dans le langage WSDL. Cette description indique les fonctions de ce service, ainsi que les paramètres d'entrée/sortie et le protocole de transport.

- ❖ **Découverte** (rechercher) : Une fois que les Services Web sont publiés dans un annuaire par les fournisseurs, l'utilisateur peut chercher des Services Web en lançant une requête auprès de l'annuaire UDDI pour recevoir la liste des Services Web pertinents par rapport à sa requête.

- ❖ **Réponse à la recherche** : Le résultat de la recherche dans l'annuaire est un fichier WSDL qui sera transmis, dans un message SOAP, au client demandeur. Ce fichier contient la description du service et les informations techniques nécessaires pour son invocation [3].

- ❖ **Invocation** (consommer) : Une fois que l'utilisateur a découvert le Service Web via l'interface UDDI, et qu'il a pris la décision d'utiliser ce service dans son application, il a besoin d'invoquer le Service Web.

- ❖ **Réponse à l'invocation** : Le fournisseur réagit à la réception du message d'invocation de la part du client en envoyant au client la réponse résultant de l'exécution de la procédure. Cette réponse est transmise sous la forme d'un document XML via SOAP [3].

4.2. Architecture étendue :

Cette architecture est aussi appelée « pile des services Web », du fait qu'elle est constituée de plusieurs couches se superposant les unes aux autres où chaque couche s'appuyant sur un standard particulier et répond à des préoccupations fonctionnelles différentes telles que la sécurité, la description, la messagerie fiable, le transport et les transactions.

Dans cette section, nous allons présenter d'abord le langage XML. Après quoi nous définirons les différentes couches horizontales illustrées dans la figure 1.3 qui décrit la pile de langages, de protocoles et de modèles des services web. Il est à noter que les couches verticales présentées dans cette même figure sortent du cadre de cette thèse.

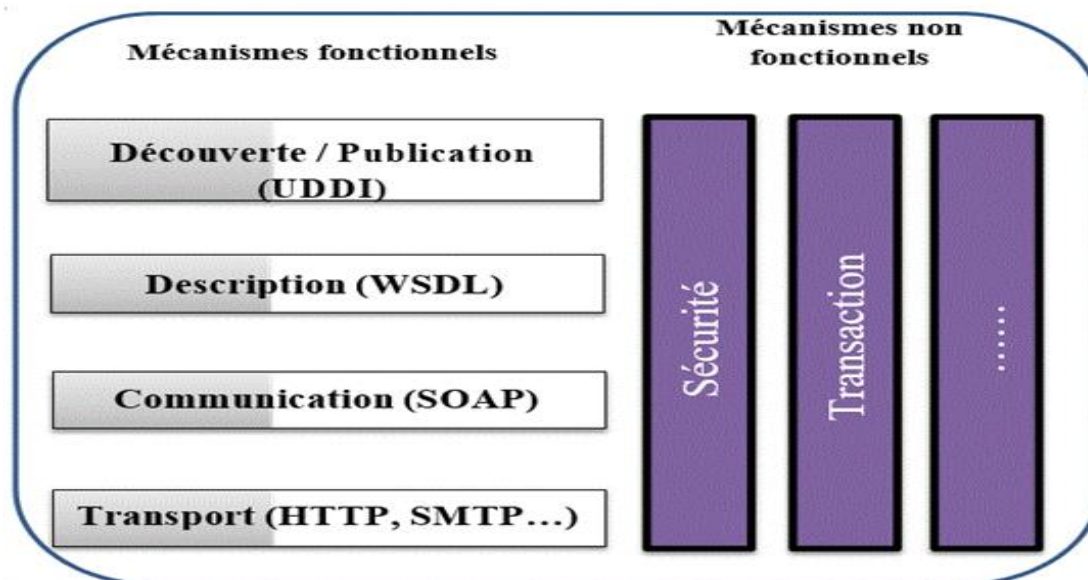


Figure 1.3 : Architecture en La pile Protocolaire simplifiée des services web.

4.3. Langages et protocoles utilisés par les services web :

L'architecture fonctionnelle des services web que nous avons présenté précédemment montre l'utilisation de nombreux langages et protocoles durant le déploiement et l'invocation des services web. L'atout principal des protocoles SOAP et UDDI ainsi que du langage WSDL est de se reposer sur le langage XML (eXtensible Markup Language). Cette même architecture montre aussi que les services web se basent sur l'utilisation des normes actuelles d'Internet comme le protocole HTTP.[7]

4.3.1 Le langage XML (eXtensible Markup Language) :

Le langage XML standardisé par le W3C en 1998 est aujourd'hui largement reconnu et utilisé par de nombreuses entreprises comme format universel d'échange de données. XML est un métalangage de représentation de données. Il définit un ensemble de règles de formatage pour composer des données valides.

XML constitue la technologie de base des architectures Web services ; c'est un facteur important pour contourner les barrières techniques. XML est un standard qui permet de décrire des documents structurés transportables sur les protocoles d'Internet. En effet, il apporte à l'architecture des services web l'extensibilité et la neutralité vis-à-vis des plateformes et des langages de développement [8]. De plus, grâce à la structuration, XML permet la distinction entre les données des applications et les données des protocoles.

La technologie des services web a été conçue pour fonctionner dans des environnements totalement hétérogènes. Cependant, l'interopérabilité entre les systèmes hétérogènes demande des mécanismes puissants de correspondance et de gestion des types de données des messages entre les différents participants (clients et fournisseurs). C'est une tâche où les schémas de type de données XML s'avèrent bien adaptés. C'est pour cette raison que la technologie des services web est essentiellement basée sur XML ainsi que les différentes spécifications qui tournent autour (les espaces de nom, les schémas XML, et les schémas de Type) [9].

Parmi les spécifications XML, nous soulignons :

- ↳ **XSD (XML Schema)** : c'est un langage qui sert à décrire formellement un vocabulaire [10].
- ↳ **XSLT (Extensible Stylesheet Language Transformations)** : est utilisé pour transformer un document XML basé sur un certain schéma en un autre document XML qui peut être un document lui-même basé sur un autre schéma [11].
- ↳ **XPath (XML Path Language)** : fournit une syntaxe d'expressions utilisées pour créer des chemins de localisation [12].

4.3.2 La couche de Transport :

Est le mécanisme utilisé pour 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 . . . permettant ainsi aux services Web de rester indépendants du mode de transport utilisé.

4.3.3 La couche de Communication (SOAP) :

Est un mécanisme utilisé par le fournisseur et le client de service pour communiquer le contenu des requêtes et des réponses. Cette couche spécifie les protocoles d'échanges de documents XML entre le service web et ses clients, elle caractérise aussi le mode d'échange (s'il est bloquant ou non). Le protocole SOAP est adopté comme un standard pour la messagerie entre les services web.

- ↳ **SOAP** : (Simple Object Access Protocol) [8m] est un protocole d'échange de message indépendant des plateformes, c'est un produit de Microsoft et IBM.

La première version de SOAP a été acceptée par le W3C (World Wide Web Consortium) en 2000. SOAP est constitué de deux parties : une enveloppe XML, et un entête d'un

protocole de transport. La spécification du protocole SOAP ne donne aucune indication sur le mécanisme de transport du message.[7].

SOAP fait une séparation entre le message (C.à.d. le document XML) et le moyen de transport utilisé. Actuellement, SOAP utilise des protocoles tels que : HTTP ou SMTP pour assurer le transport. L'enveloppe XML contient à son tour trois sous éléments :

- ↳ **L'entête « Header »**: est optionnel, il peut contenir des informations de sécurité (telles que les signatures électroniques), des informations transactionnelles, des informations de traçabilités, etc.

- ↳ **L'élément « Body »**: est obligatoire, il contient les éléments suivants :
 - Soit le nom de méthode, avec les données correspondantes, ou un simple document XML, pour le cas d'une requête.
 - Soit les valeurs de retour pour le cas d'une réponse.

- ↳ **L'élément « Fault »**: est optionnel, il fournit des informations sur d'éventuelles erreurs survenues lors de l'analyse du message.

La figure 1.4 représente le format standard d'un message SOAP. [7]

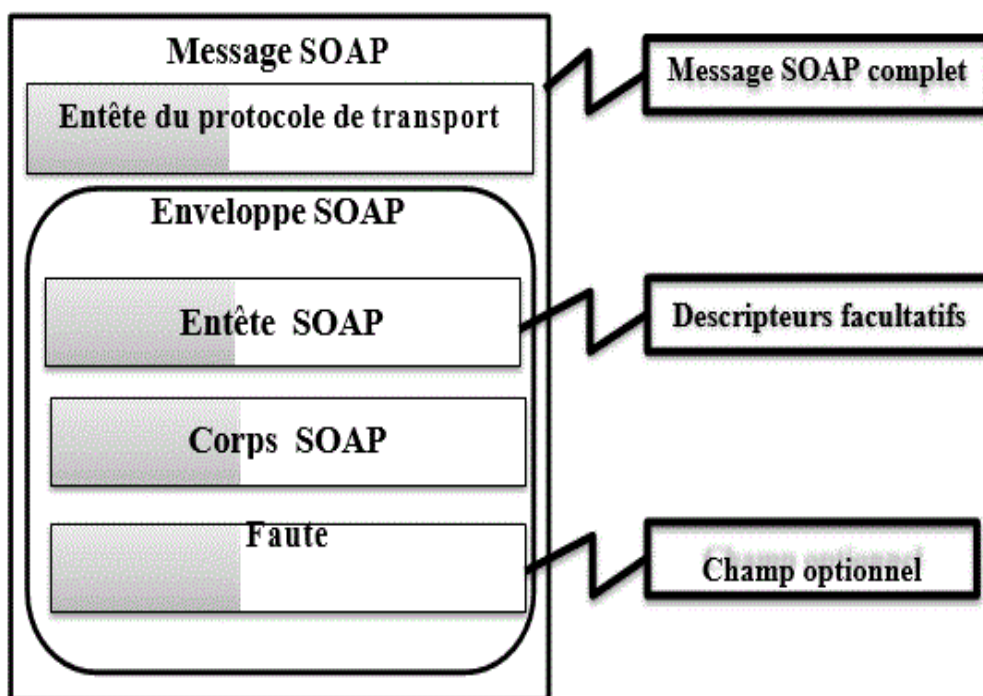
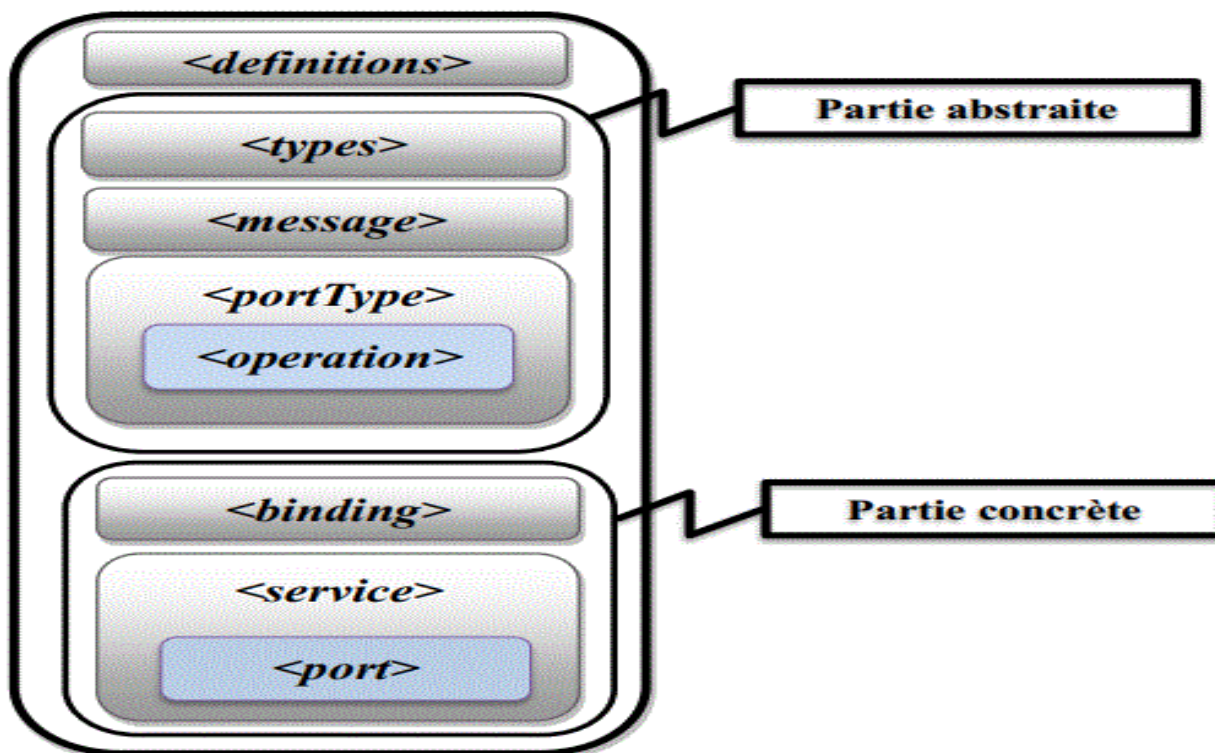


Figure 1.4 : Schéma d'un message SOAP.

4.3.4 La couche de Description (WSDL) :

WSDL (Web Service Description Language) est un standard du W3C qui permet de définir la structure abstraite et concrète d'un service. C'est un document XML qui décrit la signature des opérations offertes par le service (nom d'opérations, noms et types des paramètres d'entrées/sorties), il définit aussi des liaisons concrètes pour ces opérations telles que le protocole de transport, l'URI du service, le style du service, et les règles d'encodage employées pour les paramètres d'entrées/sorties. [7]

Nous notons que cette interface décrit juste la structure du service (la partie quoi) et non pas le comportement (la partie comment). Le document WSDL comporte six éléments: <definitions>, <types>, <message>, <portType>, <binding> et <service>.



La figure 1.5 offre une représentation concise de la spécification WSDL. [7]

Figure 1.5 : Structure d'une description WSDL 1.1

- ↪ **<definitions>** : L'élément « définitions » représente la racine du document WSDL. Il définit le nom du service web, déclare les différents espaces de nommage utilisés dans le reste du document, et contient tous les éléments qui décrivent le service web.

- ↪ **<types>** : L'élément « types » décrit tous les types de données complexes utilisés entre le client et le serveur. WSDL n'est pas exclusivement lié à un système de typage spécifique, mais il utilise la spécification W3C XML Schéma comme choix par défaut. Si le service utilise uniquement des types simples, tels que les chaînes de caractères et les entiers, l'élément types n'est pas requis.
- ↪ **<message>** : L'élément « message » décrit les messages échangés par le service (les messages entrants et sortants). Il est composé d'un ou plusieurs éléments nommés <part>. ces derniers peuvent faire référence à des paramètres d'entrée ou à des valeurs de retour.
- ↪ **<portType>** : L'élément « portType » représente une collection d'opérations. Une <operation> est une action abstraite accomplie par le service. Elle peut contenir éventuellement les sous éléments <input> et /ou <output> décrivant respectivement le message d'entrée et le message de sortie.

La spécification WSDL décrit quatre types d'opérations:

- ✓ Opération à sens unique (One-way) : Le service reçoit un message. L'opération a donc un seul élément <input>.
 - ✓ Opération de type requête – réponse (Request-response): Le service reçoit un message et envoie une réponse. L'opération a donc un élément <input>, suivi d'un élément <output>. Pour encapsuler des erreurs, un élément optionnel<fault> peut également être spécifié.
 - ✓ Opération de type réponse sollicitée (Solicit-response): Le service envoie un message et reçoit une réponse. L'opération a donc un élément <output>, suivi d'un élément <input>. Pour encapsuler des erreurs, un élément optionnel<fault> peut également être spécifié.
 - ✓ Opération de type notification (Notification): Le service envoie un message. L'opération a donc un seul élément <output>.
- ↪ **<binding>** : L'élément « binding » associe un portType avec des informations concrètes telles que le protocole de transport (http, ftp...), les règles d'encodage de données, et le style du service (RPC ou DOC).

- le style RPC impose un format précis aux messages SOAP (c.à.d. le nom de la méthode et les arguments pour la requête, et la valeur de retour pour la réponse). Ce style est adapté aux échanges synchrones.
 - Le style DOC autorise n'importe quel document XML bien formé comme requête ou réponse. Ce style est adapté aux échanges asynchrones.
- ↪ **<service>** : L'élément « service » est constitué d'un ensemble de <port>. Un <port> est une association entre un <binding> et un point d'accès (c.à.d. l'url) qui indique l'adresse du service web. Nous pouvons avoir plusieurs ports par services, (un même binding et des URLs différents, ou des « binding » différents et éventuellement des URLs différents).

En plus de ces six principaux éléments, la spécification WSDL définit également les éléments facultatifs suivants:

- ↪ **<documentation>** : L'élément « documentation » est utilisé pour fournir des informations lisibles par l'utilisateur (par exemple : description textuelle du service en utilisant le langage naturel). Il peut être inclus à l'intérieur de n'importe quel autre élément WSDL.
- ↪ **<import>** : L'élément « import » est utilisé pour importer d'autres documents WSDL ou des schémas XML. Par exemple, deux documents WSDL peuvent importer les mêmes éléments de base et encore inclure leurs propres éléments pour assurer le même service sur deux adresses physiques différentes. Il est à noter que cette fonctionnalité n'est pas supportée par tous les outils WSDL pour l'instant.

Pour plus d'illustration, la figure 1.6 représente la description WSDL du service simple « HelloService ». Ce dernier offre une fonction (opération) appelée « sayHello ». La fonction reçoit un message « SayHelloRequest » avec une chaîne de caractères représentant le paramètre d'entrée, et retourne un message « SayHelloResponse » qui contient une autre chaîne de caractères. Par exemple, si vous passez le mot «world» comme paramètre, le service renvoie le message d'accueil, « Hello, world! ».

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions name="HelloService"
  targetNamespace="http://www.ecerami.com/wsdl/HelloService.wsdl"
  xmlns="http://schemas.xmlsoap.org/wsdl/"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns:tns="http://www.ecerami.com/wsdl/HelloService.wsdl"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <message name="SayHelloRequest">
    <part name="firstName" type="xsd:string"/>
  </message>
  <message name="SayHelloResponse">
    <part name="greeting" type="xsd:string"/>
  </message>
  <portType name="Hello_PortType">
    <operation name="sayHello">
      <input message="tns:SayHelloRequest"/>
      <output message="tns:SayHelloResponse"/>
    </operation>
  </portType>
  <binding name="Hello_Binding" type="tns:Hello_PortType">
    <soap:binding style="rpc"
      transport="http://schemas.xmlsoap.org/soap/http"/>
    <operation name="sayHello">
      <soap:operation soapAction="sayHello"/>
      <input>
        <soap:body
          encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
          namespace="urn:examples:helloservice"
          use="encoded"/>
      </input>
      <output>
        <soap:body
          encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
          namespace="urn:examples:helloservice"
          use="encoded"/>
      </output>
    </operation>
  </binding>
  <service name="Hello_Service">
    <documentation>WSDL File for HelloService</documentation>
    <port binding="tns:Hello_Binding" name="Hello_Port">
      <soap:address
        location="http://localhost:8080/soap/servlet/rpcrouter"/>
    </port>
  </service>
</definitions>
```

Figure 1.6 : Exemple d'un document WSDL [13].

La version 1.0 de l'interface WSDL a été créée en 2000 par IBM, Microsoft et Ariba, ensuite le consortium W3C a publié la version WSDL 1.1 en mars 2001 comme une note [14].

En juin 2007, la version WSDL 2.0 a été créée comme une recommandation W3C (C.à.d. norme). WSDL 2.0 a apporté quelques changements à la version WSDL 1.1, ils sont résumés comme suit :

- ✓ <portType> est devenu <interface>.
- ✓ <port> est remplacé par <endpoints>.
- ✓ <message> est supprimé et l'élément <xs :element> est utilisé pour spécifier les données échangées.

4.3.5 La couche de Découverte et de Publication (UDDI) :

UDDI (Universal Description, Discovery and Integration) est une spécification d'annuaires de services web, cette norme W3C propose un ensemble de structures à publier par les fournisseurs de services. Ces structures sont formalisées en XML, elles proposent 03 types d'informations (voir la figure 1.7) : [15]

- ✓ **Les pages blanches** : qui décrivent les informations de contacts sur les entreprises.
- ✓ **Les pages jaunes** : qui décrivent des informations de classification de services.
- ✓ **Les pages vertes** : qui donnent des informations techniques des services.

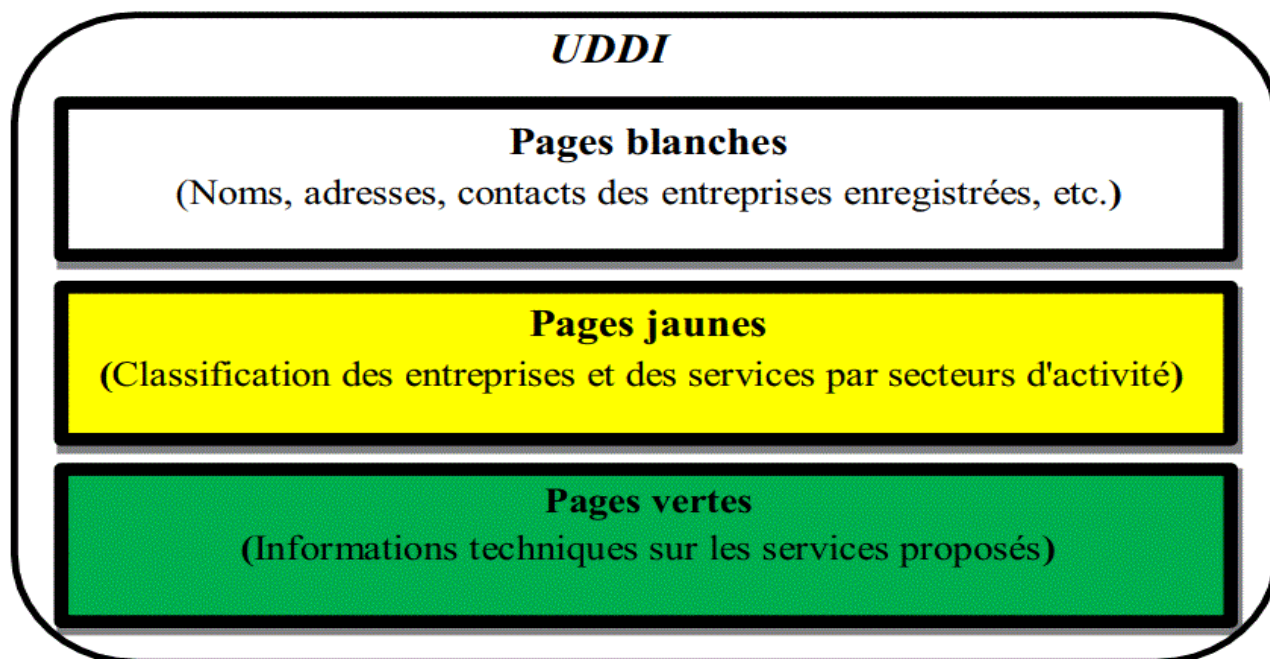


Figure 1.7 : Le contenu de l'annuaire UDDI[7].

La norme UDDI offre aussi une API aux applications clientes. L'API UDDI propose deux fonctionnalités principales : **la recherche** et **la publication**, de services dans un annuaire UDDI. La recherche de services est réalisée en se basant sur différentes méthodes de recherche : mots clés, identifiant du service, nom du service, etc. La publication de services est basée sur des formulaires de publication : la description d'un document tModel, la recommandation de service (permet à des clients de services non propriétaires de ces derniers de les recommander), etc.

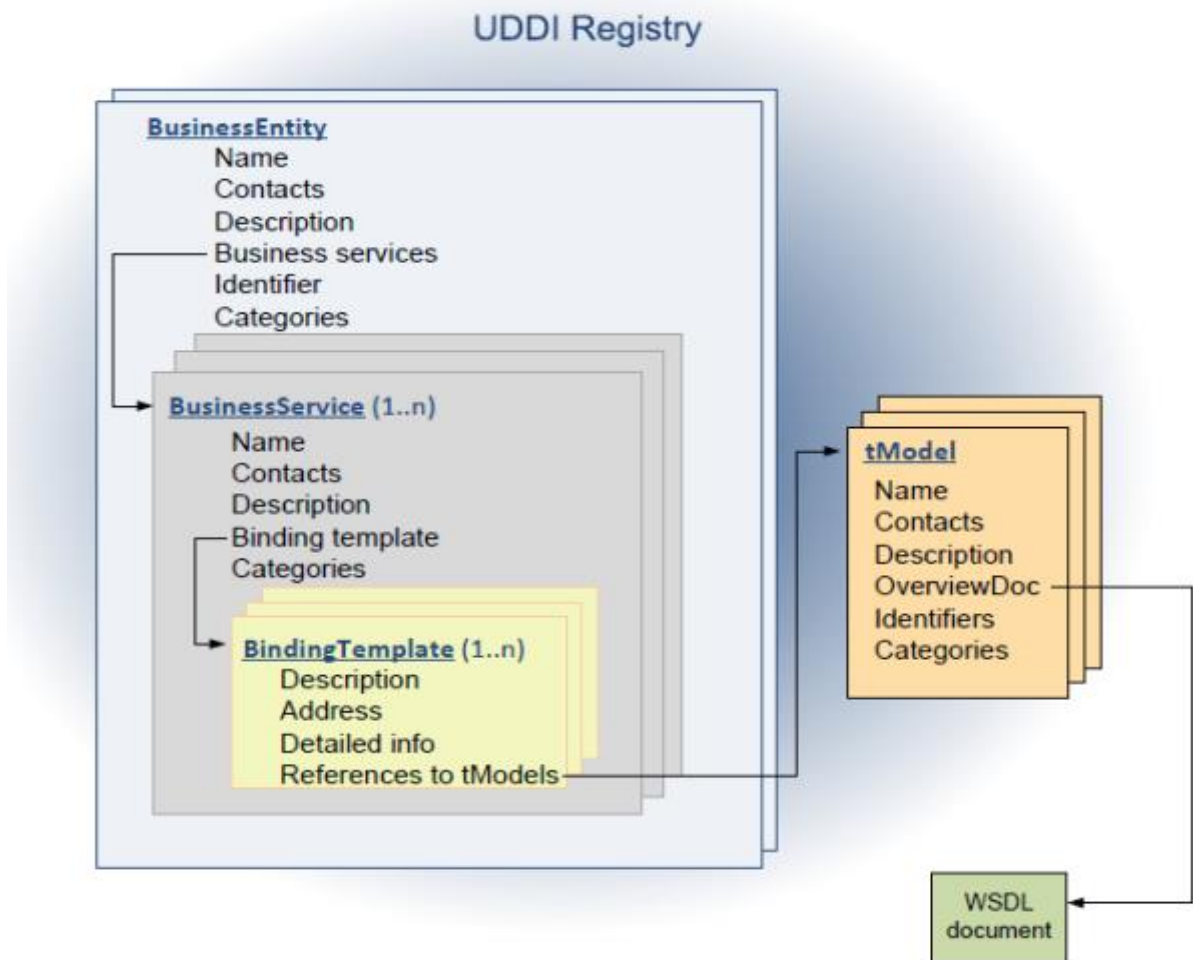


Figure 1. 8 : Entités composant un annuaire UDDI .

Nous distinguons deux types d'annuaires UDDI (publiques et privés) :

- **L'annuaire UDDI public** : est implémenté sous forme d'un réseau de nœuds UDDI, ces nœuds sont synchronisés, chacun d'eux est possédé par une entreprise donnée (telles que SAP, IBM et Microsoft). La publication d'un service chez une entreprise Propage automatiquement ses informations (pages blanches, jaunes et vertes) aux différents nœuds

UDDI. L'accès à l'ensemble des informations des registres peut se faire à n'importe quel nœud UDDI. Ce type d'annuaire est gratuit.

- **L'annuaire UDDI privé** : permet à une entreprise de choisir les partenaires pour lesquels elle autorise la publication et l'invocation de ses services web.

Les structures de données d'un UDDI sont illustrées dans la figure 1.9. [7]

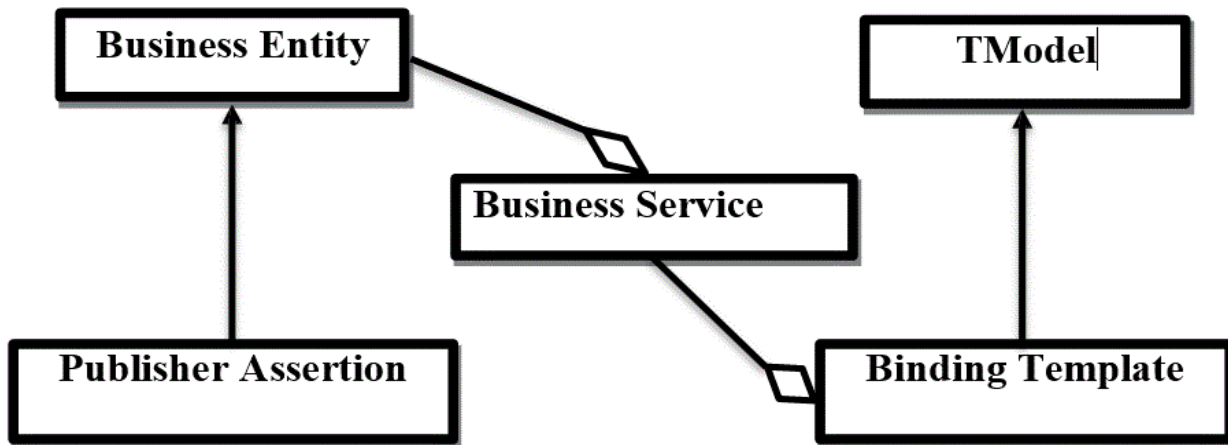


Figure 1.9 : Les structures de données d'UDDI.

- ❖ **La partie « pages blanches »** : est constituée de deux éléments `<businessEntity>` et `<publisherAssertion>`.
 - ✓ `<businessEntity>` donne des informations de contact sur l'organisation fournissant les services (les noms des dirigeants, les emails, les numéros de téléphones, le site web, etc.).
 - ✓ `<publisherAssertion>` spécifie les liens d'affiliation entre deux entreprises (mère et fille). Lorsque deux éléments `<businessEntity>` référencent la même `<publisherAssertion>`, nous parlons de relation d'affiliation entre ces `<businessEntity>`.
- ❖ **La partie « pages jaunes »** : est constituée des éléments `<businessService>`.
 - ✓ `<businessService>` décrit un ensemble de services fournis par une organisation (le nom du service, la description textuelle, les informations de classification). Un `<businessService>` est un sous élément de `<businessEntity>`.

- ❖ **La partie « pages vertes »** : est constituée des éléments `<bindingTemplate>` et `<tModel>`.
 - ✓ `<bindingTemplate>` spécifie des informations techniques, telles que l'URI du service. Un `<bindingTemplate>` est un sous élément de `<businessService>`.
 - ✓ `<tModel>` définit des structures d'informations techniques telles que l'interface WSDL, les taxonomies industrielles, les ontologies, etc. Un élément `<tModel>` peut être réutilisé par plusieurs éléments `<bindingTemplate>`.

Le protocole d'utilisation de l'UDDI offre trois fonctions de base :

- ❖ **Publish** : permet de publier un nouveau service.
- ❖ **find** : permet d'interroger l'annuaire UDDI.
- ❖ **bind** : permet d'établir une connexion entre l'application cliente et le service.

4.3.6 La couches transversales (Sécurité, Transactions, QoS) :

Ces couches rendent viable l'utilisation effective des services dans le monde industriel.

Ce sont les aspects concernant la qualité de services : [2]

- ↪ **La sécurité** : La gestion de la sécurité est actuellement le frein le plus important à la mise en place d'architectures distribuées à base de services Web. Elle représente la normalisation des moyens permettant 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'authentification, 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 il veut 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.

↳ **Management** : Est le processus chargé de surveiller et de contrôler de manière proactive le comportement d'un service ou d'un ensemble de services.

5. Les avantages et inconvénients des services Web :

5.1 Les Avantages des Services Web: [16,17]

L'idée essentielle derrière les services Web est de partager les applications et les programmes en un ensemble d'éléments réutilisables appelés service, de sorte que, chacun de ces éléments effectuent une tâche principale et efficace, afin de faciliter l'interopérabilité entre tous ces services Web.

- Les services Web fournissent l'interopérabilité entre divers logiciels fonctionnant sur diverses plates-formes.
- Permettent de profiter de différents environnements et langages de développement par une publication, localisation, description et une invocation via XML.
- Les services Web sont très flexibles, indépendants des langages de programmation et des systèmes d'exploitation.
- Les protocoles et les formats de données sont au format texte dans la mesure du possible, facilitant ainsi 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.

5.2 Les Inconvénients des Service Web[18] :

- La sémantique n'est pas prise en charge de façon efficace car le WSDL décrit les services de manière syntaxique.
- Les services Web ont de faibles performances par rapport à d'autres approches de l'informatique répartie telles que le RMI, CORBA, ou DCOM.
- En l'utilisation du protocole HTTP, les services Web peuvent contourner les mesures de sécurité mises en place à travers les firewalls.
- Ils ne sont pas sécurisés à 100 %.

6. Quelques domaines d'application de services Web :

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


- L'application des services Web est multiple, autant dans les domaines du B2B, B2C que pour les domaines de gestion de stock, etc.
- B2C (Business to Consumer) : qualifie une application, un site internet destiné au grand public.
- B2B (Business to Business) : qualifie une application, un site internet destiné au commerce professionnel à professionnel [6].

7. Conclusion :

Ce chapitre couvre le développement des applications orientées services. Nous avons présenté les principes de l'architecture SOA, par la suite nous avons introduit les services Web comme étant des technologies standards qui implémentent SOA en permettant d'interagir, de gérer et d'automatiser plus facilement et plus rapidement les processus métier intra-entreprise et inter-entreprises en échangeant des informations au format XML. Ensuite nous avons introduit quelques protocoles et fonctionnalités de services qui permettent l'intégration et la réutilisation des services dans diverses applications.

Dans le prochain chapitre nous allons présenter un état de l'art sur la qualité des services web (QoS).

Chapitre 2



Qualité de service
QoS

1. Introduction :

Avec l'augmentation croissante du nombre de fournisseurs de Web services, il arrive très fréquemment que de nombreux Web services répondent à un même besoin fonctionnel, mais qui offrent des propriétés de qualité de services différentes, une sélection doit être alors réalisée pour déterminer quels sont les Web services pertinents qui satisferaient les besoins d'un utilisateur.

Dans ce chapitre nous présentons quelques notions fondamentales dans le paradigme de QoS et la sélection à base des propriétés fonctionnelles et non fonctionnelles.

2. Qualité de service :

Le terme QoS ou QdS (acronyme de “Qualité of Service “, en français “Qualité de Service”).

La qualité de service d'un service Web est définie comme étant une combinaison de plusieurs critères qui peuvent être quantitatifs tels que le temps de réponse, la disponibilité, ... ou qualitatifs tels que la sécurité, la disponibilité, l'accessibilité, la fiabilité, etc. Ces paramètres peuvent être alors considérés comme un critère de choix lorsqu'on à sélectionner parmi plusieurs services web découverts ceux qui respectent les contraintes imposées.

Cependant, la plus part des études proposées utilisent une succession d'évaluations des différents critères de qualité de service (ou critères non fonctionnels) dans le but d'attribuer un niveau de qualité de service global et ainsi sélectionner le **meilleur** service.[28]

2.1. Différents paramètres de qualité d'un service :

Les paramètres de qualité d'un service peuvent être nombreux et variés.

Ils prouvent être deux types :

- Des paramétrés quantitatifs : peuvent être évalué ou estimé de manière numérique. Exemple : coût de service, le temps max d'exécution ..., etc.
- Des paramétrés qualitatifs : ne peuvent pas être évalué de manière numérique. Ils sont plutôt liés à une appréciation (non valuée) du service rendu.[27]

2.2. Critères de Qualité de Service :

Jusqu'à les années récentes les services web ne prennent pas en considération les qualités de services non-fonctionnelles, ils prennent seulement les qualités fonctionnelles.

Chapitre2 :« Qualité de service (QoS) »

Les propriétés de qualités de services web (QoS) sont constituées de deux types de propriétés : Propriétés fonctionnelles et propriétés non fonctionnelles [20].

Plusieurs critères de qualités de services sont décrits dans différents travaux les propriétés de QoS les plus représentatives sont présentées comme suit :

- 1) **Disponibilité** (Availability) est la probabilité que les ressources, les services soient disponibles pour les parties autorisées en tout temps ou pour le pourcentage de temps pendant lequel le service fonctionne [21].
- 2) **Fiabilité** (Reliability) est la capacité d'un service à remplir ses fonctions requises dans les conditions indiquées pour une période de temps déterminée [22].
- 3) **Débit** (Throughput) est le nombre de demandes de service complétées sur une période de temps [23].
- 4) **Temps de Réponse** (Response time) la durée entre un utilisateur de service qui envoie une demande et reçoit une réponse [24].
- 5) **Compatibilité** (Interoperability) la compatibilité d'un service Web pour intégrer un ensemble de normes données.
- 6) **Accessibilité** (Accessibility) est l'aspect qualité d'un service qui représente le degré auquel il est capable de répondre à une demande de service Web. Il peut être exprimé comme une mesure de probabilité indiquant le taux de réussite ou les chances d'une instanciation de service réussie à un moment donné [25].
- 7) **Prix d'exécution** (Execution price) le cout à payer pour consommer le service, ce coût peut être fourni par le fournisseur du service [26].
- 8) **Reputation** le taux moyen du service déclaré par le client ; il est basé sur le service client, Il dépend principalement de l'expérience de l'utilisateur final de l'utilisation du service s. Différents utilisateurs peuvent avoir des opinions divergentes sur le même service.

Latency Execution time Response time Throughput Transaction time Availability Integrity Capacity Robustness	Security Interoperability Competence Stability Supported standard Execution price Accuracy Accessibility Autres QoS
---	---

Figure 2.1 : Le modèle QoS .

3. Modèles QoS existants :

Le groupe de travaille Architecture des Service Web W3C travaillant sur les architectures des services web a identifié et décrit un ensemble des paramètres des QoS pour les services web [27] à savoir :

3.1 Paramètre liée au Temps d'exécution :

- ❖ La performance qui englobe :
 - **Débit – capacité d'exécution:** Le nombre de requêtes accomplies par le service pendant une période de temps.
 - **Temps de réponse:** Le temps maximum garanti demandé pour compléter une requête du service.
 - **Le Temps d'exécution (Latence):** Temps pris entre l'arrivée de la requête du service et la réponse émise par le service. [27]
- ❖ **la Fiabilité:** La capacité d'un service d'exécuter ses fonctions dans des conditions indiquées dans une période de temps spécifié . Elle peut être mesurée par :
 - **MTBF:** “Mean time between failure” - Temps moyen entre pannes.
 - **MTF:** “Mean Time to Failure” - Temps moyen par panne.
 - **MTTT:** “Mean Time To Transition” – Temps moyen pour la transition. Il est très lie à la disponibilité
- ❖ **la Capacité:** Limite de demandes concourantes pour une performance garantie.
- ❖ **Traitement d'exception :** – Puisqu'il n'est pas possible pour le concepteur du service de spécifier tout les résultats possibles et alternatifs (particulièrement avec de divers cas spéciaux et possibilités imprévues), des exceptions peuvent être attendues.

Le traitement d'exception est comme le service traite ces exceptions. Il peut être d'une manière brutale ou appropriée.
- ❖ **L'Exactitude:** Définit le taux d'erreur produit par le service.

❖ **La Disponibilité** : Elle est la probabilité que le système soit actif. Elle est liée à la fiabilité.

Elle peut être mesurée comme :

$$A = \frac{\text{tempsActif}}{\text{tempsTotal}}$$

$$A = \frac{\text{tempsActif}}{(\text{tempsActif} + \text{tempsInactif})}$$

< tempsActif > est le temps total pendant lequel le système a été actif durant la période mesurée.

< tempsInactif > est le temps total pendant lequel le système a été inactif durant la période mesurée.

< tempsTotal > est le temps mesuré total, il est la somme de < tempsActif > et < tempsInactif >.

3.2 Paramètre de QoS liée aux transactions :

Intégrité : Les transactions peuvent être groupées dans une unité pour garantir l'intégrité des données opérées par ces transactions. L'unité peut être réussie si toutes les transactions dans l'unité s'engagent, ou toutes les transactions retournent à l'état original en cas d'échec de transaction.

C'est décrit par les propriétés ACID :

- l'Atomicité (exécute entièrement ou pas du tout).
- la consistance (maintient l'intégrité des données).
- l'isolement (des transactions individuelles exécutées comme si aucune autre transaction n'est présente).
- la durée (les résultats sont persistantes).

3.3 Paramètre de QoS liée à la sécurité :

Il mesure la fiabilité et la sécurité de mécanismes mis en œuvre.

- **Authentification** - Comment le service authentifie-t-il des principaux (des utilisateurs ou d'autres services) qui peuvent avoir accès au service et des données ?
- **Autorisation** - Comment le service autorise-t-il des principaux pour que seulement eux puissent avoir accès aux services protégés ?
- **Confidentialité** - Comment le service traite-t-il les données, pour que seulement les principaux autorisés puissent avoir accès ou modifier les données ?
- **Responsabilité** - le fournisseur peut-il être responsable pour ses services ?
- **Cryptage des données** - Comment le service chiffre-t-il les données ?

3.4 Paramètre de (QoS) liée à la gestion de la configuration et coût :

- **Coût** - C'est une mesure du coût impliqué dans la requête du service.
- **Etat complet** - une mesure de la différence entre le jeu indiqué de caractéristiques et le jeu mis en oeuvre de caractéristiques.
- **Cycle de stabilité/changement** - une mesure de la fréquence de changement lié au service en termes de son interface et-ou mise en oeuvre.[27]

4. La sélection des services web :

Le domaine de l'informatique a beaucoup évolué depuis ces dernières années. L'émergence de l'approche à services a été accompagnée par une augmentation exponentielle du nombre de services et des fournisseurs. Avec la sélection des services web, on cherche à choisir parmi plusieurs services similaires, le meilleur service en termes de qualité fonctionnelle et qui répondent au mieux aux critères exigés par l'utilisateur.

Dans cette phase, le contexte est intégré entre l'utilisateur et l'annuaire. Lors de la recherche des Services Web élémentaires, certains travaux intègrent le contexte d'utilisation afin de trouver les Services Web les mieux adaptés à la demande de l'utilisateur. Le système peut avoir à sa disposition un ensemble de Services Web qui peuvent répondre de manière fonctionnelle à la demande. La technique de sélection mise alors en jeu qui consiste à sélectionner le service web qui correspond au mieux au contexte.

Une technique de sélection consiste en un ensemble d'instructions à suivre pour sélectionner un service Web au moment de l'exécution, en fonction des informations disponibles pour le système.

Le processus de sélection de Services Web est basé sur trois étapes : l'expression de la requête, la recherche de services et la validation du résultat de la recherche par l'utilisateur.

- **L'expression de la requête** : l'utilisateur lance sa requête. Afin de permettre à l'utilisateur de découvrir les Services Web pertinents par rapport à son contexte.
- **La recherche de Services Web** : la recherche est effectuée par un processus de correspondance entre les différents critères de qualité et les services représentant la requête.
- **La validation de la recherche par l'utilisateur** : le résultat de l'étape précédente est retourné à l'utilisateur qui déclare s'il est satisfait de ce résultat. Si l'utilisateur n'est pas satisfait, le processus de recherche est renouvelé. [28]

4.1 Propriétés fonctionnelles et non fonctionnelles dans les Web services :[28]

En effet les critères fonctionnels et non fonctionnels qui définissent le comportement des services de manière direct. Ainsi deux services offrant les mêmes fonctionnalités dits services similaire.

Par exemple un service *Température* et un service *Heat* qui offrent des fonctions de mesure de température peuvent se différencier par leurs propriétés non fonctionnelles. Par exemple le service *Température* a un temps d'exécution inférieur au service *Heat* alors que ce dernier dispose de propriétés de sécurité tel que la confidentialité [4].

4.1.1 Les propriétés fonctionnelles :

Par ailleurs, on peut définir les propriétés fonctionnelles comme un ensemble des opérations que peut fournir un service. Qui apparaissent sous la forme de couplet (entrées/sorties). Dans ce cas, les services sont vus comme une fonction ayant des paramètres.

4.1.2 Les propriétés non fonctionnelles :

On peut définir les critères non fonctionnels comme un ensemble des qualités de service ce qui revient à définir la capacité de service à fonctionner en terme de conditions tel que disponibilité, performance, coût d'invocation, fiabilité, etc.

Le groupe W3C propose un guide pour définir les QoS et leurs métriques. Le guide proposé regroupe les QoS en 4 catégories : les attributs de performance, les attributs qui assurent la sûreté de fonctionnement "Dependability", les attributs de sécurité, et les attributs spécifiques au domaine d'application [4].

Le tableau suivant illustre cette catégorisation [4] :

Catégorie de QoS	Attributs de QoS
Performance	Temps d'exécution, Débit, Temps de réponse et Latence.
Sureté de fonctionnement	Disponibilité, Accessibilité, Fiabilité, Capacité, ...
Sécurité	Authentification, Non-Répudiation, Confidentialité, Intégrité, Identification.
Attributs spécifiques aux domaines d'application	Prix, Mode de paiement, Taux de pénalité, ...

Figure 2.2: Guide proposé par W3C pour les QoS et leurs métriques .

4.2 Sélection basée sur les besoins fonctionnels et non fonctionnels :

4.2.1 Sélection basée sur les besoins fonctionnels :

Dans ce cas la sélection d'un service se base sur la signification syntaxique exacte entre l'interface de service qui est publié par les fournisseurs et l'interface de l'utilisateur.

Nous remarquons que la recherche syntaxique est plus faible car elle se base sur des mots clés qui rarement donne un résultat efficace, alors cette approche présente un manque qui ne garantit pas la qualité de service.

4.2.2 Sélection basée sur les besoins non fonctionnels :

La sélection des services web à base de qualité de service est la meilleure solution qui permet de sélectionner le meilleur service parmi un ensemble des services offrant les mêmes fonctionnalités. Ce sont les propriétés non fonctionnelles qu'on les appelle qualité de Service (QoS).

5. Conclusion:

Dans ce chapitre nous avons présenté un état de l'art sur les paramètres de qualité des services et les différentes méthodes de sélection des services web à base de QoS.

Chapitre 3



*Approches de sélection
De service web*

1. Introduction aux Approches de sélection de services web :

La sélection automatique de services web, a fait l'objet de plusieurs travaux, de façon générale on distingue 03 grandes classes : la sélection mono objective, La sélection multi objective, et la sélection hybride (mono et multi objective, par la suite on va détailler chacune des approches en donnant quelques algorithmes utilisés par ces dernières

2. La selection mono-objective:

Cette catégorie suppose que les m valeurs de qualité de service soient agrégés en un seul score, c.-à-d. On considère une seule fonction objective qui associe des poids aux différents attributs de QoS. Cette dernière est aussi divisée en 03 sous approches (classes) : la sélection locale, la sélection globale et hybride c.-à-d. globale et locale.

A. La sélection globale :

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

i. L'optimisation globale exacte :

Elle se base sur la programmation entière ou la programmation par contrainte, ou les énumérations exhaustives, ces méthodes donnent des résultats optimaux. [37]. Ces approches ont un temps d'exécution exponentiel, les travaux de [38] utilisent les techniques de programmation entière et mixtes (MIP) [39] pour trouver la composition optimale des services.

Dans la même optique [40] Étendent le modèle de programmation linéaire afin d'inclure les contraintes locales. Dans ce modèle, les contraintes globales sont exprimées sur la composition entière, et par l'utilisateur final, tandis que les contraintes locales peuvent être spécifiées par le concepteur de la composition au niveau des classes.

[41] 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. [42] Proposent une sélection globale de composition de services en adoptant 04 structures de flux de contrôle : parallélisme, séquence, choix conditionnels, et les boucles, la requête est formalisée en BPEL, elle possède 05 critères de QoS.

L'algorithme est nommé « qssac », il peut donner un résultat proche de l'optimal (mais les auteurs n'ont pas d'expérimentations sur le degré d'optimalité).

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

Selon [43] la programmation entière est efficace si le nombre de service l est petit par contre si l dépasse une certaine limite (quelques milliers) alors le temps d'exécution n'est plus raisonnable.

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

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

En [44] qui est l'un des premiers algorithmes proposés pour le problème de sac à dos (version MMKP). L'auteur propose une heuristique UHE pour sa résolution, UHE utilise une mesure appelée la consommation des ressources agrégés pour mettre à jour un élément de chaque groupe à chaque tour de sélection.

En [45] Modifient l'heuristique en créant M-UHE, ils proposent une étape de prétraitement de trouver une solution réalisable et une étape de post-traitement pour améliorer la valeur totale de la solution.

En [46], proposent une autre heuristique, C-UHE, et évaluent sa performance et son optimalité contre plusieurs heuristiques, y compris la M-UHE. Les résultats montrent que la C-UHE est meilleur par rapport à M-UHE en termes de temps d'exécution. Cependant, les expériences montrent également que M-UHE est la meilleure en termes de degré d'optimalité, tandis que l'optimalité de C-UHE diminue à mesure que le nombre de candidats par classe augmente. Les expériences montrent que l'algorithme C-UHE fonctionne mieux dans le cas où l'objectif à maximiser (par exemple, la valeur d'utilité de la composition de services) n'est pas proportionnel aux besoins en ressources (c.-à-d. les valeurs de QoS). Et de ce fait elle ne peut être appliquée pour la sélection de services composé à base de qualité (puisque l'utilité dépend des qualités de service).

Une version modifiée de l'algorithme M-UHE, appelée WS-UHE, est conçue par [47]. La complexité temporelle de WS-UHE est polynomiale. En dépit de l'amélioration significative de ces algorithmes par rapport aux solutions exactes, les deux algorithmes ne sont pas scalables (c.-à-d. si le nombre de services Web croît de manière dramatique alors le temps d'exécution n'est plus temps réel). De plus, l'algorithme WS-UHE n'est pas adapté au caractère distribué des services web. Ceci est dû au fait que WS-UHE applique des améliorations sur une composition de services dont les valeurs de QoS proviennent de plusieurs facilitateurs. (Chaque facilitateur gère une classe donnée).

L'optimisation par essaim particulière (SPO) est utilisée dans une approche de sélection de services web pour faire aussi une optimisation mono-objective et globale. C'est une nouvelle classe des méta-heuristiques proposée en 1995 par Kennedy et Eberhart. [48], TRIBES est un algorithme SPO créé par M.

Clerc dans [49]. Cet algorithme permet d'avoir un paramétrage automatique de l'optimisation par essaim particulière. La sélection clonale un autre algorithme utilisé pour faire une optimisation mono-objective et globale sur l'espace de recherche dans une approche de sélection de services web. Dans le même but de résoudre le problème de sélection de services web, et toujours des idées inspirées des mécanismes naturels qui ont été exploitées pour développer des heuristiques inspirées de la nature.

Le système immunitaire artificiel (SIA) est un paradigme récent qui tente de capturer des caractéristiques intéressantes des systèmes immunitaires naturels, comme la mémorisation, la reconnaissance de formes, l'apprentissage... La théorie de la sélection clonale a été utilisée comme source d'inspiration pour le développement des SIA qui effectuent des tâches d'optimisation, Castro et Von Zuben [50] ont développés l'un des algorithmes de SIA inspiré de la sélection clonale les plus populaires et largement utilisés appelé « Clonalg », qui a été utilisé pour effectuer les tâches de filtrage et d'optimisation.

B. La sélection locale :

Elle consiste à filtrer un seul service de chaque classe en utilisant une fonction objective et indépendamment des autres classes, ensuite elle compose les n résultats qui correspondent aux n classes. Cette approche possède une complexité linéaire $O(n)$, en plus elle est fortement adaptée aux environnements distribués, en effet la gestion de la QoS (mesures, mise à jours ...) est faite par des facilitateurs (brokers) distribués.

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

Dans [52] les auteurs considèrent la sélection de services web en prenant en compte les préférences sur les sorties de services, pour cela ils appliquent un tri local (pour chaque classe) en adoptant la notion de fuzzy dominance, cette dernière permet la comparaison de 02 services. En fin ils retiennent les K premières solutions. Les auteurs ne gèrent pas les contraintes globales.

C. La sélection hybride :

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

Alrifai en [53] montre que plusieurs chercheurs ont proposé l'utilisation de la programmation de nombre entier mixte [39] pour résoudre le problème de composition de service web à base QoS. Des variables binaires de décision sont employées dans le modèle pour représenter les candidats de service. Un service candidat s_{ij} est choisi dans la composition optimale si son variable x_{ij} correspondant est placé à 1 dans la solution du modèle et jeté autrement.

Pour inclure les variables de décision, le problème de résoudre le modèle peut être formulé comme problème de maximisation de valeur de service globale, sujet aux contraintes globales de QoS, tout en satisfaisant les contraintes d'attribution sur les variables de décision.

iii. La sélection multi-objective :

Un grand nombre d'approches existent pour résoudre les problèmes d'optimisation multi-objective. Certains d'entre eux utilisent la connaissance qu'ils ont au sujet du problème pour donner des préférences à quelques objectifs, de ce fait déviant l'aspect multi-objective. D'autres donnent à tous les objectifs le même niveau d'importance,...

Parmi ces approches, nous devrions distinguer deux catégories : approches non-Pareto et Pareto. Les approches de Non-Pareto ne traitent pas réellement le problème comme problème multi-objective. Elles essayent de le convertir en problème mono-objectif.

D'autre part, les approches de Pareto ne transforment pas les objectifs du problème, mais essayent de les optimiser simultanément.

L'optimisation à base de skyline « the skyline based optimization » peut être manipulée en employant les techniques de base de données multiples [54], par exemple nous pouvons employer l'algorithme de clivage et conquérir, l'algorithme Bitmap, l'algorithme basé sur index (B tree, Hash table), et l'algorithme de plus proche voisin (R tree).

Il existe plusieurs travaux qui tiennent compte des préférences d'utilisateur pour sélectionner les k top skylines dominants [55] certains d'entre eux utilisent la théorie des ensembles flous pour modéliser les préférences et le rapport de dominance, les autres emploient le concept de parteo-dominance pour ranger les services web.

L'utilisation des algorithmes évolutionnaires pour résoudre des problèmes Multiobjective a été motivée principalement en raison de la nature d'AEs basée sur la population qui permet la génération de plusieurs éléments du Pareto ensemble optimal dans une seule course. Les Algorithmes Évolutionnaires multi-objective (MOEA) figurent parmi les méthodes de résolution les plus puissantes pour l'optimisation multi-objective [56]. MOEA tiennent compte des objectifs contradictoires et laissent trouver un ensemble des solutions non dominées.

iv. La sélection mono et multi-objective :

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

En [57] utilisent une approche de sélection hybride, telle qu'ils effectuent une recherche multi objective pour chaque classe, en groupant hiérarchiquement les services de chaque classe en choisissant les services skylines car seulement ces services ne sont pas dominés par n'importe quel autre service et sont valides comme candidats pour la composition en utilisant l'algorithme intelligent Kmeans, ensuite elle continue avec une recherche mono-objective sur les représentants des groupes en utilisant le standard MIP pour décomposer les contraintes de QoS en contraintes locales, qui sont alors employées pour choisir efficacement le meilleur service de chaque classe. Les variables dans le modèle MIP de l'approche hybride représentent les niveaux locaux de QoS de chaque classe de service plutôt que les candidats

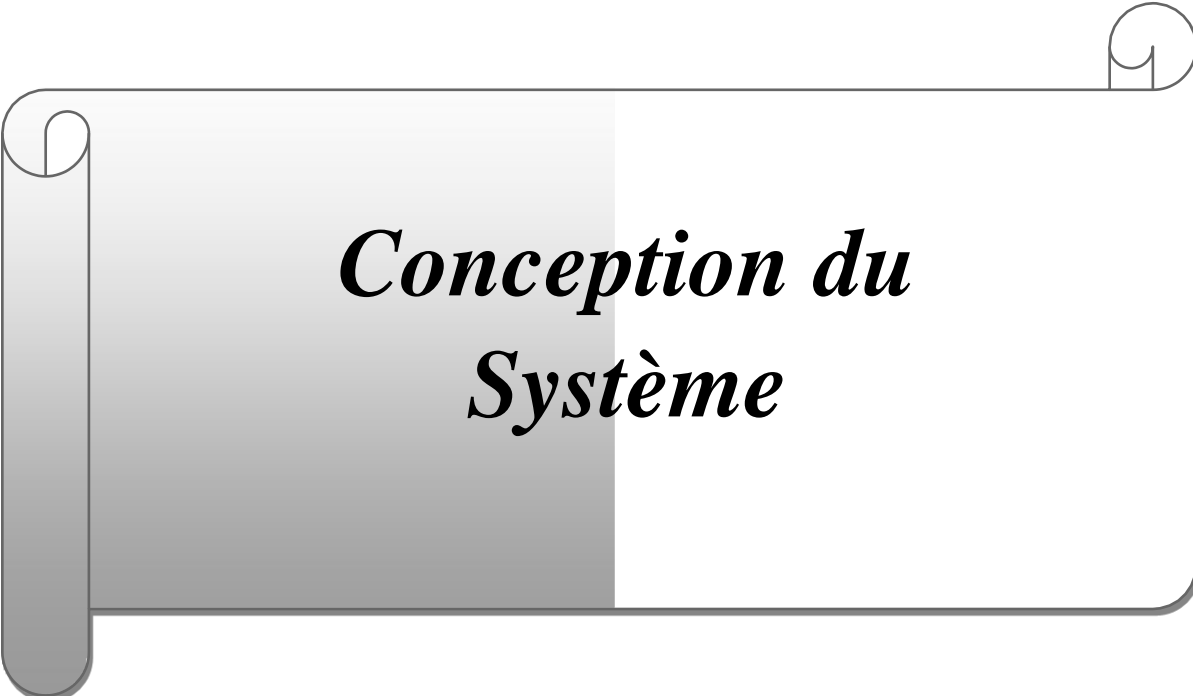
Chapitre 3: « Approches de sélection de service web »

réels de service, en le rendant scalable au nombre de service candidats que l'approche d'optimisation globale.

Conclusion

Dans ce chapitre on a présenté un état de l'art des différentes approches de sélection de services Web à base QoS.

Chapitre 4



*Conception du
Système*

1. Introduction:

Ce chapitre fournit une vue détaillée des besoins fonctionnels de notre application, scenarios des cas d'utilisations, diagrammes de séquence, et diagramme d'activité réalisés avec le langage de modélisation UML, puis un scenario global qui résume le fonctionnement de l'application, et enfin le diagramme de classe qui illustre les relations entre les différentes classes de l'application.

2. La Méthode TOPSIS :

TOPSIS a été développé par Hwang et Yoon en 1981 comme une alternative à la méthode ELECTRE et peut être considéré comme l'une de ses variantes les plus largement acceptées. Le concept de base de cette méthode est que l'alternative sélectionnée doit avoir la distance la plus courte de la solution idéale et la distance la plus éloignée de la solution idéale négative dans un certain sens géométrique [30].

La méthode TOPSIS suppose que chaque critère a une tendance à l'utilité monotone croissante ou décroissante. Il est donc facile de définir les solutions idéales et négatives idéales. L'approche distance euclidienne a été proposée pour évaluer la proximité relative des alternatives à la solution idéale. Ainsi, l'ordre de préférence des alternatives peut être obtenu par une série de comparaisons de ces distances relatives [30].

La méthode TOPSIS convertit d'abord les différentes dimensions de critères en critères non dimensionnels comme c'était le cas avec la méthode ELECTRE. A titre de remarque, dans les méthodes ELECTRE et TOPSIS, la distance euclidienne représente des hypothèses plausibles. D'autres mesures de distance alternatives pourraient également être utilisées, auquel cas il est possible d'obtenir des réponses différentes pour le même problème [30].

Cependant, il est raisonnable de supposer ici que pour les critères de prestations, le décideur veut avoir une valeur maximale parmi les alternatives. Pour les critères de coût, le décideur veut avoir une valeur minimale parmi les alternatives [30].

Généralement la méthode TOPSIS suppose donc deux alternatives artificielles, à savoir [31] :

- l'alternative idéale (« ideal alternative ») : c'est l'alternative qui possède la meilleure évaluation (ou valeur) par rapport à tous les critères considérés [31].
- et la pire alternative (« negativeideal alternative ») : c'est l'alternative qui possède la pire évaluation (ou valeur) par rapport à tous les critères considérés [31].

Les alternatives artificielles sont construites à partir des valeurs des attributs des alternatives existantes, en prenant en compte, respectivement, les meilleures et les pires valeurs de chacun des attributs. La méthode TOPSIS consiste donc à sélectionner l'alternative la plus proche de l'alternative idéale et la plus loin de la pire alternative [31].

La méthode TOPSIS, tout comme les autres méthodes MCDM (Méthodes d'aides à la décision multicritère) , se base sur la matrice de décision pour effectuer l'agrégation des résultats des évaluations. La représentation d'une matrice de décision est présentée à la figure suivante.

Alternatives	Critères					
	C_1	C_2	...	C_j	...	C_n
	w_1	w_2	...	w_j	...	w_n
A_1	a_{11}	a_{12}		a_{1j}		a_{1n}
A_2	a_{21}	a_{22}		a_{2j}		a_{2n}
...	...					
A_i	a_{i1}	a_{i2}		a_{ij}		a_{in}
...	...					
A_m	a_{m1}	a_{m2}		a_{mj}		a_{mn}

Figure 4.1: Représentation d'une matrice de décision [31].

La matrice de décision, telle que définie à la figure précédente, est composée des éléments suivants [31] :

- **Ensemble des alternatives potentielles** : $A = \{a_1, a_2, \dots, a_i, \dots, a_m\}$, tel que m est le nombre d'alternatives potentielles (a_i où $i=1,2,\dots, m$) ;
- **Différents critères d'évaluation** : $C = \{c_1, c_2, \dots, c_j, \dots, c_n\}$, tel que n est le nombre des critères d'évaluation (c_j où $j=1,2,\dots, n$) ;
- **Poids des critères** = $\{w_1, w_2, \dots, w_j, \dots, w_n\}$, tel que n est le nombre de critères (w_j où $j=1,2,\dots, n$) ;

➤ **Évaluations (ou jugements) :** $E_{ij} = \{e_{11}, e_{12}, \dots, e_{1j}, e_{21}, e_{22}, \dots, e_{2j}, e_{i1}, e_{i2}, \dots, e_{ij}, \dots, e_{m1}, e_{m2}, \dots, e_{mj}\}$, tel que e_{ij} représente l'évaluation de l'alternative a_i selon le critère c_j (e_{ij} où $i=1,2,\dots,m, j=1,2,\dots,n$).

La méthode TOPSIS suppose donc qu'il existe m alternatives ($a_i=1\dots m$) et n critères d'évaluation ($c_j=1\dots n$). Soit e_{ij} l'évaluation (i.e. la valeur) de chaque alternative par rapport à chaque critère, et soient $\mathbf{M} = (e_{ij})$ la matrice d'évaluation de $(m \times n)$ éléments, \mathbf{J} l'ensemble des critères avantageux (i.e. meilleur quand la valeur est grande, tel que la rentabilité) et \mathbf{J}' l'ensemble des critères désavantageux (i.e. meilleur quand la valeur est petite, tel que le temps de réponse), la méthode TOPSIS suit une démarche composée de plusieurs étapes principales [31].

Ci-dessous la description des différentes étapes [31] :

-Étape 1 : construire la matrice normalisée

Les valeurs des différents critères sont souvent exprimées dans des unités différentes (euro, kilomètre, kilogramme, etc.). La normalisation permet d'uniformiser ces valeurs, afin de pouvoir les comparer. La normalisation des valeurs produit la matrice \mathbf{R} et elle se fait comme suit :

$$r_{ij} = e_{ij} / \sum e_{ij}^2, \text{ telque : } i = 1\dots m; j = 1\dots n \quad (2.1)$$

-Étape 2 : construire la matrice normalisée pondérée

Soit w_j où $j=1,2,\dots, n$ le poids associé à chacun des critères. Il s'agit de multiplier chaque colonne de la matrice normalisée par le poids correspondant. La pondération des valeurs normalisées produit la matrice \mathbf{V} . Celle-ci se calcule comme suit :

$$v_{ij} = r_{ij} * w_j, \text{ telque : } i = 1\dots m; j = 1\dots n \quad (2.2)$$

-Étape 3 : Définir les solutions artificielles

L'alternative idéale est généralement une alternative fictive. Elle est définie comme suit :

$$A^* = \{v_1^*, \dots, v_n^*\} \quad v^* = \{max(v_{ij}) \text{ si } j \in J; min(v_{ij}) \text{ si } j \in J'\} \quad (2.3)$$

La pire alternative est généralement, aussi, une alternative fictive. Elle est déterminée comme suit :

$$A' = \{v'_1, \dots, v'_n\} \quad v' = \{max(v_{ij}) \text{ si } j \in J; min(v_{ij}) \text{ si } j \in J'\} \quad (2.4)$$

Sachant que « J » est l'ensemble des critères avantageux (i.e. meilleur quand la valeur est grande) et « J' » est l'ensemble des critères désavantageux (i.e. meilleur quand la valeur est petite).

-Étape 4 : calculer les distances Euclidiennes.

La méthode TOPSIS propose de calculer la distance Euclidienne de chaque alternative a_i (où $i=1,2,\dots, m$) à l'alternative idéale, comme suit :

$$S_i^* = \left[\sum (v_j^* - v_{ij})^2 \right]^{1/2}, \text{ telque } i = 1 \dots m \quad (2.5)$$

La distance euclidienne de a_i (où $i=1,2,\dots, m$) à la pire alternative est calculée comme suit :

$$S'_i = \left[\sum (v'_j - v_{ij})^2 \right]^{1/2}, \text{ telque } i = 1 \dots m \quad (2.6)$$

-Étape 5 : Calculer la proximité (ou la similarité) relative.

La proximité relative $C^*_{i=1..m}$ permet de comparer les différentes alternatives par rapport aux alternatives artificielles A^* et A' . Il est considéré comme étant le degré de préférence des m alternatives. Le calcul de $C^*_{i=1..m}$ se fait comme suit :

$$C_i^* = S'_i / (S_i^* + S'_i), \text{ telque } : i = 1, \dots, m \text{ et } 0 < C_i^* < 1 \quad (2.7)$$

Il s'agit de maximiser $C^*_{i=1..m}$: les alternatives dont les $C^*_{i=1..m}$ sont proches de 1 sont les plus proches de l'alternative idéale et les plus éloignées de la pire alternative. Par conséquent, lorsque $C^*_{i=1..m} = 1$ alors $a_i = A^*$, c'est-à-dire qu'il s'agit de la meilleure alternative. Si $C^*_{i=1..m} = 0$ alors $a_i = A'$, c'est-à-dire qu'il s'agit de la pire alternative.

-Étape 6 : classer les alternatives selon le degré de préférence.

La meilleure alternative peut désormais être décidé en fonction du degré de préférence $C^*_{i=1..m}$: les meilleures alternatives seront celles qui ont, d'une part, la plus courte distance de l'alternative idéale (i.e. $S^*_{i=1..m}$), et d'autre part, la plus longue distance de la pire alternative (i.e. $S'_{i=1..m}$).

Nous présentons, dans ce qui suit, un résumé de l'algorithme général de la méthode TOPSIS (figure 4.2 [29]).

```
TOPSIS ()

Entrée
m : nombre d'alternatives
n : nombre de critères
a[i] : les alternatives possibles
c[j] : les critères d'évaluation
w[j] : les poids pour chaque critère d'évaluation  $c_{j/j=1..n}$ 
e[i,j] : évaluation de chaque alternative a[i] suivant les critères c[j]
M : matrice de décision de (m×n) éléments
R : matrice de décision normalisée
V : matrice de décision normalisée pondérée

Sortie
ListeAlternative[i] : classement descendant des alternatives dont C* est
proche de 1

DEBUT
J= {cj/j=1..s} : s est le nombre des meilleurs cj/j=1..n quand la valeur est
grande
J'= {cj/j=1..t} : t est le nombre des meilleurs cj/j=1..n quand la valeur est
petite
M[i,j]=ConstruireM (a[i],m,c[j],n)
R[i,j]=ConstruireR (M)
V[i,j]=ConstruireV (R)
A*[j]=CalculerA*(V[i,j],J,J')
```

Figure 4.2 : La méthode TOPSIS [29].

3. Dominance :

Soit r un ensemble de points multidimensionnels et $P = (p_1, p_2 \dots p_n)$ et $(q_1, q_2 \dots q_n)$ deux points de r . La relation de dominance est une relation binaire entre deux éléments, si on considère p et q deux éléments de r . On dit que p domine q si et seulement si sur chaque dimension $p_i \geq q_i$ (Pour $1 \leq i \leq d$).

Les fonctions utilisées «Domin_Proba »: elle prend comme paramètre le service S_i et le service S_j , elle retourne une valeur qui représenté la probabilité dominance entre eux. «Degré»: elle s'incrémente en fonction de valeur retournée par Domin-proba. «Domin_Proba » : « dom »: elle comparer deux valeur sur chaque dimension (p_i, q_i) de service et le zéro «0» valeur inférieur ,et numéro supérieur égal un «1» etc...

$$\text{domin-proba}(S_i, S_j) = \sum_{l=1}^n \sum_{l'=1}^n |(S_i^l) > S_j^l)|, i, j = 1 \rightarrow 3, + |(S_i^l) < S_j^l)|, i, j = 3 \rightarrow 6$$

Chapitre 4: « Conception du système »

Domin-proba égal la total de dom «Trier_ordre_croissant»: elle a pour rôle de trier la liste des services par ordre croissant. Déroulement de l'algorithme

« score»: Elle comparer entre les valeurs de Domin-proba des services $S_{i,j}$, elle retourne une valeur qui représenté la probabilité plus grand déroulement de dom et en classé ordre croissant les scores « score»:

$$scor(S_{c_i}, S_{c_j}) = \sum_{l=1}^n |S_{c_i} \geq S_{c_j}| \quad i, j \rightarrow n$$

-nous effectuons plusieurs cycles du concours Condorcet (la boucle) de service.

-nous déterminons les gagnants de Condorcet qui sont retournés par la fonction « Dom-Proba», en localisant « Degré » pour sauvegarder l'incréméntation de score d'un service, tel que:

- L'incréméntation s'effectue à S_i lorsque la domination probabiliste de S_i par rapport à un autre service S_j est supérieure à la domination probabiliste de S_j par rapport S_i . Le cas d'inverse l'incréméntation s'effectue à S_j .
- L'incréméntation s'effectue à SC_i lorsque la domination probabiliste de domin-proba ou S_i par rapport à un autre service S_j est supérieure à la domination probabiliste de S_j par rapport S_i . Le cas d'inverse l'incréméntation s'effectue à S_j .

-nous effectuons plusieurs cycles du concours Condorcet (la boucle) de service.

- Si la dominance probabiliste de S_i par rapport à un autre service S_j est égale à la domination probabiliste de S_j par rapport à S_i dans ce cas nous calculons la distance entre le centre de service (par le biais des 5 mesures de similarité) et le 0 l'incréméntation s'effectue pour les services qui a la plus de distance que le deuxième service

-a la fin on trie le degre par ordre croissant.

-nous renvoyons les éléments TopK de la liste degré.

Ex : {min,min,max }

$S_1 = \{0.15, 0.18, 0.23\}$, $S_2 = \{0.50, 0.1, 0.22\}$.

$\text{Domin-Proba}(S_1, S_2) = 1$, $\text{Domin-Proba}(S_2, S_1) = 2$.

$SC(S_1) = 0$, $SC(S_2) = 2$.

4. Analyse globale :

Nous avons conçu notre outil comme un calculateur, classificateur de qualité de service, il aboutit à un tableau de service web classés selon ordre croissant des qualités de service . Ce calcul est effectué en tenant compte du poids de chaque paramètre de QoS ou QdS.

Services enregistrés dans la base de données de service. Il peut aussi ajouter, modifier et supprimer des services de la base de données. Paramètres du compte qui sont : Poids, peut être modifiés.

5. Besoin fonctionnel :

L'application utilisera un algorithme de classement pour rendre la liste des services en ordre croissant. Le système doit satisfaire les exigences de l'utilisateur, raison pour laquelle nos besoins vont être classés selon :

- Gestion des services :
 - ✓ Ajouter un service,
 - ✓ Modifier un service,
 - ✓ Supprimer un service,
- Modification les paramétrés de calcul :
 - ✓ Modifier le poids,
- Lancement du calcul:
 - ✓ Appliquer algorithme de classement (TOPSIS)
 - ✓ Appliquer une fonction de dominance
 - ✓ Filtrage et Recherche
 - ✓ Selection

6. Diagramme Cas d'utilisation

Le diagramme cas d'utilisation montre les différentes interactions fonctionnelles entre l'acteur et le système.

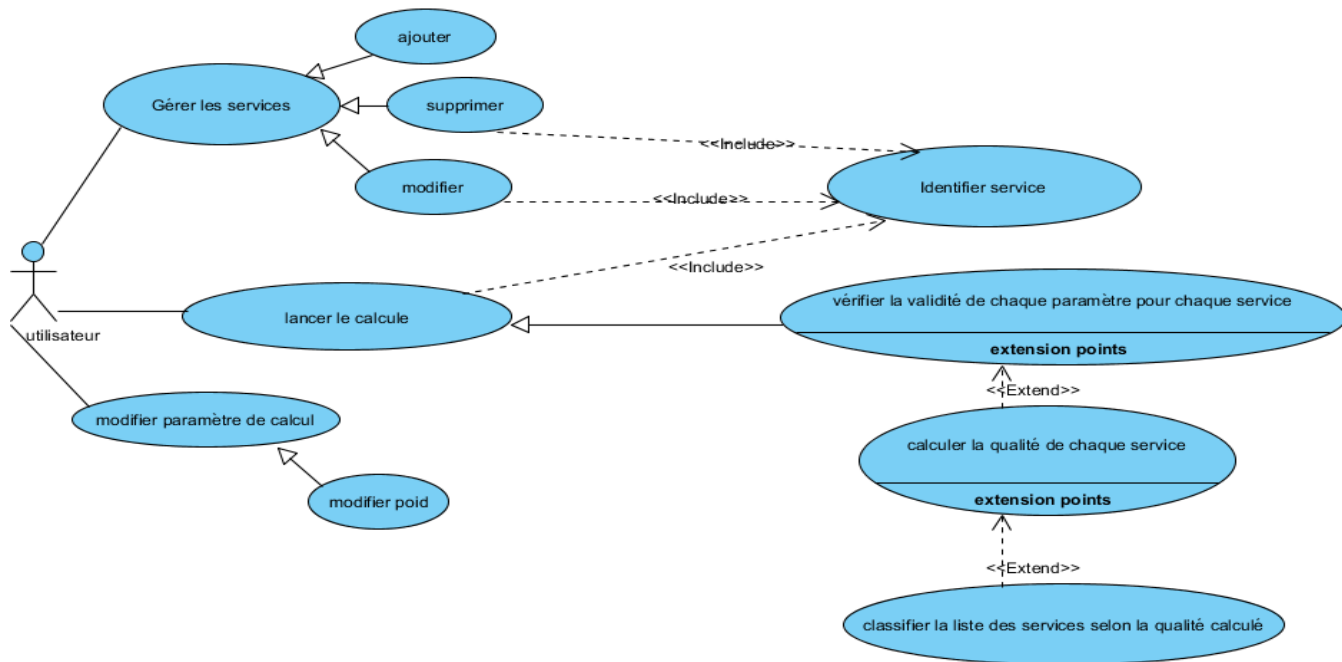


Figure 4.3 : Cas d'utilisation.

7. Scenario Global

Dans cette partie on va citer les scénarios qui peuvent être exécutés, selon le diagramme de cas d'utilisation illustré dans figure 4.3

7.1. 1^{er} scénario/l'ajout d'un service.

Titre : L'ajout d'un service

But : ajouter un nouveau service à la base de données.

Résumer : ajouter un nouveau service à la base de données pour pouvoir le comparer avec d'autre service.

Acteur : Utilisateur.

7.1.1. Scénario normal:

1. L'utilisateur accède à <Gérer les services >
2. Le système affiche le tableau des services
3. L'utilisateur sélectionne <ajouter un service>
4. Le système ajoute une ligne vide à remplir dans le tableau des services
5. L'utilisateur saisie les données du service
6. L'utilisateur enregistre le nouveau service ajouté
7. Le système ajoute le nouveau service à la base de données
8. Le système affiche la nouvelle liste des services

7.1.2.Scenario alternatif :

- A. L'utilisateur annule l'ajout du nouveau service
- 6.Quitté l'ajout.
- 7.Retour à2.

7.1.3.Diagramme de séquence :

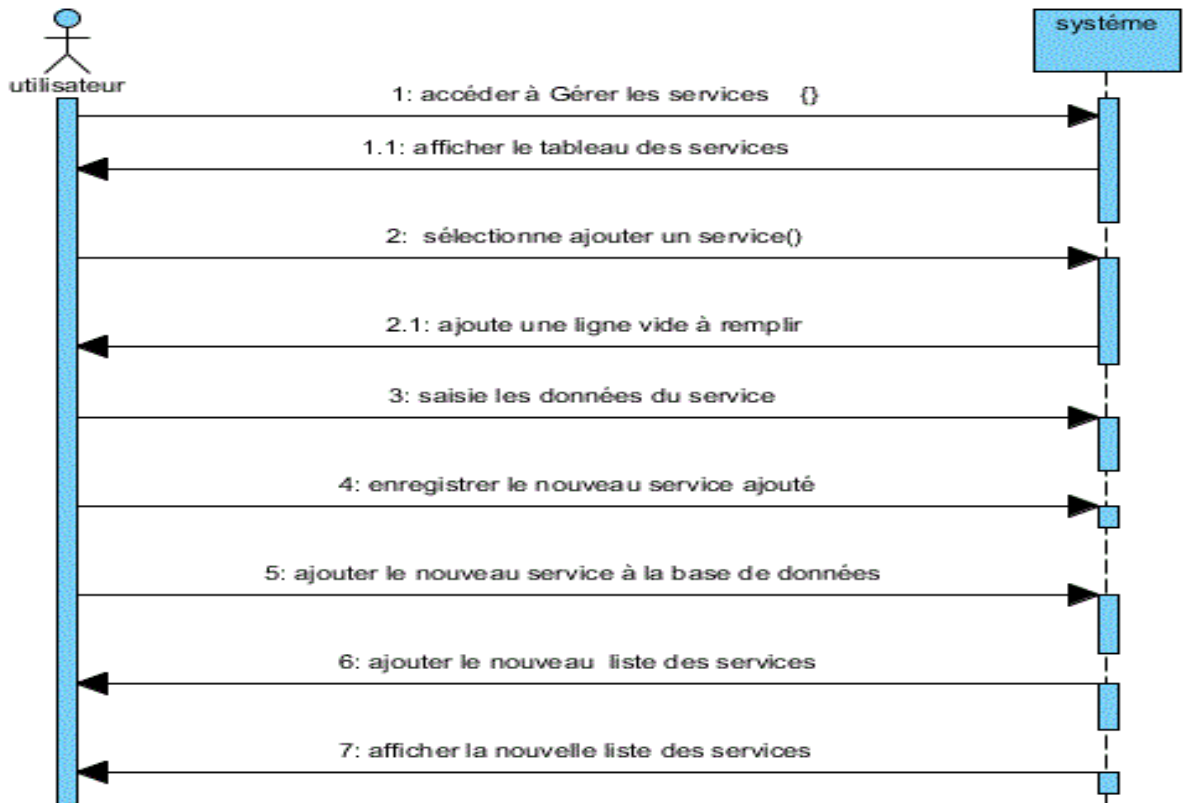


Figure 4.4 : Diagramme de séquence de l'ajout d'un service

7.1.4.Diagramme d'activité :

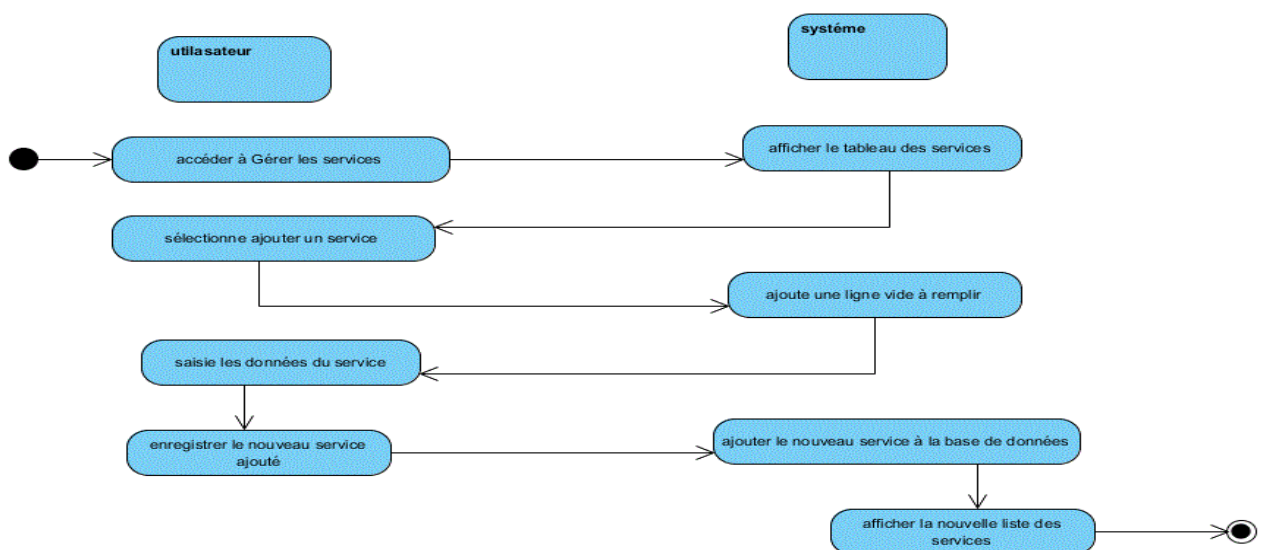


Figure 4.5 : Diagramme d'activité de l'ajout d'un service.

7.2. 2^{ème} scenario/la suppression d'un service.

Titre : la suppression d'un service.

But : supprimer un service dans la base de données.

Résumer : supprimer un service du tableau de la liste des services .

Acteur : Utilisateur.

7.2.1.Scenario normal :

1. L'utilisateur accède à <Gérer les services >.
2. Le système affiche le tableau des services.
3. L'utilisateur sélectionne le service à supprimer .
4. L'utilisateur appuie sur <supprimer>.
5. Le système retire le service supprimé.

7.2.2.Scenario alternatif :

A. L'utilisateur sélectionne aucun service.

4.Retour à2.

7.2.3.Diagramme de séquence :

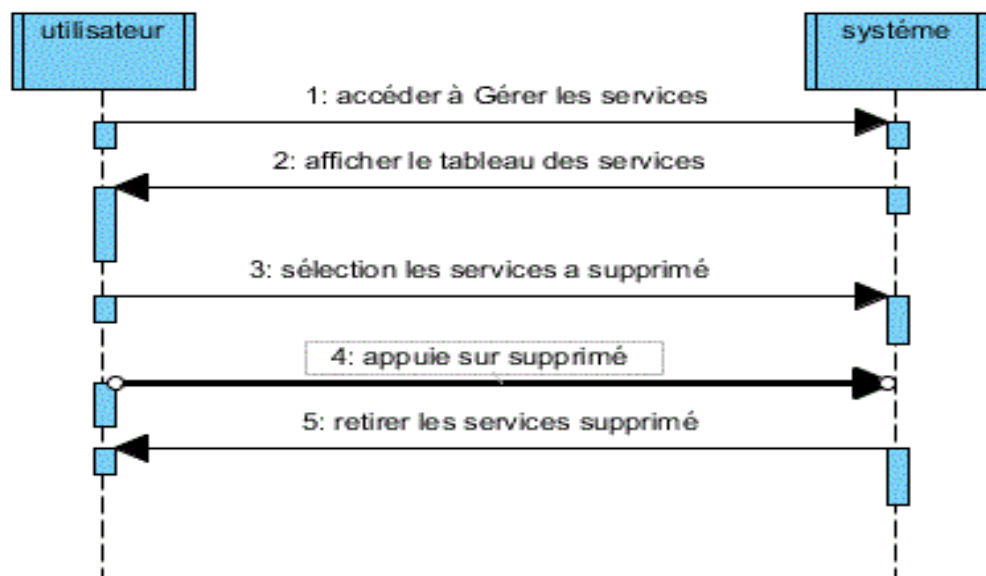


Figure 4.6: Diagramme de séquence de la suppression d'un service.

7.2.4. Diagramme d'activité :

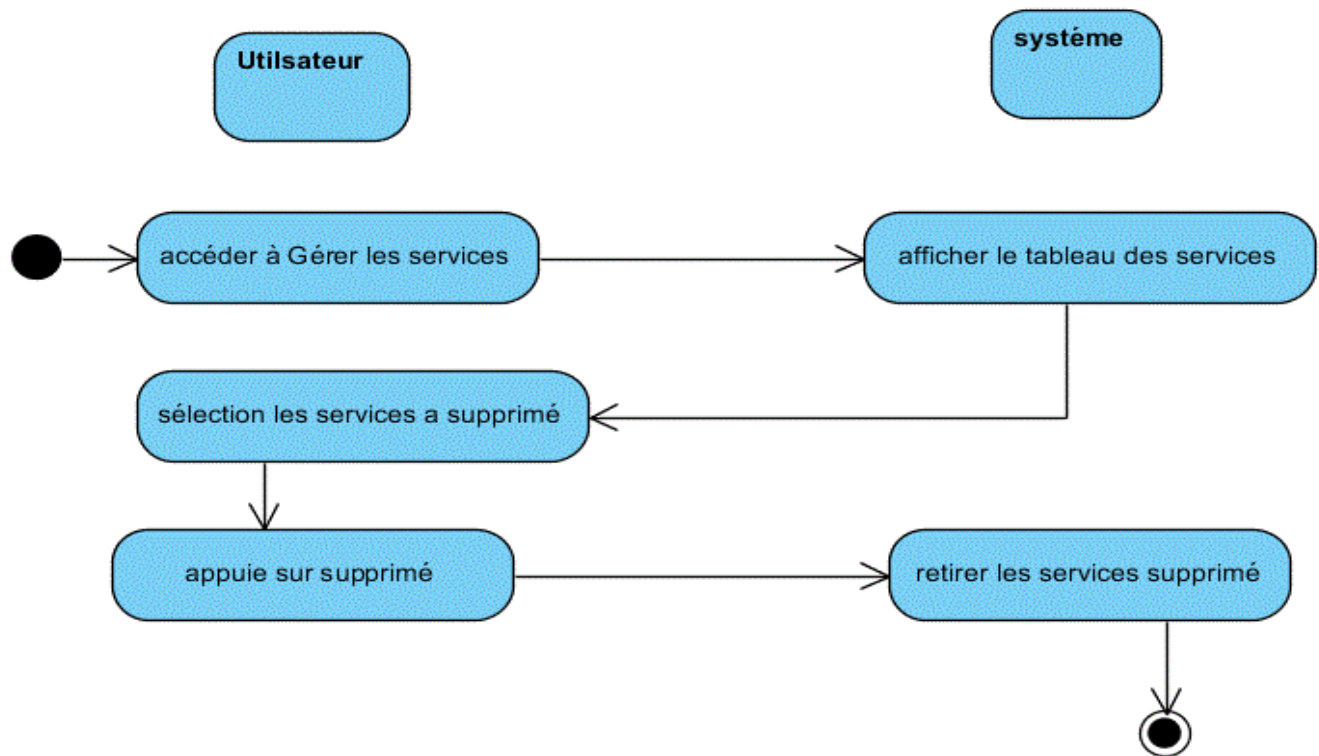


Figure 4.7 : Diagramme d'activité de la suppression d'un service.

7.3. 3^{ème} scenario/la modification d'un service.

Titre : la modification d'un service.

But : changer les paramètres d'un service

Résumer : changer les valeurs des paramètres concourant le servicesélectionne.

Acteur : Utilisateur.

7.3.1. Scenario normal :

1. L'utilisateur accède à <Gérer les services >
2. Le système affiche le tableau des services.
3. L'utilisateur appuie sur <Service à Modifier>.
4. Le système permet la modification des valeurs des paramètres du service sélectionné.
5. L'Utilisateur change les valeurs.
6. L'Utilisateur enregistre la modification .
7. Retour à2.

7.3.2.Scenario alternatif :

- A. L'utilisateur annule la modification du service.
- 6. Quitté la modification.
- 7. Retour à 2.

7.3.3.Diagramme de séquence :

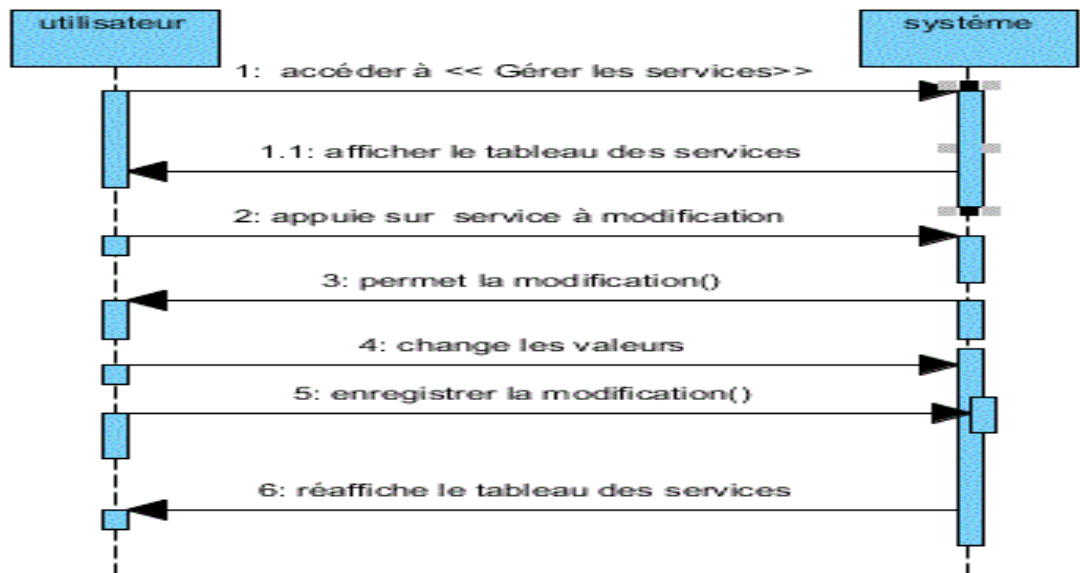


Figure 4.8: Diagramme de séquence de la modification du service.

7.3.4.Diagramme d'activité :

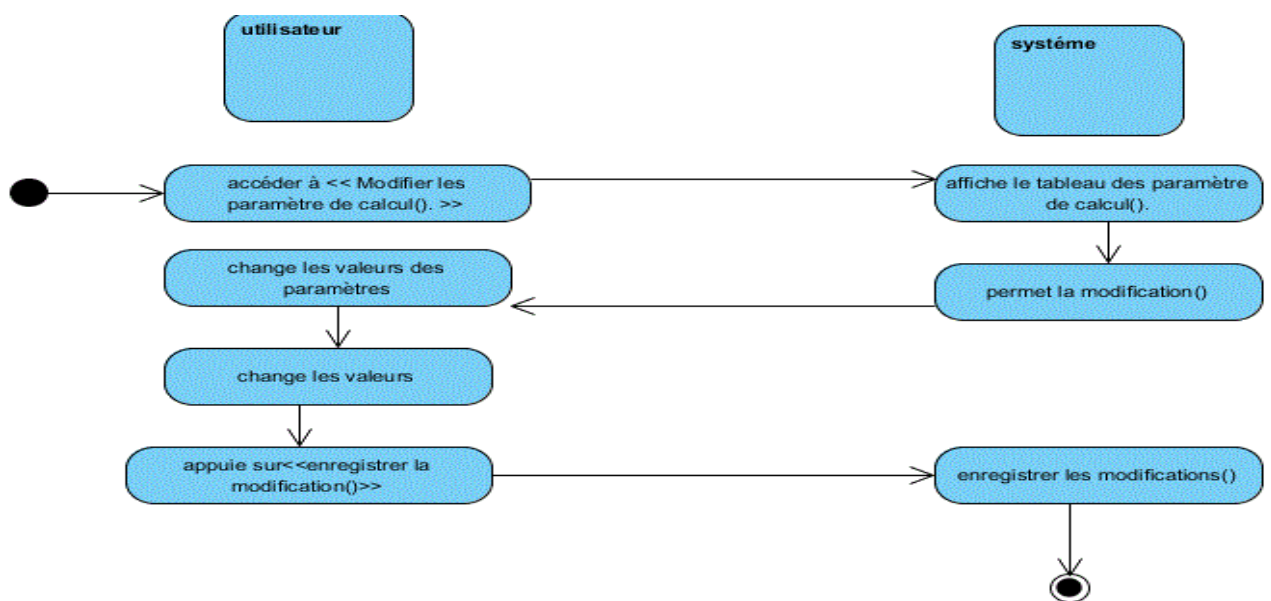


Figure4.9 : Diagramme d'activité de la modification du service.

7.4. 4^{eme} scenario/la modification des paramètres de calcul.(poids).

Titre : Modifier les paramètres de calcul.

But : changer les paramètres de calcul.

Résumer : changer les valeurs du poids, valeur éliminatoire, valeur optimale.

Acteur : Utilisateur.

7.4.1.Scenario normal :

1. L'utilisateur accède à <Modifier les paramètres de calcul (les poids)>
2. Le système affiche le tableau des paramètres de calcul (les poids).
3. L'utilisateur change les valeurs des paramètres.
4. L'Utilisateur enregistre les modifications.
5. Retour à2.

7.4.2.Scenario alternatif :

- A. L'utilisateur annule la modification du service.
- 6.Quitté la modification.
7. Retour à2.

7.4.3.Diagramme de séquence :

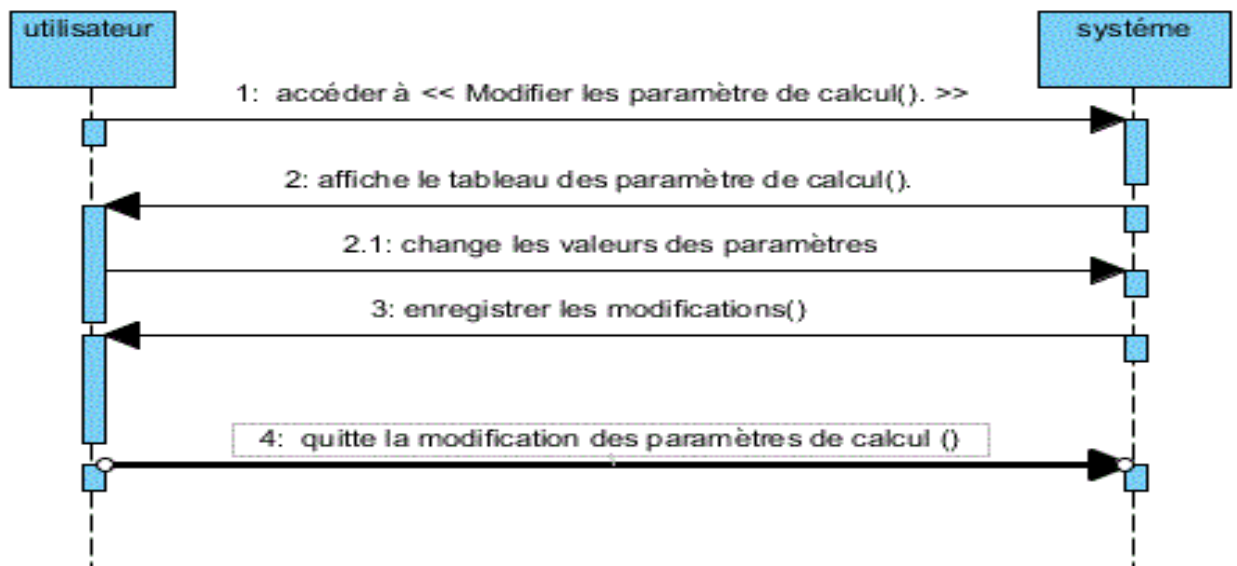


Figure 4.10: Diagramme de séquence la modification des paramètres de calcul.(poids).

7.4.4. Diagramme d'activité :

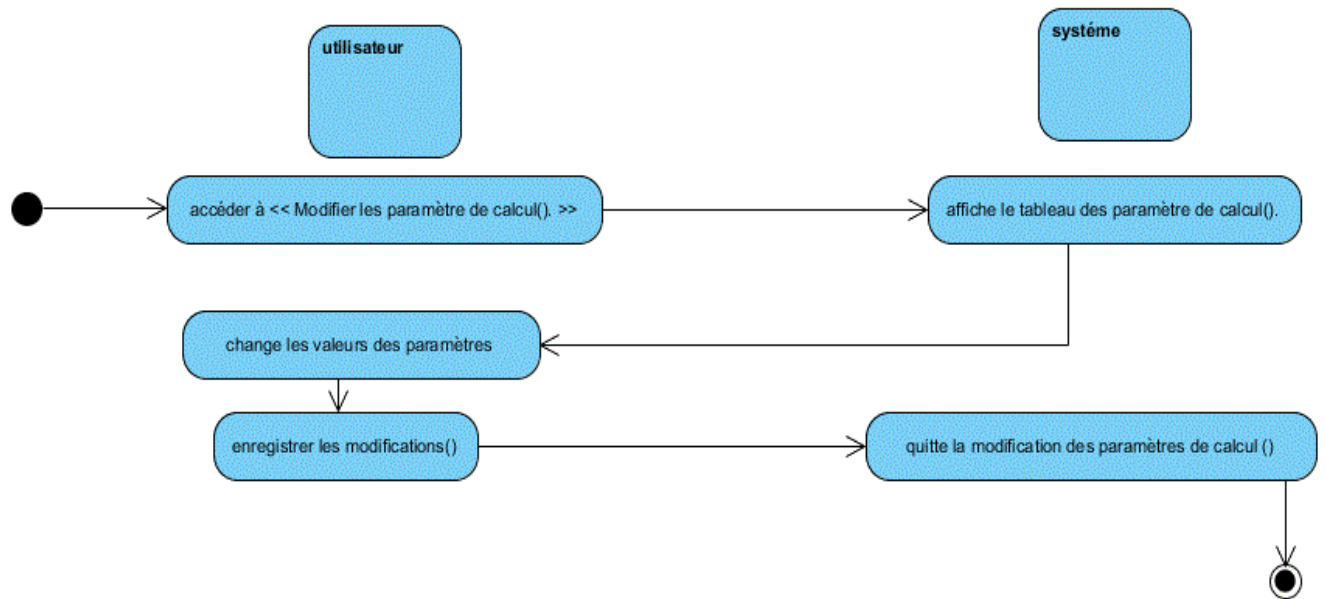


Figure 4.11 : Diagramme d'activité la modification des paramètres de calcul. (Poids).

7.5. 5^{ème} scénario/lancer le calcul.

Titre : lancer le calcul.

But : calculer la qualité de chaque service.

Résumer : calculer la qualité de chaque service afin de sélectionner le meilleur service.

Acteur : Utilisateur.

7.5.1. Scénario normal :

1. L'utilisateur accède à <lancer le calcul>
2. Le système vérifie l'existence des services déjà enregistré.
3. Le système affiche le tableau de liste des services.
4. Le système permet le lancement de la sélection.
5. L'Utilisateur appui sur<lancer la sélection>.
6. Le système démarre le calcul selon un algorithme TOPSIS.
7. Le système démarre le calcul selon une fonction de dominance.
8. Le système affiche la liste des services sélectionné classifié.

7.5.2.Scenario alternatif :

A1.la liste des services sur la base donnée est vide.

3.le système affiche le message d’erreur.

4. Quitte<lancer le calcul>.

7.5.3.Diagramme de sequence:

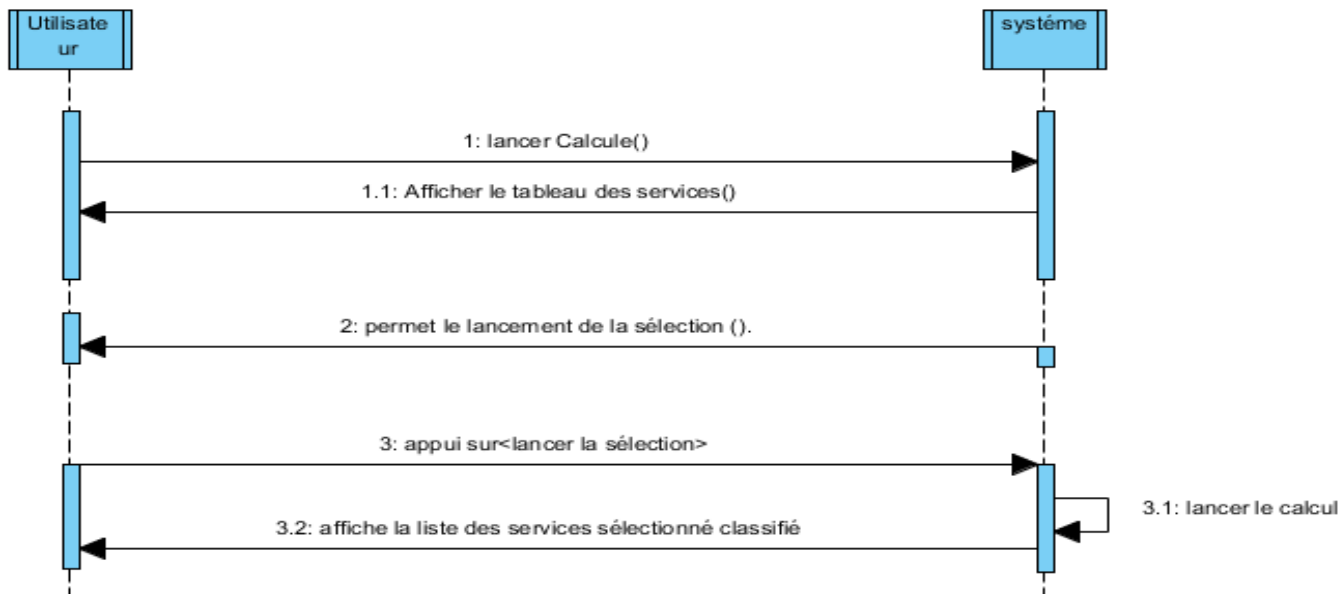


Figure 4.12 : Diagramme de séquence de lancement du calcul.

7.5.4.Diagramme d’activité :

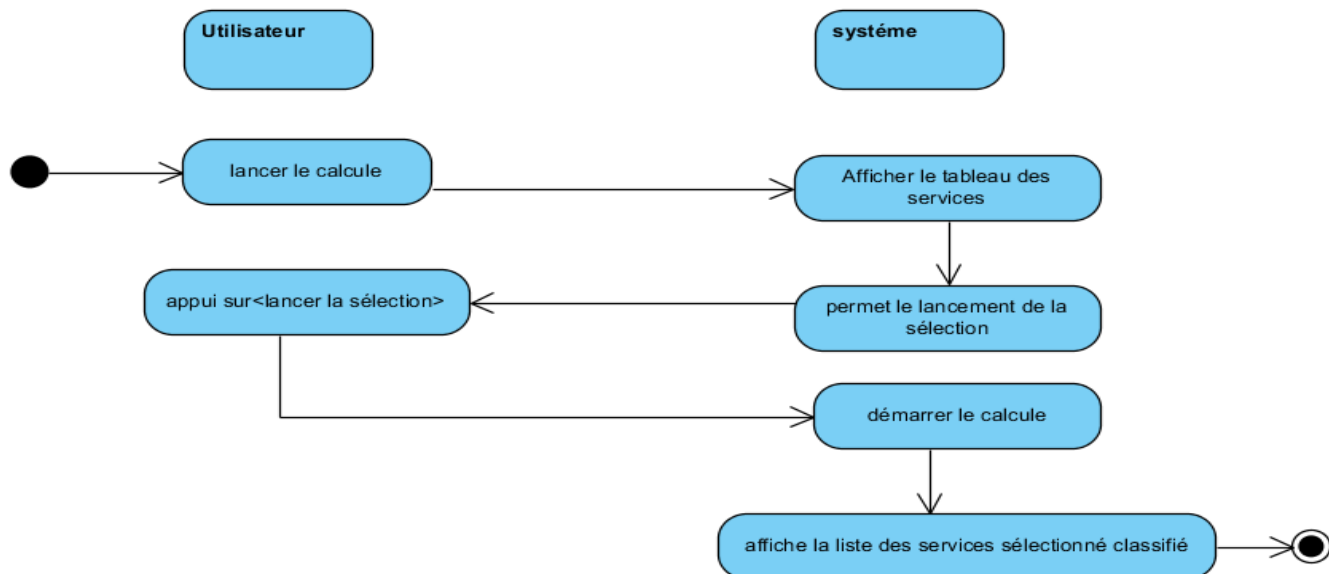


Figure 4.13 : Diagramme d’activité de lancement du calcul.

7.6. Diagramme de classe

La figure suivante montre le diagramme de classe de notre application

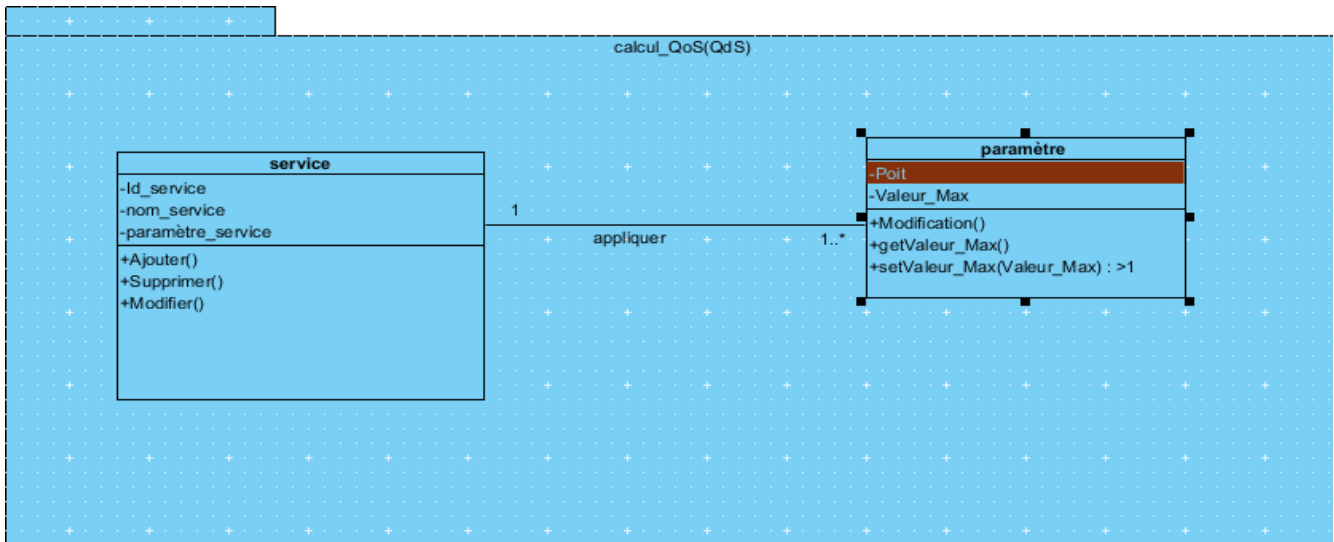


Figure 4.14 : Diagramme de classe.

8. Conclusion:

Dans ce chapitre nous avons présenté notre modèle proposé pour la sélection des services web en se basant sur des critères de qualité de service (QoS). On a intégré un algorithme de décision qu'il va faire le classement des services par ordre selon des critères non fonctionnels. Nous avons exprimé également le fonctionnement de notre système en se basant principalement sur les différents diagrammes UML et sur les scénarios de description pour chaque phase.

Cette étude conceptuelle présente l'architecture générale de notre travail. Dans le prochain chapitre nous allons présenter les techniques utilisées pour implémenter l'application. Ainsi que, une étude de cas sur un exemple concrète.

Chapitre 5



***Implémentation et
Étude de cas***

1. Introduction :

Après avoir exposé notre approche de modélisation de la sélection des services Web en prenant en compte les critères de qualité de service avec l'utilisation d'un algorithme qui aide à la décision multicritère TOPSIS, nous allons à présent décrire la phase de réalisation relative à cette dernière.

Nous allons montrer dans ce chapitre comment nous avons réalisé et implémenté notre système. Nous aborderons l'aspect pratique de notre application, il s'agit ici d'expliquer l'environnement matériel sur lequel notre système a été développé, les langages de programmation et les outils/technologies utilisés. Pour terminer, nous allons présenter les interfaces graphiques en décrivant les différentes fonctionnalités de notre application et nous présenterons aussi un exemple¹ réel sous la forme d'une étude de cas.

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.

2.1. Environnement matériel et logiciel :

Pour réaliser notre système, nous avons un PC I3 doté de Windows 10 (64bits) qui est décrit avec la figure suivante :

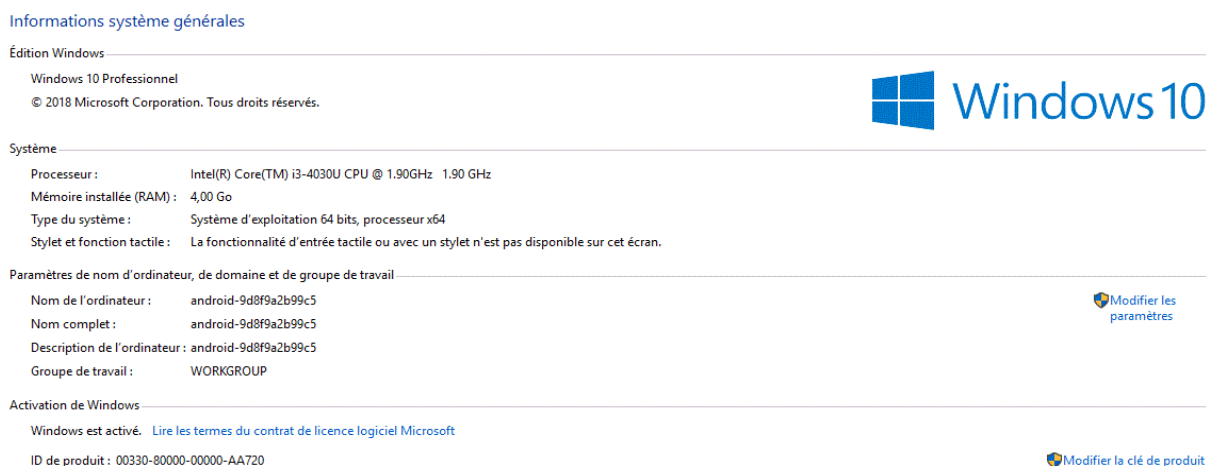


Figure 5.1 – Environnement logiciel utilisé.

¹ "<http://www.uoquelp.ca/qmahmoud/qws/search.html>"

2.2. Langage de programmation :

Pour l'élaboration du système conçu, Nous avons utilisé les langages de programmation suivants :

2.2.1. Le langage Java :

Le langage Java est un langage de programmation et une plate-forme informatique évolué et orienté objet qui est créé par James Gosling et Patrick Naughton, employés de Sun Microsystems, avec le soutien de Bill Joy (cofondateur de Sun Microsystems en 1982), présenté officiellement le 23 mai 1995 au SunWorld. La société Sun a été ensuite rachetée en 2009 par la société Oracle qui détient et maintient désormais Java. Aujourd'hui, Java rassemble derrière lui une large communauté d'acteurs informatiques majeurs tels que HP, IBM, Oracle, Borland [Java]. Il est rapide, sécurisé et fiable. En outre, beaucoup d'applications et de sites Web ne fonctionnent pas si Java n'est pas installé et leur nombre ne cesse de croître chaque jour. À cause de Sa simplicité, sa robustesse, sa portabilité ainsi que sa performance lui ont permis d'être le choix préféré pour le développement de notre application [34].

Les caractéristiques du langage java sont [35,36] :

- **Interprété** : Le fonctionnement de Java est assuré JVM (Java Virtual Machine) et JDK qui peuvent être installés dans les différents systèmes d'exploitation. Les instructions JVM sont traduites lors de leur exécution en instructions natives de la machine ; Compilateur Java traduit le code source Java en bytecode (code portable). Par la suite un interpréteur Java spécifique JVM traduit et exécute le bytecode. Il permet l'encapsulation et la génération dynamique d'autres technologies.
- **Portable** : Java fonctionne en mode interprété il est donc un langage portable. Il est indépendant de toute plate-forme : Windows, UNIX, DOS. Ce concept est à la base du slogan de Sun pour Java : WORA (Write Once, Run Anywhere : écrire une fois, exécuter partout). Il n'y a pas de compilation spécifique pour chaque plate-forme. Le code reste indépendant de la machine sur laquelle il s'exécute. Il est possible d'exécuter des programmes Java sur tous les environnements qui possèdent une Java Virtual Machine. Cette indépendance est assurée au niveau du code source grâce à Unicode et au niveau du code byte.
- **Orienté objet** : Comme la plupart des langages récents, Java est orienté objet. Chaque fichier source contient la définition d'une ou plusieurs classes qui sont utilisées les unes avec les autres pour former une application. Java n'est pas complètement objet car il définit des types primitifs (entier, caractère, flottant, booléen,...). Ainsi, Java est un langage orienté objet simple

ce qui réduit les risques d'incohérence et son programme n'est pas un ensemble de procédures qui s'appellent les unes les autres mais un ensemble d'objets.

- Simple : Le choix de ses auteurs a été d'abandonner des éléments mal compris ou mal exploités des autres langages tels que la notion de pointeurs (pour éviter les incidents en manipulant directement la mémoire), l'héritage multiple et la surcharge des opérateurs.
- Sécurisé : La sécurité fait partie intégrante du système d'exécution et du compilateur. Un programme Java planté ne menace pas le système d'exploitation. Java assure une certaine sécurité au système à travers des tests qui vérifient en permanence la conformité du pseudo-code à certaines règles ; Security Manager : protection des fichiers et accès au réseau.
- Multitâche : Il permet l'utilisation de threads qui sont des unités d'exécution isolées. La JVM utilise plusieurs threads.
- Riche bibliothèque : Java possède une riche bibliothèque de classes comprenant des fonctions diverses telles que les fonctions standards, le système de gestion de fichiers, les fonctions multimédia et plusieurs d'autres fonctionnalités.

2.2.2. Le Langage de Modélisation Unifié : (UML) Unified Modeling Language est un langage de modélisation graphique pour visualiser la conception d'un système. (diagramme de classe, diagramme de séquence, diagramme d'activité, diagramme de cas d'utilisation).

2.2.3. Le langage XML :

XML permet de définir des balises et de leur associer une interprétation. Dans HTML, on n'utilise les balises que pour décrire 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.

2.3. Outils et technologies :

2.3.1. Eclipse :

Eclipse est un projet, décliné et organisé en un ensemble de sous-projets de développements logiciels, de la fondation Eclipse visant à développer un environnement de

production de logiciels libre qui soit extensible, universel et polyvalent, en s'appuyant principalement sur Java.

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

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

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

Les logiciels commerciaux Lotus Notes 8, IBM Lotus Symphony ou WebSphere Studio Application Developer sont notamment basés sur Eclipse[34].

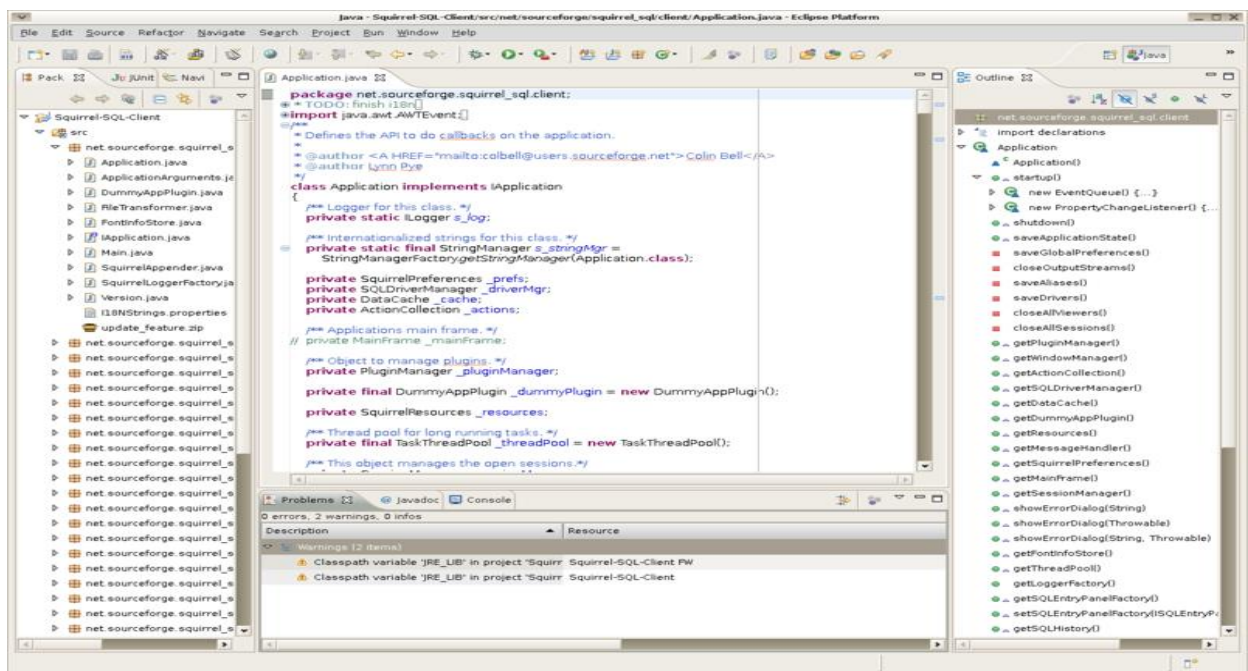


Figure 5.2 – Interface principale de Eclipse .

2.3.2. Gestion de base de données (phpMyAdmin)

phpMyAdmin (anciennement MySQL administrator) est un logiciel de gestion et d'administration de bases de données MySQL . Via une interface graphique intuitive, il permet, entre autres, de créer, modifier ou supprimer des tables, des comptes utilisateurs, et d'effectuer toutes les opérations inhérentes à la gestion d'une base de données. Pour ce faire, il doit être connecté à un serveur MySQL [32].

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



Figure 5.3 : Interface serveur phpMyAdmin.

2.3.3. Visual Paradigm (VP-UML) est un outil UML CASE supportant UML2.

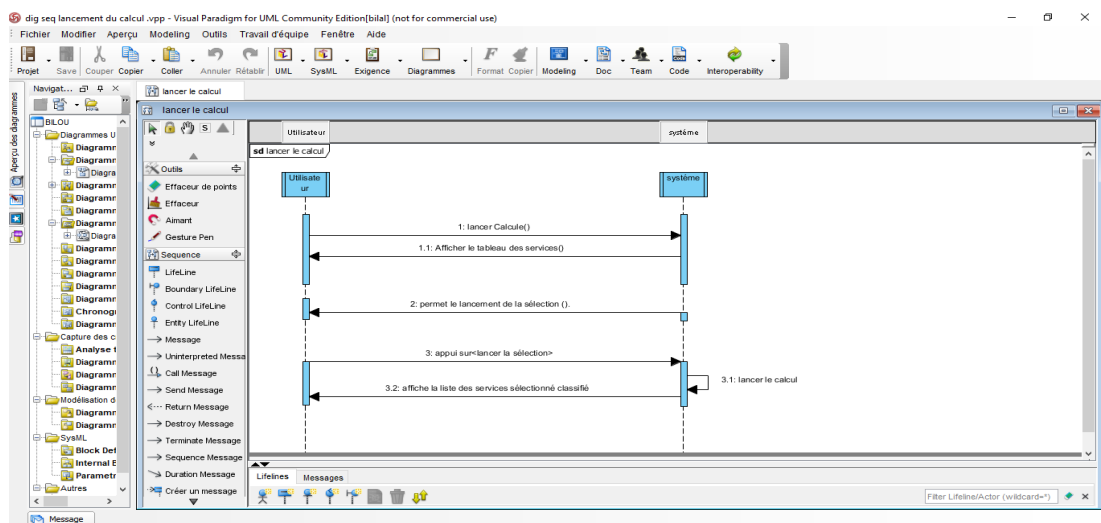


Figure 5.4 : Interface visual paradigm.

3. Fonctionnalités :

3.1. Visualisation graphique des performances :

L'une des spécificités de ce client de gestion de base de données réside dans sa possibilité de visualiser en temps réel, via des indicateurs graphiques, la charge du serveur MySQL, le pourcentage d'occupation en mémoire, les connexions courantes, le nombre de requêtes continues sur une base de données. Il permet aussi de voir l'occupation disque, ainsi que l'espace alloué aux tables et aux fichiers de logs [32].

Enfin, les tables sont disponibles sous forme de diagramme, permettant ainsi la modélisation des données [32].

3.2. Sauvegarde et restauration des données

Le logiciel permet, comme les autres clients de gestion de base de données, de créer facilement des sauvegardes de tables et de bases. Il permet aussi de restaurer rapidement des données, par simple sélection des tables via l'interface de gestion [32].

3.3. Transfert entre SGBD

Une fonction permet d'exporter des bases de Microsoft SQL Server vers MySQL [32].

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

- A. **Temps de réponse** (ResponseTime) : est une performance qui représente la vitesse avec laquelle un service Web répond à une requête.
- B. **Fiabilité** (Reliability) : est la capacité d'un service à remplir ses fonctions requises dans les conditions indiquées pour une période de temps déterminée.
- C. **Documentation** (Documentation) : L'une des principales propriétés des services Web est la documentation appropriée. La propriété QWS de documentation fournit une mesure où un service Web est auto-datable et repose sur l'examen des documents WSDL, y compris le nom du service, la description, le nom de l'opération, la description, le nom du message et les balises de description des messages.

D. **Sécurité** (Security) : Il existe plusieurs traitements permettant de sécuriser les communications entre services Web (Authentification, Autorisation, Confidentialité, Traçabilité, Cryptage de données, Non répudiation).

E. **Coût d'un service** (Cost) : est le coût à payer pour consommer le service, ce coût peut être fourni par le fournisseur du service [9].

4. Présentation des interfaces graphiques :

4.1. 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.

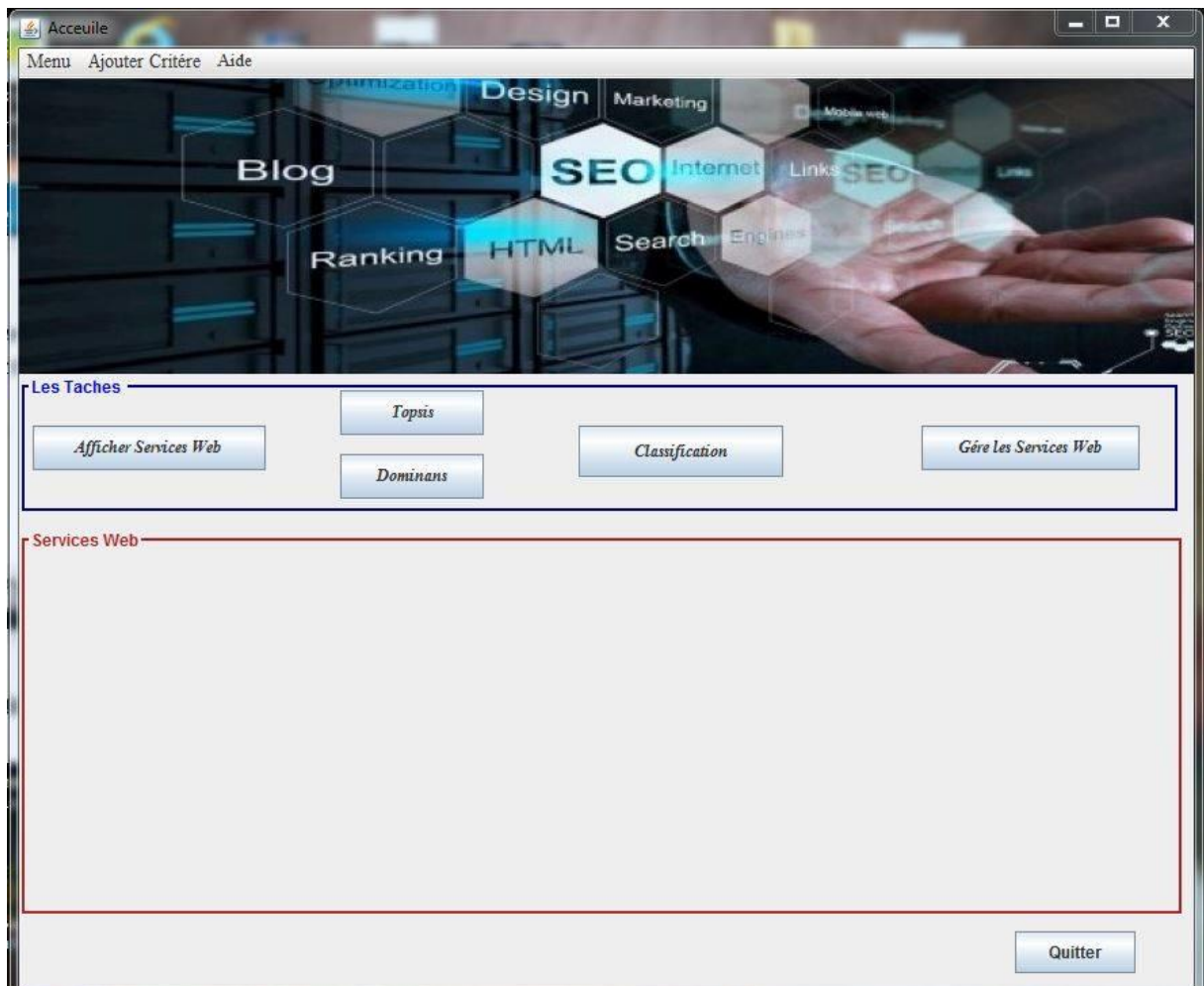


Figure 5.5 : Interface principale de l'application.

4.2. Les interfaces principales de notre prototype :

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

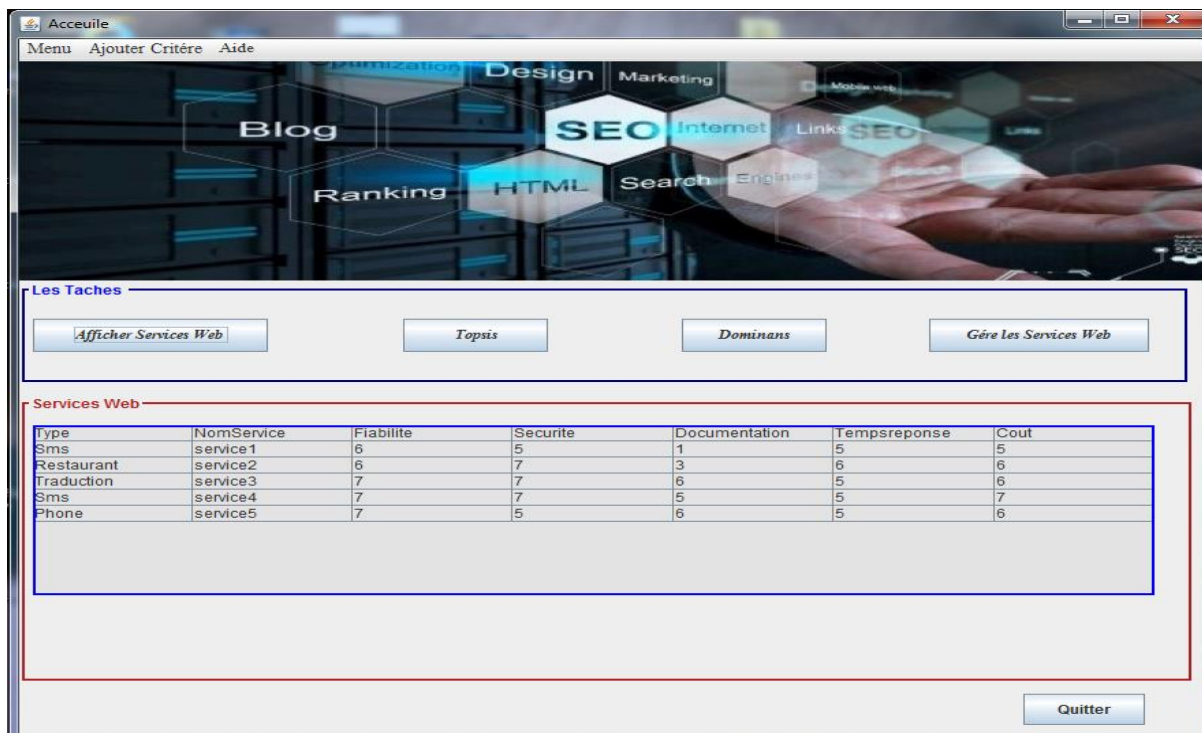


Figure 5.6 : Interface d'affichage des services .

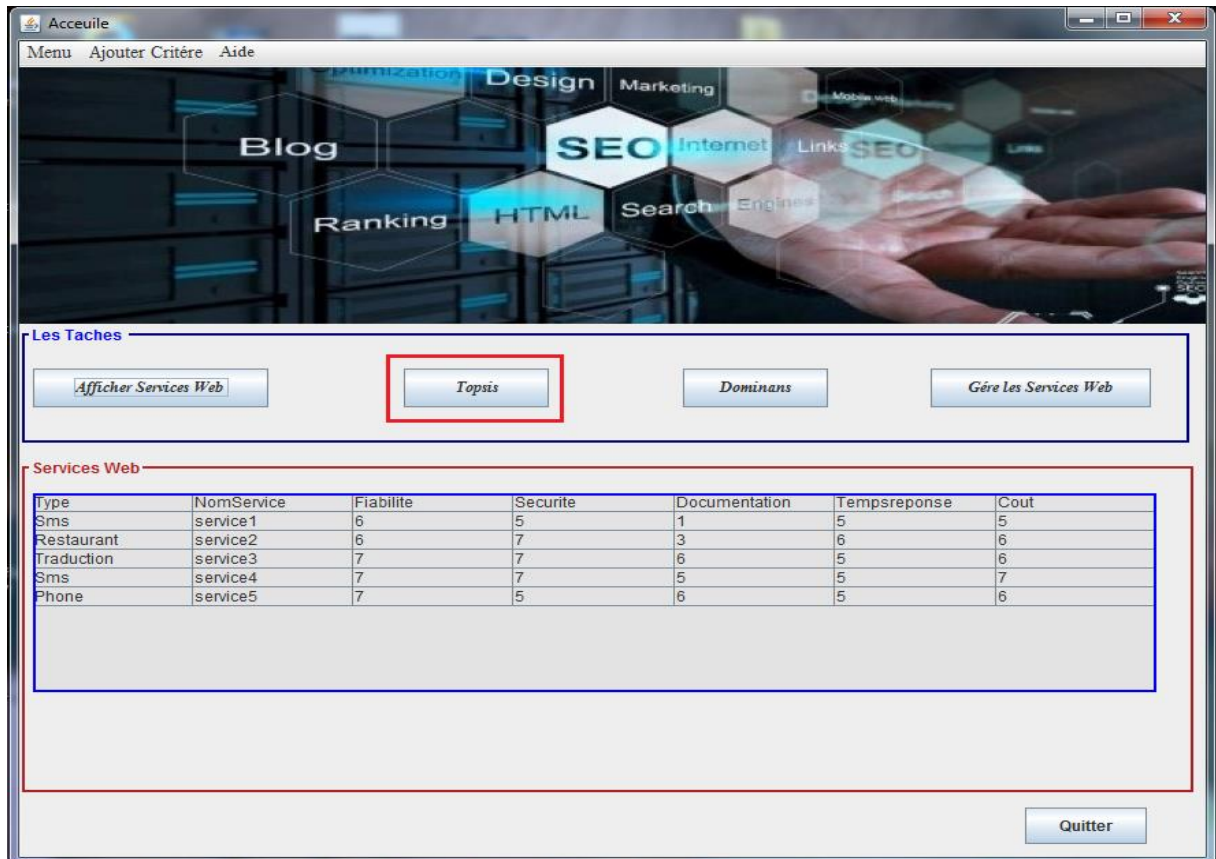


Figure 5.7 : Interface choix du bouton TOPSIS pour sélection les services.



Figure 5.8 : Résultat calculé par TOPSIS.

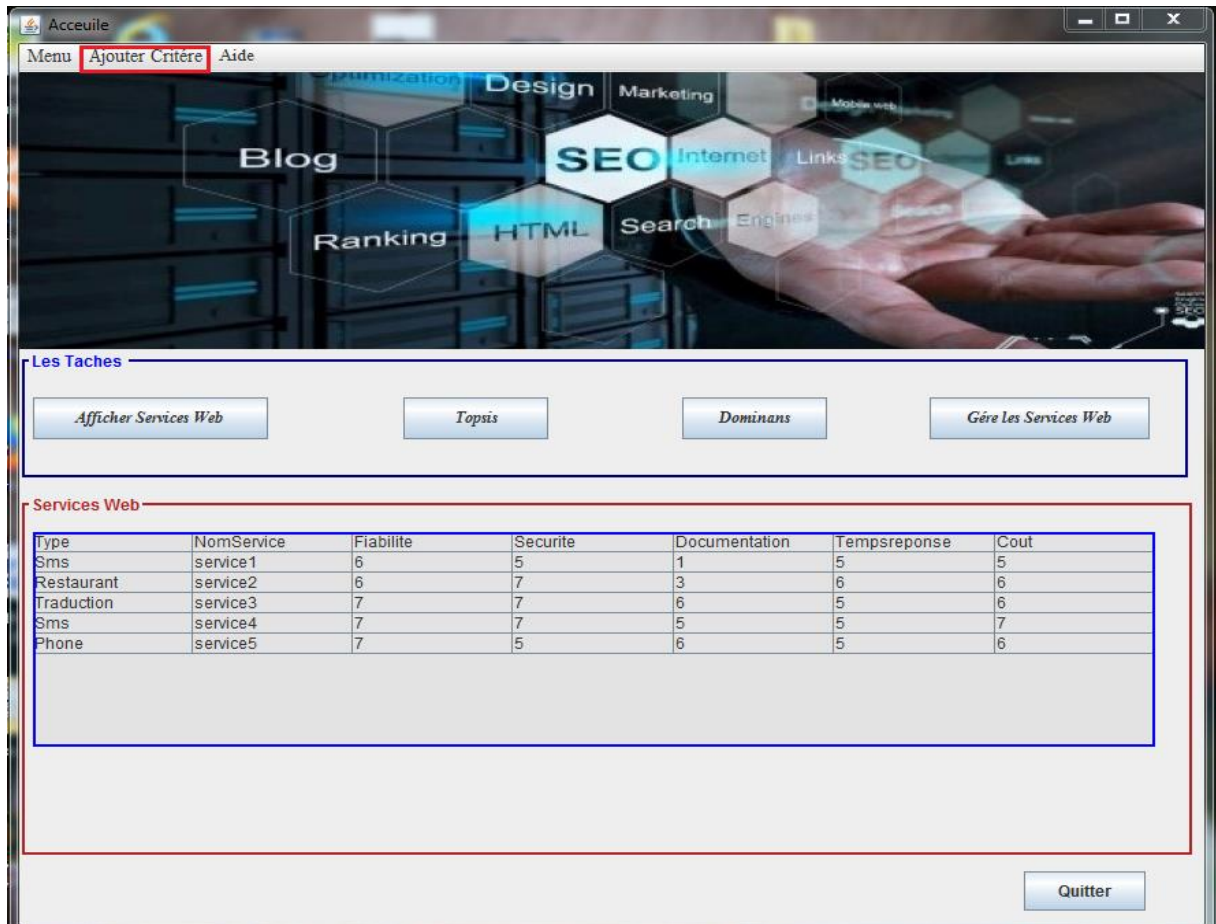


Figure 5.9 : Interface (1) modification des poids.



Figure 5.10 : Interface (2) modification des poids.



Figure 5.11 : Résultat calculé par TOPSIS (Modification des poids).

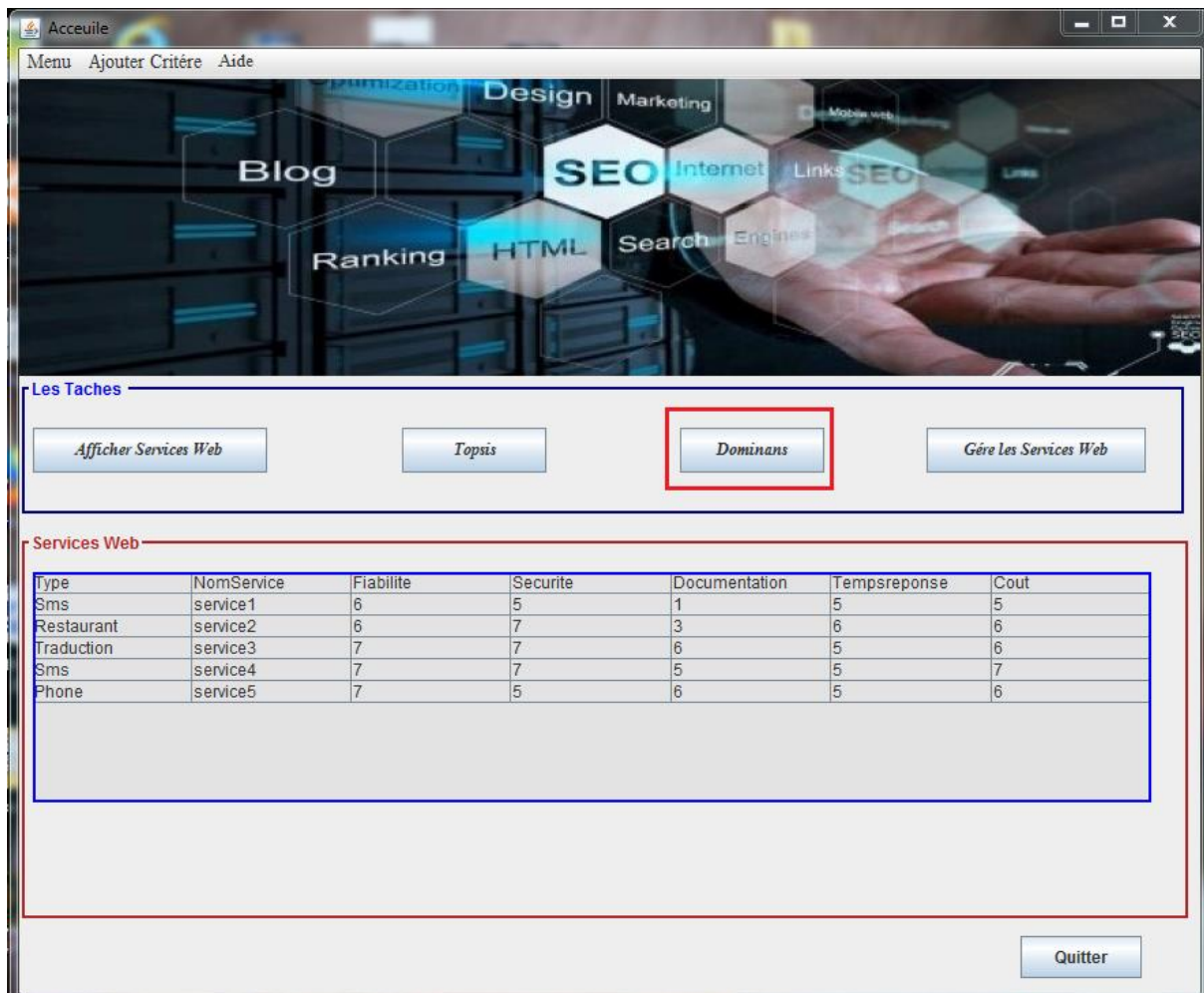


Figure 5.12: Interface choix du bouton Dominance pour sélection les services.

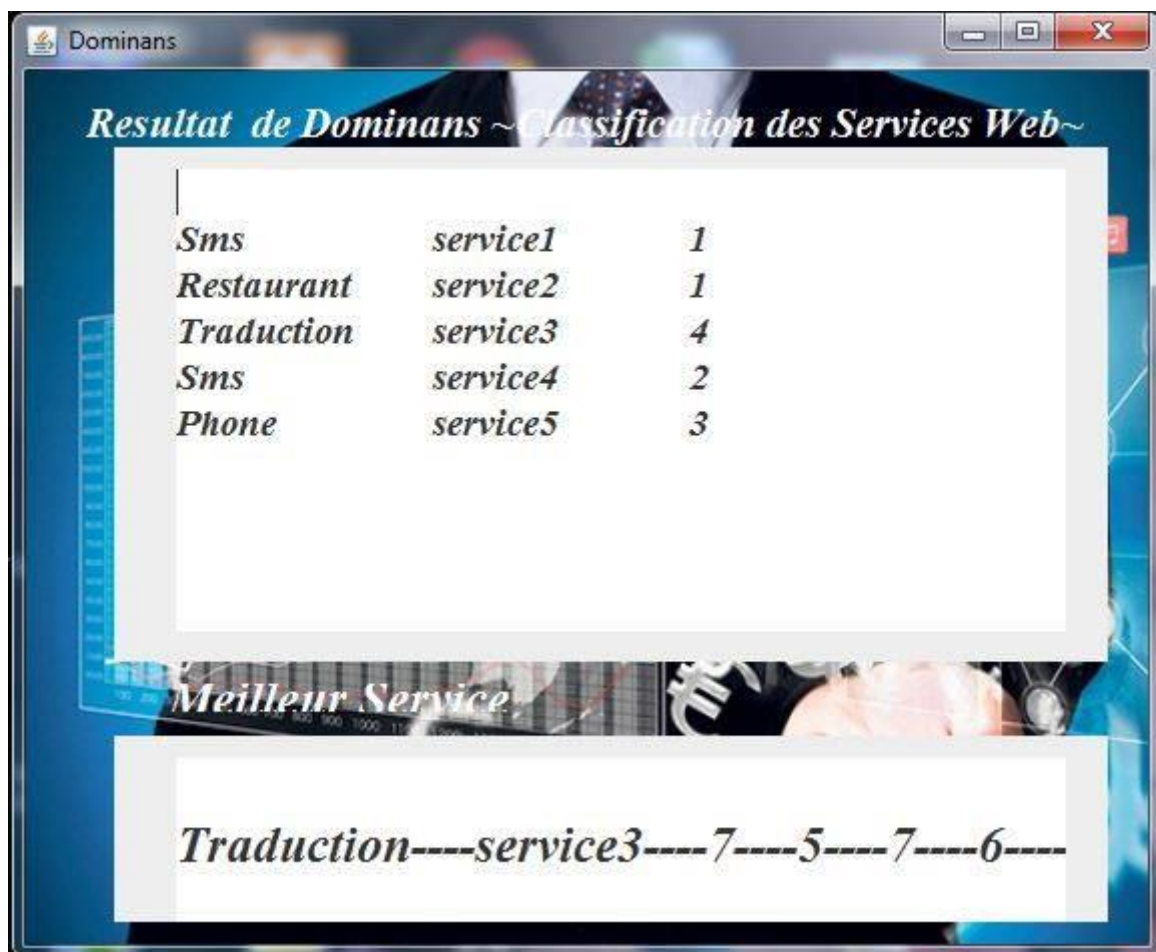


Figure 5.13: Résultat calculé par Dominance.

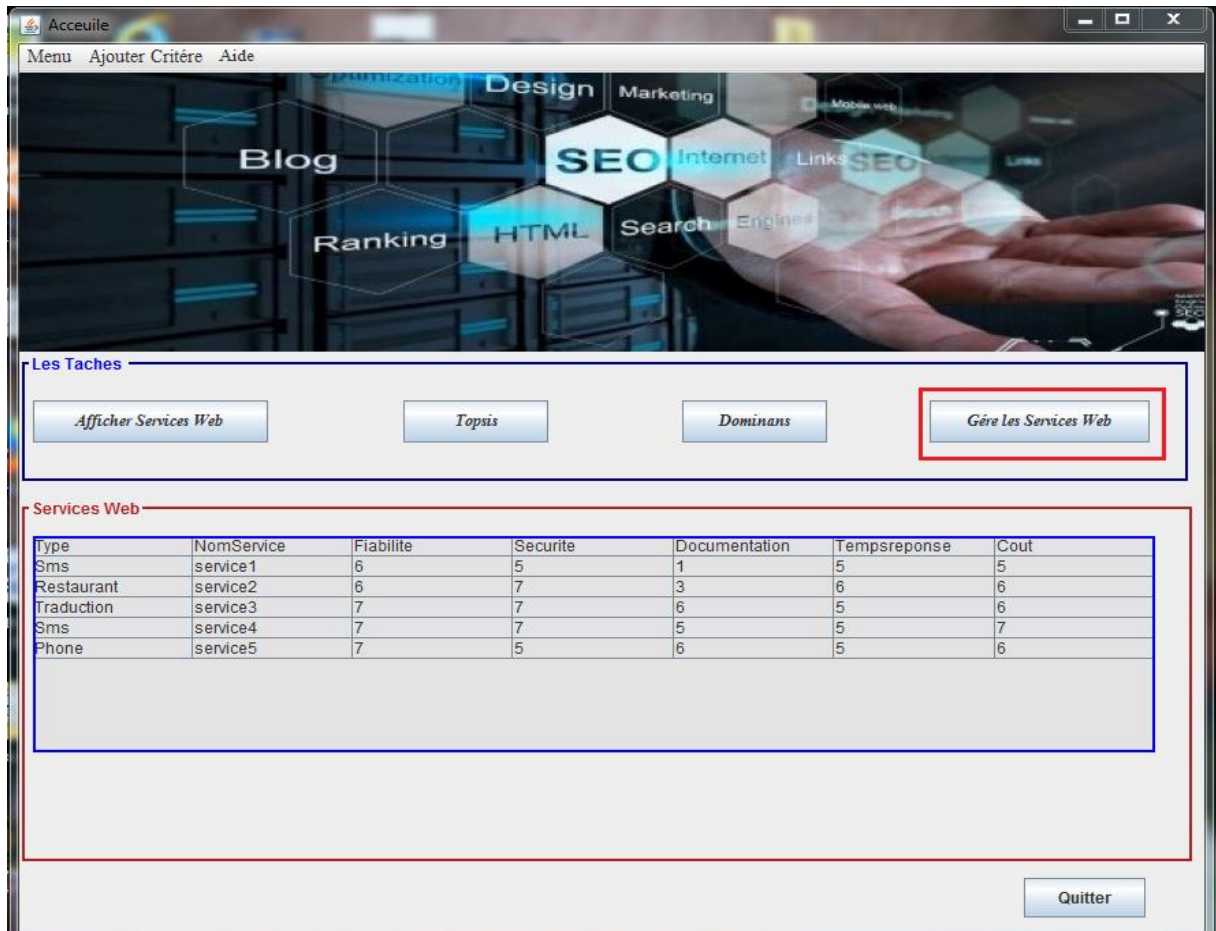


Figure 5.14 : Interface pour gère les services web.

- Dans le cas d'ajout d'un nouveau service Web à notre base de données par un fournisseur des services il faut d'abord authentifier l'identité de fournisseur.

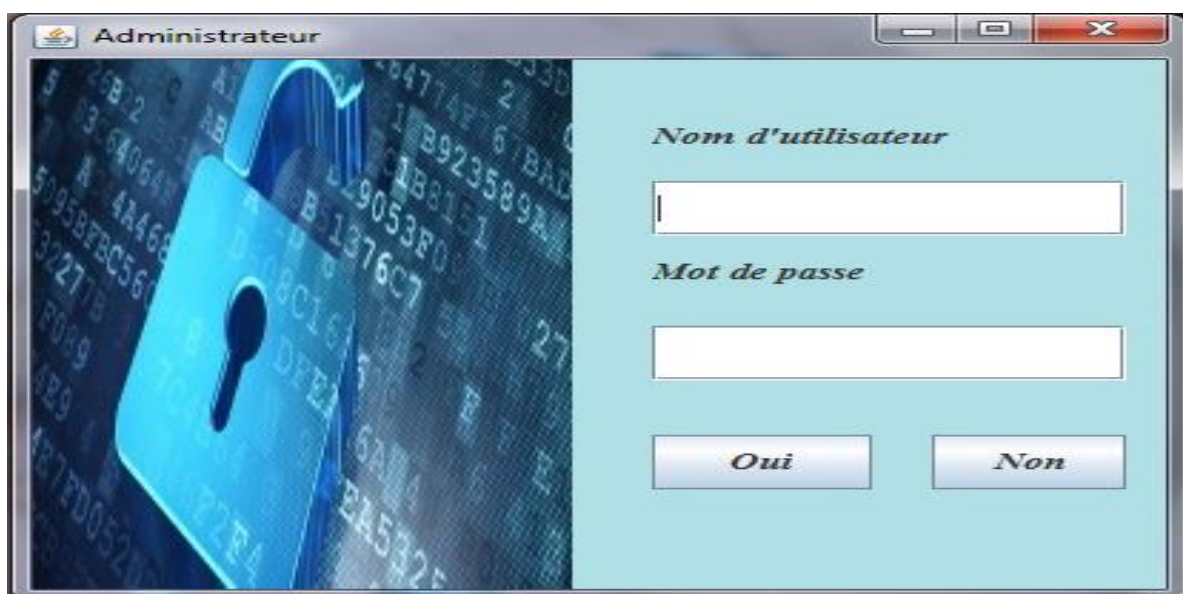


Figure 5.15 : Interface login pour gère les services.

Service WEB

Type
Phone

Nom
Service 6

Fiabilité
10

Sécurité
9

Documentation
8

Temps de reponse
4

Cout
7

Oui Non

Figure 5.16 : Interface pour l'ajout de nouveaux services à la base de données..

```
SELECT *  
FROM `web_service`  
LIMIT 0, 30
```

Profilage [En ligne] [Modifier] [Expliquer SQL] [Créer source PH

Afficher : 30 ligne(s) à partir de la ligne n° 0 en mode horizontal et répéter les en-têtes à chaque groupe de 100

Trier sur l'index: Aucune

+ Options

	ID	Type	NomService	Fiabilite	Securite	Documentation	TempsReponse	Cout
<input type="checkbox"/> Modifier <input type="checkbox"/> Éditer en place <input type="checkbox"/> Copier <input type="checkbox"/> Effacer	1	Sms	service1	6	5	1	5	5
<input type="checkbox"/> Modifier <input type="checkbox"/> Éditer en place <input type="checkbox"/> Copier <input type="checkbox"/> Effacer	3	Restaurant	service2	6	7	3	6	6
<input type="checkbox"/> Modifier <input type="checkbox"/> Éditer en place <input type="checkbox"/> Copier <input type="checkbox"/> Effacer	5	Traduction	service3	7	7	6	5	6
<input type="checkbox"/> Modifier <input type="checkbox"/> Éditer en place <input type="checkbox"/> Copier <input type="checkbox"/> Effacer	6	Sms	service4	7	7	5	5	7
<input type="checkbox"/> Modifier <input type="checkbox"/> Éditer en place <input type="checkbox"/> Copier <input type="checkbox"/> Effacer	7	Phone	service5	7	5	6	5	6
<input type="checkbox"/> Modifier <input type="checkbox"/> Éditer en place <input type="checkbox"/> Copier <input type="checkbox"/> Effacer	8	Phone	Service 6	10	9	8	4	7

Tout cocher / Tout décocher Pour la sélection : Modifier Effacer Exporter

Afficher : 30 ligne(s) à partir de la ligne n° 0 en mode horizontal et répéter les en-têtes à chaque groupe de 100

Figure 5.17: L'ajout d'un nouveau service à la base de données.

Modification un Service Web

Type

Nom

Oui Non

Type

Nom

Fiabilité

Sécurité *Modifier*

Documentation

Temps de Reponse

Cout

Sauvegarder Annuler

Figure 5.18 : modification des paramètres d'un service

Supprimer un Service Web

Type

Nom Service

Suppri... Annuler

Figure 5.19 : supprimer un service dans la base de données.

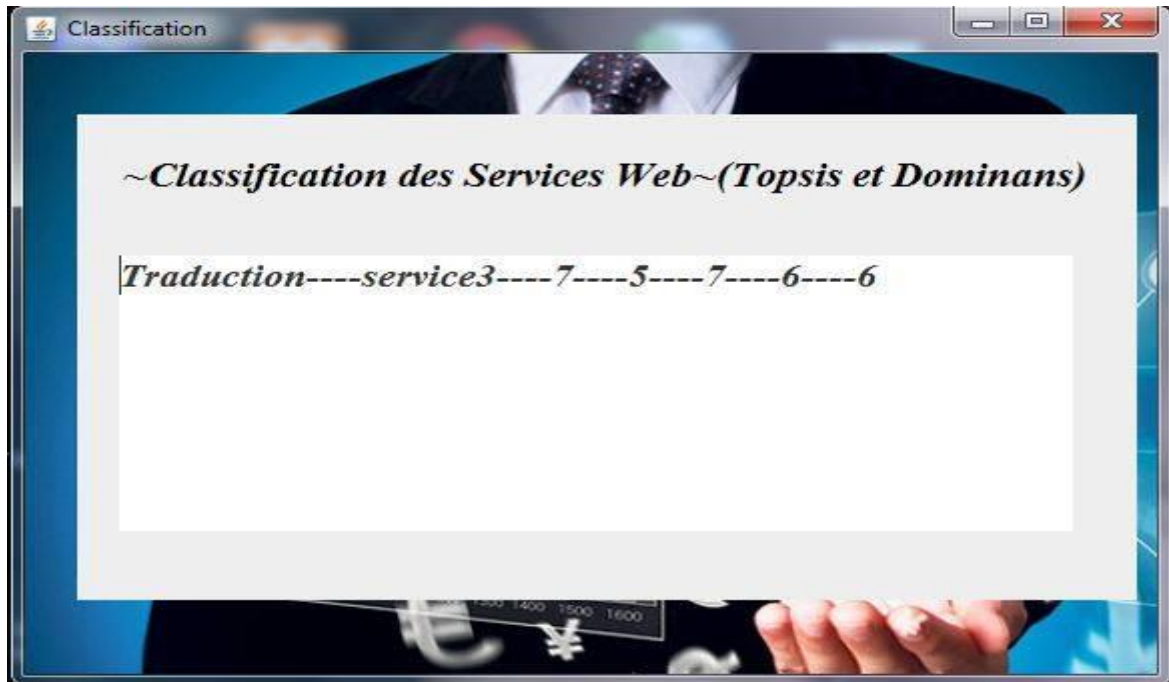


Figure 5.20: le meilleur résultat est classé à partir de Dominance et TOPSIS.

5. Conclusion :

Dans ce dernier chapitre nous avons proposé un nouveau modèle pour résoudre le problème de sélection de services Web à base de qualité de service (QoS) avec une intégration d'un algorithme qui s'appelle TOPSIS, qui il va faire un classement des services avec leurs qualités à travers des étapes bien définies et bien spécifier, en se basent sur l'évaluation de ses critères de qualité de service dans chaque étape avec l'utilisation des formules mathématiques bien choisi pour remédier se problème.

Cet exemple d'application nous a permis de bien illustrer le travail, de présenter les résultats de la validation de notre modèle, afin d'évaluer l'efficacité de ce dernier. D'après les résultats obtenus, nous avons confirmé l'efficacité de l'algorithme TOPSIS, dans le domaine de sélection de services Web, en effet les résultats obtenus sont acceptables et montrent leur supériorité par rapport à d'autres approches de l'état de l'art par exemple, ELECTRE, ELECTRE TRI et la méthode de somme pondérée (WSM).

Nous avons présenté quelques détails concernant la réalisation de notre application, en choisissant le langage de programmation java pour sa compatibilité avec les concepts orienté objet, et également pour ces avantages multiples de simplicité, portabilité et sécurité. Nous avons aussi présenté un exemple illustrant les différents services offerts par notre application.



Conclusion générale

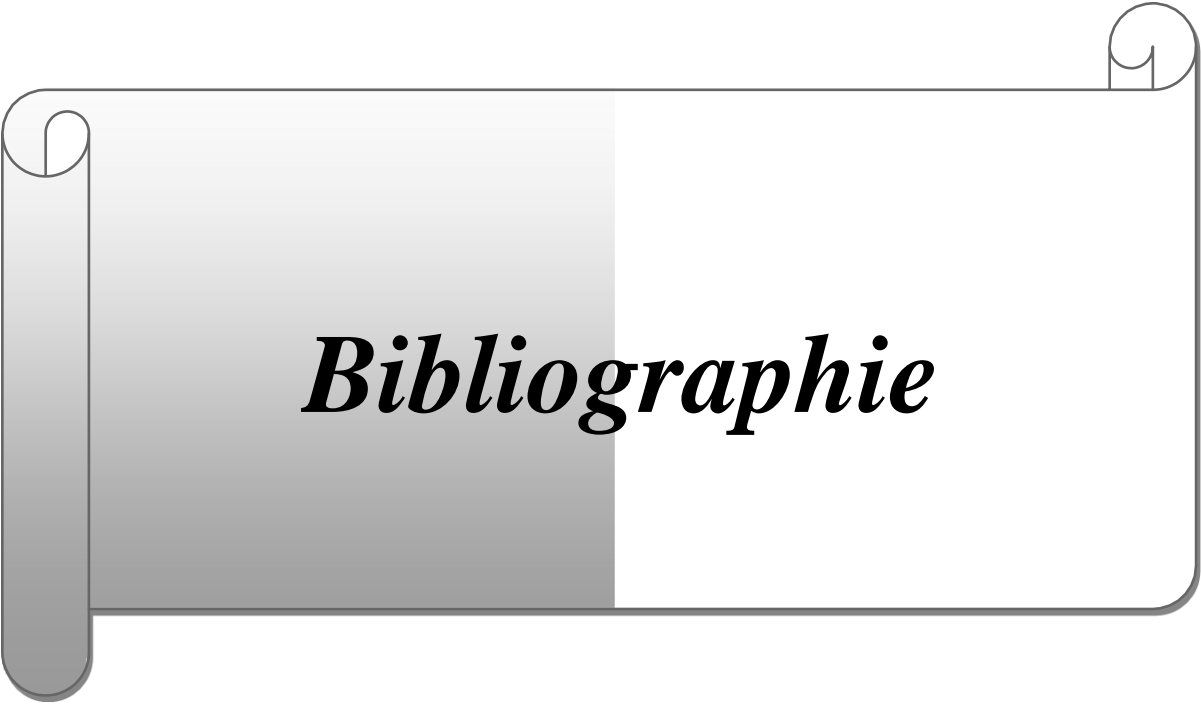
Conclusion Générale et Perspectives :

Le marché de commercialisation des services Web sur internet ne cesse d'augmenter, ce qui résulte en un nombre de plus en plus croissant de services offrant des fonctionnalités équivalentes. De ce fait, la sélection d'un service web approprié pour une tâche particulière est devenue un défi difficile pour l'utilisateur. Cependant, ces services diffèrent dans leurs qualités de services (QoS). Ces dernières deviennent alors cruciales pour l'utilisateur, car elles permettent de l'aider à choisir parmi les services Web qui sont fonctionnellement équivalents. Notre travail consiste à développer une application basée sur une méthode d'aides à la décision multicritère pour la sélection des meilleurs services Web en fonction de QoS.

Dans la littérature, divergents efforts ont été dépensés pour résoudre ce problème en se basant sur les propriétés non fonctionnelles des services Web. Dans ce papier, nous proposons un modèle pour la sélection des services Web en se basant sur des critères non fonctionnelles des QoS. Ainsi pour classer les services similaires, nous appuyons sur un algorithme TOPSIS qui classe les services web en fonction des exigences non-fonctionnelles. Afin de montrer la faisabilité de l'architecture proposée, nous avons développé un prototype de sélection de services web qui est une démonstration pour des services réels.

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

- La prise en considération des aspects et des critères relatifs au cloud : définir la sélection dans le contexte du cloud computing où d'autres critères de sélection de services sont à prendre en considération comme la distance entre services dans les data centers dans les clouds.
- Intégrer l'incertitude dans la QoS : nous constatons que les valeurs des paramètres QoS ne sont pas déterministes, mais elles évoluent et changent en fonction de l'environnement du Service web. De ce fait, il serait judicieux de développer des algorithmes qui gèrent la QoS incertaine.



Bibliographie

« Bibliographie »

- [1] **Samir Youcef**, "*Multicriteria Evaluation-Based Conceptual Framework for Composite Web Service Selection, Services Computing (SCC)*", IEEE International Conference, 2015.
- [2] **Faycal Bachtarzi**, "*Une approche de composition des services web basée transformation de graphes*", Une thèse de doctorat, Université Abdekhamid Mehri Constantine 2, 2014.
- [3] **Radhia Khellaf**, "*Vérification de la des services compatibilité web pour une composition*", Thèse de Magister, Université Abdekhamid Mehri Constantine 2, 2014.
- [4] **Hakim Boudjelaba**, "*Sélection des web services sémantiques*", Mémoire de Magister, Université de Bouira, 2012.
- [5] **Benamar Abdeladim**, et **Ait Hamouda Mounir**, "*Sélection de services web par l'optimisation d'essaim particulière hybride*", MASTER, Université Abou Bekr Belkaid Tlemcen, 2013.
- [6] **DEHANE Aicha Djihad**, et **Melle KEBIR Zohra**, "*Evaluation des techniques de codage d'ontologies sur les performances de la composition de services web*", Thèse Master, Université Abou Bekr Belkaid Tlemcen, 2012.
- [7] **Sofiane Chema**, "*Une approche de composition de services web à l'aide des réseaux de pétri orientés objet*", Thèse de Doctorat, Université Abdelhamid Mehri Constantine 2, 2014.
- [8] M. Gharzouli, "Composition des Web Services Sémantiques dans les systèmes Peer-toPeer (in french)", PhD thesis, Department of Computer Science, University of Constantine 2, Sep 2011
- [9] T. Melliti, "Interopérabilité des services Web complexes, Application aux systèmes multi-agents(in french)", PhD thesis, Department of Computer Science, University of Paris IX Dauphine, 2004.
- [10] D. C. Fallside and P. Walmsley, "XML Schema Part0: Primer Second Edition", W3C Recommendation, 28 Oct 2004, [Online]. Available: <http://www.w3c.org/TR/xmlschema-0/>.
- [11] M. Kay, "XSL Transformations (XSLT)", version 2.0, W3C Recommendation, 23 Jan2007, [Online]. Available: <http://www.w3.org/TR/xslt20/>.
- [12] J. Robie, D. Chamberlin, M. Dyck and J. Snelson, "XML Path Language (XPath)", version 3.0, W3C Recommendation, 08 Apr 2014, [Online]. Available: <http://www.w3.org/TR/xpath-30/>.
- [13] E. Cerami, "Web Services Essentials", number. 0-596-00224-6, O'Reilly, Feb 2002.
- [14] E. Christensen, F. Curbera, G. Meredith and S. Weerawarana, "Web services description language (wsdl) 1.1", Mar 2001, [Online]. Available: <http://www.w3.org/TR/wsdl>

« Bibliographie »

- [15] F. Curbera, M. Duftler, R. Khalaf, W. Nagy, N. Mukhi and S. Weerawarana, "Unraveling the web services web an introduction to soap, wsdl, and uddi", IEEE INTERNET COMPUTING, pp. 86-93, Mar./Apr.2002.
- [16] **Site web :URL :** "http://fr.wikipedia.org/wiki/Service_Web_Avantages", Consulté : 14/5/2018.
- [17] **David Chappell, et Tyler JEWELL, "Java Web Service"**, Édition : O'Reilly, Mars 2002
- [18] **Site web :** "<https://www.w3.org/TR/xmlschema-1>", Consulté : 14/02/2017
- [19] M. Driss, "Approche multi-perspective centrée exigences de composition de services Web (in french)", PhD thesis, Department of Computer Science, University of Rennes1, Dec 2011.
- [20] **Zainab Aljazzaf, "Bootstrapping quality of Web Services"**, à Journal of King Saud University –Computer and Information Sciences, 2015.
- [21] **João Negrão Antunes, "Fuzzy Logic Based Quality of Service Models"**, à IJCCI, 2012.
- [22] **Kuyoro Shade, Awodele, AkindeRonke, et Okolie Samuel, "Quality of Service (QoS) Issues in Web Services"**, in IJCSNS International Journal of Computer Science and Network Security, January 2012.
- [23] **Daniel A. Menascé, "QoS Issues in Web Services"**, in IEEE Internet Computing, Volume : 6, Issue : 6, Nov/Dec 2002.
- [24] **Zibin Zheng, et Michael R. Lyu, "QoS Evaluation of Web Services"**, in QoS Management of Web Services, 2013.
- [25] **A. Mani, et A. Nagarajan, "Understanding quality of service for Web services"**, IBM, 2002.
- [26] **Y Liu, AH Ngu, et LZ Zeng, "QoS Computation and Policing in Dynamic Web Service Selection"**, Proceedings of the 13th international World, 2004.
- [27] **khaled houchat,lallam djallal, "Une approche la sélection de services web "**, Thèse de Mastre Université Abdelhamid Mehri Constantine 2, 2013.
- [28] **Radhouane MIMOUNE, "Une approche la sélection de services web "**, Thème de Mastre Université Mohamed khider biskra , 2017.
- [29] **Yamamoto et Saeki, "La méthode TOPSIS"**, 2008.

« Bibliographie »

- [30] Siteweb pour MCDM methods: "http://shodhganga.inflibnet.ac.in/bitstream/10603/2896/12/12_chapter%205.pdf", Consulté : 22/5/2018.
- [31] **Kuyoro Shade, Awodele, AkindeRonke, et Okolie Samuel**, "*Quality of Service (QoS) Issues in Web Services*", in IJCSNS International Journal of Computer Science and Network Security, January 2012.
- [32]: "https://fr.wikipedia.org/wiki/MySQL_Workbench" + "<https://fr.wikipedia.org/wiki/phpMyAdmin>" + "https://en.wikipedia.org/wiki/Visual_Paradigm", Consulté le 16/05/2018.
- [33] "https://en.wikipedia.org/wiki/Visual_Paradigm", Consulté le 16/05/2018.
- [34] URL : "<https://www.java.com>" + "[https://fr.wikipedia.org/wiki/Eclipse_\(projet\)](https://fr.wikipedia.org/wiki/Eclipse_(projet)) ", Consulté le 18/05/2018.
- [35] Java caractéristique, URL : "<http://www.latrach.net>", Mai 2017.
- [36] **Mohamed N. Lokbani**, "*Caractéristiques de base du langage java*".
- [37] Kritikos et Plexousakis, 2009.
- [38] Zeng et al. 2003, 2004
- [39] Nemhauser et Wolsey, 1988.
- [40] Ardagna et al. 2005, 2007
- [41] Lei Xu et al., 2011
- [42] Yi Xia et al., 2011
- [42] Zhai et al. 2009
- [43] Maros, 2003
- [44] Khan, 1998
- [45] Akbar et al. 2001
- [46] Akbar et al. 2006
- [47] Yu et al., 2007
- [48] Kennedy & Eberhart, 1995.
- [49] Clerc, 2003
- [50] Von et Castro, 2001
- [51] Alrifai et al, 2008
- [52] Benouaret et al. 2011

« *Bibliographie* »

[53] Alrifai et al. 2012

[54] Bouguettaya, 2010

[55] Alrifai, 2010

[56] Coello et al., 2002

[57] Alrifai et al., 2010