Table des matières

Liste des figures

Liste des tableaux

Liste des Acronyms

A

API: Application Programming Interface

B

B2B : Business To Business

C

CORBA: Common Gateway Interface.

E

EAI : Entreprise Application Intégration.

H

HTML: Hyper Text Markup Language.

HTTP: Hyper Text Transport Protocol.

J

J2EE : Java 2 Standard Edition.

J2SE : Java 2 Entreprise Edition.

JVM : Java virtuel machine.

JRE : Java Runtime Environnement.

JDK : Java Devloppement Kit.

R

RCP: Remote Procedure Call.

REST: Representation State Transfers.

RPC: Remote Procedure Call.

S

SOAP: Service Oriented Architecture Protocol.

SMTP : Simple Object Adapter Protocol

T

TCP/IP: Transmission Control Protocol / Internet Protocol.

U

URL: Uniform locator.

URI: Uniform Resource Identifier.

UDDI: Universal Description Discovery and Integration.

W

WWW: World Wide Web.

WSDL: Web Service Description language.

W3C: World Wide Web Consortium.

X

XML : extensible Markup Language.

Introduction Générale

Introduction générale

I. Contexte

Les dernières décennies, la communication et l'interopérabilité entre des applications hétérogènes au sein des entreprises au niveau interne B2B (Business To Business) et au niveau interne EAI (Entreprise Application Intégration) sont un vieux défit dans le domaine informatique.

Les chercheurs ont tenté dans le passé de résoudre ces problématiques par diverses technologies telles que les scripts pilotés par batchs, CORBA, middlewares orienté messages, etc. Mais sans rencontrer de réel succès. De là les systèmes d'informations distribués font face à une situation intenable vis-à-vis des exigences métiers.

Les architectures orientées services (SOA) sont nées pour offrir une formidable opportunité pour résoudre ces problématiques basées sur les processus métiers et en particulier sur La technologie des services web.

L'architecture SOA (Service Oriented Architecture) est une architecture structurelle et organisationnelle qui se met en place par une vue dédiée à la convergence métier dont elle facilite la gestion de son processus , améliore le retour en investissement et la productivité de système d'information. Elle vise à combiner les services dans les processus commerciaux significatifs de plus haut niveau dans le cadres d'entreprise. Elle se met en place aussi par une vue informatique dont elle facilite la maintenance, réduit la complexité des solutions et garantit une intégration standardisée.

Elle permet, en plus, d'assurer l'infrastructure informatique qui permet d'intégrer les applications à partir de services écrits dans différentes langues et s'exécutant sur différentes plates-formes.

Les services web ont acquit un rôle considérable dans les applications e-business ainsi que les systèmes e-commerce, de plus ils ont concrétise le lien entre deux acteur : l'utilisateur et le fournisseur selon un contrat proposé par les pourvoyeurs où les

consommateurs définissent leur besoins en masquant l'hétérogénéité du système sous-jacents ainsi ils ont certifié aux utilisateurs de (ré) utiliser les services avec un minimum d'efforts.

Les services Web sont basés sur des protocoles et des standards ouverts tels que SOAP (Simple Object Access Protocol) afin d'utiliser des applications invoquées à distance par Internet., WSDL (Web Service Description Langage) qui décrit les services comme un ensemble d'opérations et de messages abstraits reliés à des protocoles et des serveurs réseaux, UDDI (Universal Description, Discovery and Integration) dans le but d'automatiser les communications entre prestataires, etc.

Ces standards sont développés par des organisations de normalisation telles que W3C et OASIS en utilisant le méta-langage XML et ils sont maniées à façade d'une interface de programmation d'application (API) et non pas à l'aide une interface utilisateur graphique (GUI) tels que les boutons.

II. Problématique

La communication entre les services Web via des technologies basées sur des normes qui donnent aux utilisateurs la possibilité d'accéder à différents services Web indépendants de leur matériel, système d'exploitation ou programmation même environnement. Cela prend en charge les organisations ayant une technologie pour créer des services qui peuvent être facilement découverts et consommés par des utilisateurs. L'un des défis de la recherche critique collaboratif du développement des logiciels est la composition du service Web (WSC) qui consiste à créer un service composite en combinant différents services Web existant pour fournir un nouveau service à valeur ajoutée.

Le problème de choisir le service web le plus approprié parmi les services disponibles qui répondent en mieux aux exigences fonctionnelles avec des fonctionnalités similaires se heurte à l'utilisateur dans une situation délicate pour sélectionner un service adéquat, ce problème est de type NP-hard avec une complexité exponentielle.

Dans cette situation, la qualité du service est considérée comme une approche notable pour la sélection. En outre, l'utilisateur de services exige un service de haute qualité pour la composition du service, car un service de faible qualité peut affecter la qualité du service global.

De ce fait, la sélection de service Web avec qualité non fonctionnelle devient l'un des débats les plus importants et l'un des sujets de recherche et d'études rattaché au domaine des services web et leur composition.

III. Contributions

Notre travail consiste à chercher et retourner une meilleure sélection des services web composites en fonction de leurs(QoS).Pour cela, nous avons proposé une approche d'optimisation mono objective basé sur les méta-heuristiques en particulier l'algorithme de cuckoo search (CA) qui se base sur le comportement des coucous et leur mode de vie ainsi que de leur productivité.

, nous avons étudié et adapté cet algorithme pour générer des services web composites tentant en compte des contraintes de QoS l'avantage de cet algorithme est sa simplicité par rapport à d'autres algorithmes méta-heuristique tels que l'optimisation des essaims de particules et la recherche d'harmonie, il n'existe essentiellement qu'un seul paramètre dans Cuckoo Search (en dehors de la taille de la population n). Par conséquent, il est très facile à mettre en œuvre.

IV. Plan du mémoire

Ce manuscrit est constitué de trois chapitres et une conclusion générale, qui sont organisé comme suit :

Chapitre 1 « Les services web » : Dans ce chapitre nous avons introduit d'abord le concept de service web ,ensuite nous avons présenté les propriétés opérationnelles de service web en mettant l'accent sur la qualité de service (QoS). En fin, nous avons abordé le principe de la composition des services web.

Chapitre 2 « Sélection des services web composites basé sur QoS » : L'objectif de ce chapitre est d'introduire l'état de l'art de la problématique de sélection les service web composites en se basant sur les aspects non fonctionnels (QoS) .Dans un premier temps, nous procéderons à une brève définition de la sélection et ses stratégies .Dans un deuxième temps, nous proposerons notre scénario de motivation.

Finalement, nous citerons quelques approches pour résoudre ce problème.

Chapitre 3 « La sélection des services web basé sur la recherche coucou » :

Notre proposition portera sur résolu le problème de la sélection du service web composite basée sur la qualité service (QoSWSC).

Nous commencerons par un exemple illustratif et les contraintes globale de QoS ainsi la fonction objectif ,nous traiterons ensuite l'algorithme de la recherche coucou en citant quelques travaux concernant cet algorithme.

Nous adapterons en final une méta heuristique cuckoo search au problème QoSWSC et pour clarifier le déroulement de cette dernière nous montrons la conception de notre système et son implémentation ainsi une série d'expérimentations sur une base de données afin d'évaluer la performance de l'approche proposée.

Nous terminons ce mémoire par une conclusion générale et des perspectives.

Chapitre 1

Les services Web

« Most science is only high falutin' nature studies.»

Stephen Strauss

1. Introduction

L'indispensabilité de l'internet partout dans le monde et la croissance continue des utilisateurs et leurs besoins, ainsi la taille et la complexité des logiciels, étaient le point principale pour développée des approches qui améliore la distribution des informations et augmente la rapidité de l'interaction entre les individus, parmi ces approches nous citons les services web qui sont une expérience inédite dans le domaine informatique ou ils offrent des fonctionnalités aux applications au travers du réseau en utilisant des standards ouverts Ils peuvent donc être utilisés par des applications écrites dans différents langages et exécutées dans différentes plate-forme sur différents systèmes puis ils permettant l'interopérabilité entre plusieurs systèmes logiciels sur un réseau informatique. Pour ces raisons ils sont une technologie excellente.

Dans ce chapitre, trois points principaux seront traités:

Le premier, présente l'approche à services en commençant par une définition de la notion de service, ensuite une présentation des différents acteurs de l'approche à services.

Le deuxième, définit les propriétés fonctionnelles et non-fonctionnelles en premier lieu et en deuxième lieu définit la qualité de service web, dans ce point nous nous intéressons plus particulièrement à la description des propriétés non-fonctionnelles.

Le troisième, aborde le principe de la composition des services web.

2. Définition de service web

Lorsqu'on parle de Web Services, on parle aussi d'architecture informatique du type (client – serveur) où les postes client sont les ordinateurs, les terminaux mobiles...etc. Vont connecter sur un serveur via un réseau, de manière générale via Internet.

Nous retrouvons plusieurs définitions des services Web, nous citons quelques unes :

Selon W3C[1]

Définition 2.1 :
Un service Web est un composant logiciel identifié par une URI, dont les interfaces publiques sont définies et appelées en XML. Sa définition peut être découverte par d'autres systèmes logiciels. Les services Web peuvent interagir entre eux d'une manière prescrite par leurs définitions, en utilisant des messages XML portés par les protocoles Internet.

Selon IBM[2]

Définition 2.2 :
Les services Web sont un ensemble de normes émergentes qui permettent une intégration interopérable entre des processus et des systèmes informatiques hétérogènes. Ils sont considérer comme une nouvelle génération d'applications Web autonome et auto-déclarant, qui peut fournir une fonctionnalité et une interopérabilité allant des processus métier et scientifiques les plus complexes.

[1] Le ***World Wide Web Consortium*** est une organisation non lucrative permettant définir des standards pour les technologies liées aux web.

[2] ***IBM*** est une société multinationale américaine présente dans les domaines du matériel informatique, du logiciel et des services informatiques.

Selon déco de net[3] :

Définition 2.3:

Technologie permettant à des applications de dialoguer à distance via Internet indépendamment des plates-formes et des langages sur lesquelles elles reposent.

La plus simple définition pourrait être « fonctionnalité utilisable au travers du réseau en mettant en œuvre un format standard utilisant généralement les méta-langages XML ».

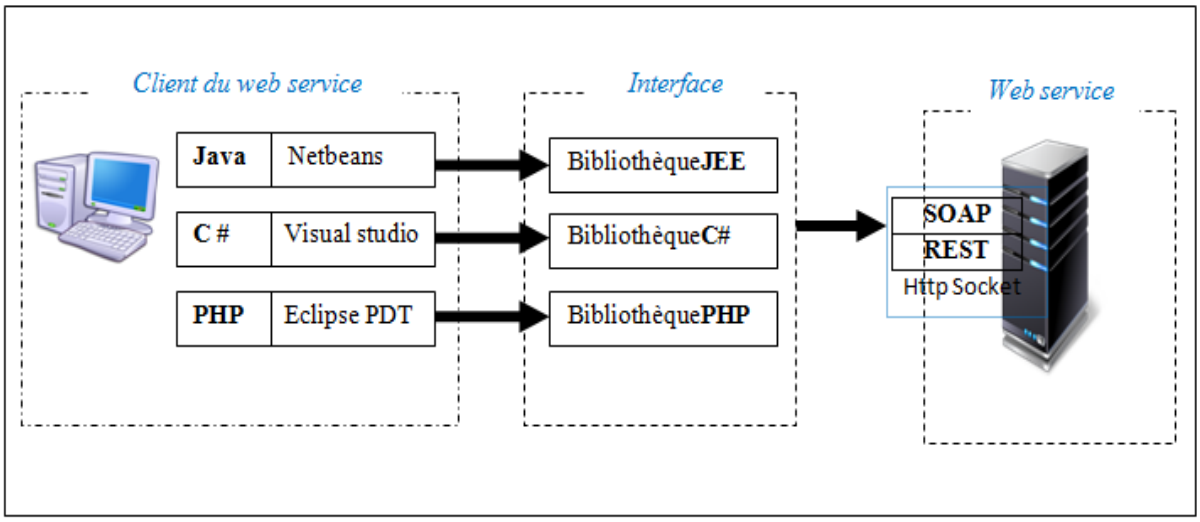


Figure 1.1: l'architecture des services web

Le service repose sur trois éléments bien particuliers :

- **Le protocole de description WSDL[4] :** permettre de décrire les différents services.

[3] Le ***Dico du Net*** est un dictionnaire collaboratif en ligne. Dans un premier temps focalisé sur les e-technologies (le référencement, la mesure d'audience, l'hébergement de sites, la création de sites web, le développement de logiciels, le moteur Google, DMOZ, les weblogs, les noms de domaine, les normes d'Internet, l'e-Marketing et l'e-Commerce...), il est désormais ouvert à tous les thèmes.

[4] ***WSDL*** (Web Service Description Language) pour le service de description : utilisé pour décrire un service à l'aide des protocoles et leur localisation au sens internet (URI / URL).

- **Le protocole de description messages SOAP :** permettre de gérer la communication et échange des données.

- **Le protocole de description des services de transport HTTP[5]/SMTP[6] :** pouvoir faire communiquer le poste client avec le serveur habituellement nous utilisons des connections du type **http** via navigation Internet.

3. Les aspects fonctionnel et non fonctionnel

3.1. Les aspects fonctionnels de service web

Les aspects du service web reflètent son fonctionnement en déterminant dans la description de service dans les paramètres d'entées/sorties et prés-conditions et les méthodes fournies.

3.2. Les aspects non fonctionnel de service web

Les aspects non fonctionnels expriment généralement la qualité service, qui est une ensemble de critères qui confie a un service la capacité à satisfaire des besoins déclarés ou implicites.

Ces besoins liés à des paramètres classifiés selon leur déterminisme.

- **Les paramètres déterministes :** lorsque sa valeur est certaine quand le SW invoqué (prix, pénalité...).

- **Les paramètres non déterministes :** lorsque sa valeur est incertaine quand le SW invoqué (durée d'exécution).

Nous pouvons résumer les critères de QoS dans ce tableau ci-dessus :

[5] ***HTTP*** (*Hypertext Transfer Protocol*) : Ce protocole définie la communication entre un client (exemple: navigateur) et un serveur sur le World Wide Web (WWW).

[6] ***SMTP*** (Simple Mail Transfer Protocol) est un protocole simple pour le Transfert des Mails.

Catégories de QoS	Attributs de QoS
Performance	Temps d'exécution, débit, temps de réponse et latence.
Sureté de fonctionnement **« Dependability »**	Authentification, accessibilité, exactitude, fiabilité, capacité, mise en échelles, stabilité, robustesse.
Attributs spécifiques aux domaines d'application	Fourchette de prix, méthode de payement, taux de pénalité.
Sécurité	Authentification, audite, non-répudiation, confidentialité, intégrité.

Tableau 1.1 : les catégories des QoS

Il existe d'autres classifications selon différents critères par exemple propriétés d'environnement, propriétés métiers, propriétés temporelles, etc.

4. Les types des services web

Les services web peuvent être classées selon abstraction, leur type d'invocation, leur description, ou encore leur granularité.

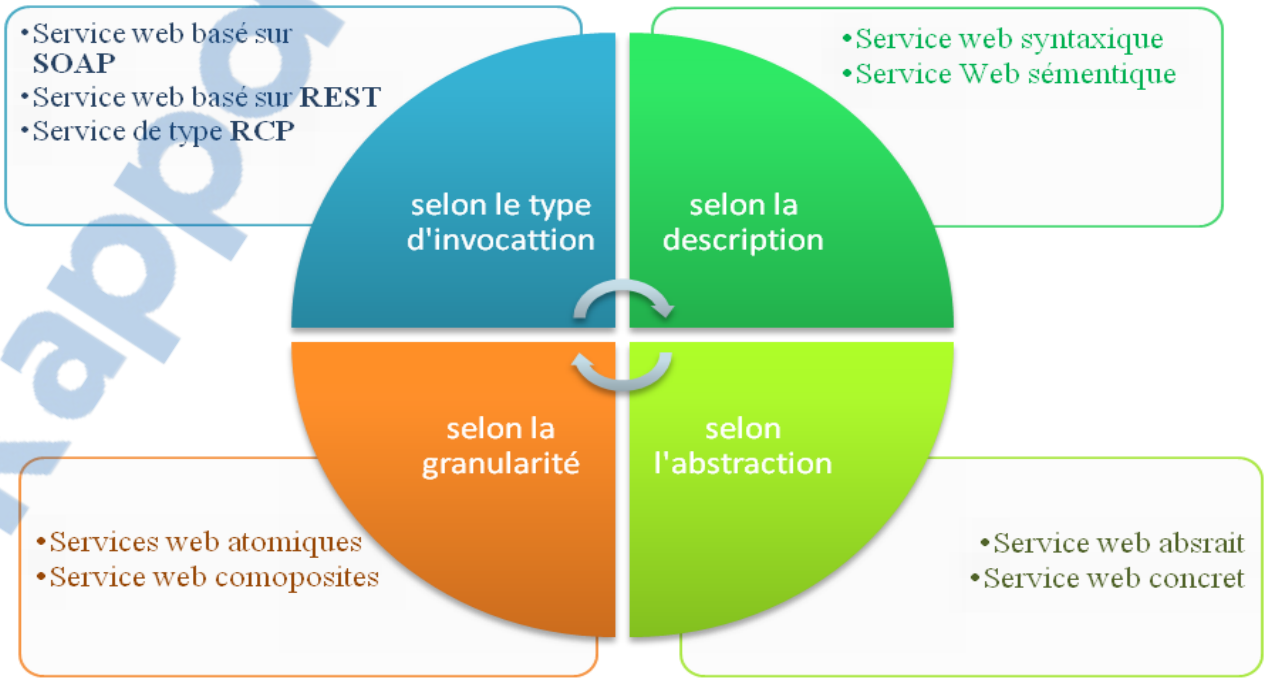


Figure 1.2 : Les types des services web

Selon le type d'invocation :

- Service web basé sur **SOAP** (Simple object Access Protocol) : est un protocole standard de communication. C'est l'épine dorsale du système d'interopérabilité. SOAP est un protocole décrit en XML et standardisé par le W3C. Il se présente comme une enveloppe pouvant être signée et pouvant contenir des données ou des pièces jointes.
  Il circule sur le protocole HTTP et permet d'effectuer des appels de méthodes à distance. (Nebra, 10)

- Service web basé sur **REST** (Representational State Transfer) : est une architecture de services Web. Élaborée en l'an 2000 par *Roy* Fiedling, l'un des créateurs du protocole HTTP, du serveur Apache http et d'autres travaux fondamentaux, REST est une manière de construire une application pour les systèmes distribués comme le World Wide Web. (Nebra, 10)

- Service web basé de type **RCP :** est un protocole simple utilisant XML pour effectuer des messages RPC. Les requêtes sont écrites en XML et envoyées via HTTP POST. Les requêtes sont intégrées dans le corps de la réponse HTTP. XML-RPC est indépendant de la plate-forme, ce qui lui permet de communiquer avec diverses applications. (Nebra, 10)

Selon l'abstraction :

Service web concret : est une fonction qui agit sur des données d'entrée pour fournir des résultats en sortie. Il est décrit par des propriétés fonctionnelles et non-fonctionnelles. La partie fonctionnelle englobe généralement les données d'entrée du service, Les données de sortie du service *et* les pré-conditions du service, et les effets du Service. En revanche, la partie non-fonctionnelle est définie par : un ensemble d'informations contextuelles auxquelles le service est sensible (par exemple la localisation), des attributs de la qualité de service, et des caractéristiques relatives. (KHANOUCHE, 2016 – 2017)

- **Un service abstrait** (classe de services) : représente un ensemble de services concrets qui offrent des fonctionnalités similaires. Les services concrets d'une même classe ont les mêmes données d'entrée et produisent le même résultat en sortie. Un service abstrait est décrit par ses données d'entrée, ses données de sortie et l'ensemble de ses services concrets. (KHANOUCHE, 2016 – 2017)

Selon la description :

- **Services Web Syntaxiques :** La description syntaxique du profil fonctionnel contient les informations relatives aux méthodes proposées, leurs paramètres et les protocoles à utiliser. Ces informations permettent de faire l'appel du service. (Halfoui, 2017)

- **Services Web Sémantiques** : un service Web est dit sémantique si sa description intègre, en plus de la description du profil fonctionnel, une dimension sémantique. Cette dimension englobe toute description complémentaire (du fournisseur, des objectifs du service Web, etc.) de la description fonctionnelle qui permet d'ajouter de l'information à la description classique du service Web. Cette information peut ensuite être traitée par des mécanismes (tels que l'inférence) afin d'en extraire de la connaissance**.** (Halfoui, 2017)

Selon la granularité :

- **Service atomique** (élémentaire): offre une fonctionnalité de base et représente la forme la plus granulaire d'une architecture orientée services. Un service atomique peut être créé en utilisant différents langages (Java, C++, Perl, Python, etc.) et déployé sur diverses plateformes (J2EE, .NET, etc.). Lors du traitement d'une requête, un service atomique ne fait pas appel à d'autres services. Il est alors caractérisé par une exécution autonome dans la mesure où il n'invoque pas d'autres services lors de son exécution. Puisque l'exécution des services atomiques ne nécessite pas de résultats provenant d'autres services. (KHANOUCHE, 2016 – 2017)

- **Service composites:** les technologies sur lesquelles s'appuient ces services sont relativement mature. L'un des principes de base de l'Informatique Orientée Services est que les services atomiques peuvent être combinés (composés) pour créer un nouveau service répondant à une fonctionnalité plus complexe.

  Nous parlons dans ce cas de service composite, et en détaille ce derniers dans la section suivante. (Xiang, Apr. 2007)

5. Composition de service web

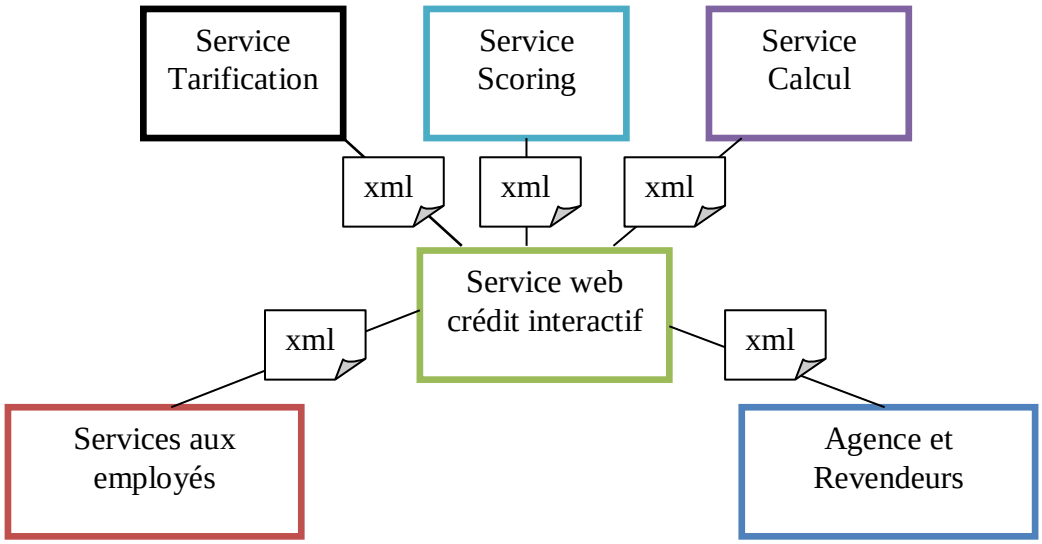


Figure 1.3: exemple d'un service composite « service web crédit interactif »

5.1. Définition

Il existe plusieurs définitions pour le concept de composition, nous citons les plus communes :

« Le processus de sélection, de combinaison et d'exécution de services en vue d'accomplir un objectif donné ». (F. Casati, 2002)

Selon [Georges Gardin] qui définit la composition de service web comme : « technique permettant d'assembler des services web afin d'atteindre un objectif particulier, par l'intermédiaire de primitives de contrôle (boucle, test ; traitement d'exception, etc.) et d'échange (envoi et réception de messages). Les services composants existent au préalable et peuvent ne pas être fournis par la même organisation ». (Gardarin, 28 Nov 2002)

En d'autres termes, la composition de service web est un processus complexe qui consiste a construire un service composite offrent des nouvelles fonctionnalités, à partir des services web existants et publies sur le web, par le processus de découverte , d'exécution et d'intégration dans un ordre bien définis afin de satisfaire un besoin bien défini et déterminé par l'utilisateur .

5.2. Cycle de vie d'une composition des services web

Le cycle de vie d'une composition de services web est définit à partir de six activités (B. Benatallah, 2005) :

- **L'encapsulation de services natif (Wrapping services) :** cette première activité garantit que tout service peut être appelé lors d'une composition sans mettant l'accent de son modèle de données, son protocole d'interaction et de son format de message.

- **L'établissement d'accord d'externalisation (Setting outsourcing agreements)** :

  Cette seconde activité est constituée à établir et négocier des obligations contractuelle entre les services.

- **L'assemblage de service de services composants (assembling composite** services) :

  Cette activité décrire la notion d'assemblage qui compote deux phases, une phase pour identifier les services à utilise et une pour spécifier les échanges et interactions inter service à un haut niveau d'abstraction. (Elle consiste aussi à décrire les conventions et les descriptions externes de SWC).

- **L'exécution de services composants (executing services) :**

Cette activité satisfait l'exécution les spécifications de la composition selon des critères particulières.

- **Le contrôle de l'exécution de service composite (monitoring services) :**

  Cette tache permet de détecter les erreurs, de vérifier les aspects et les propriétés ainsi de mesurer les performances des services appelés et prédire des exceptions.

- **L'évolutivité des services (Evolving services) :**

  Cette tache garantit l'évolution de la composition des services la modification des services appelés, utilisation de nouveaux services et des retours du contrôle.

.

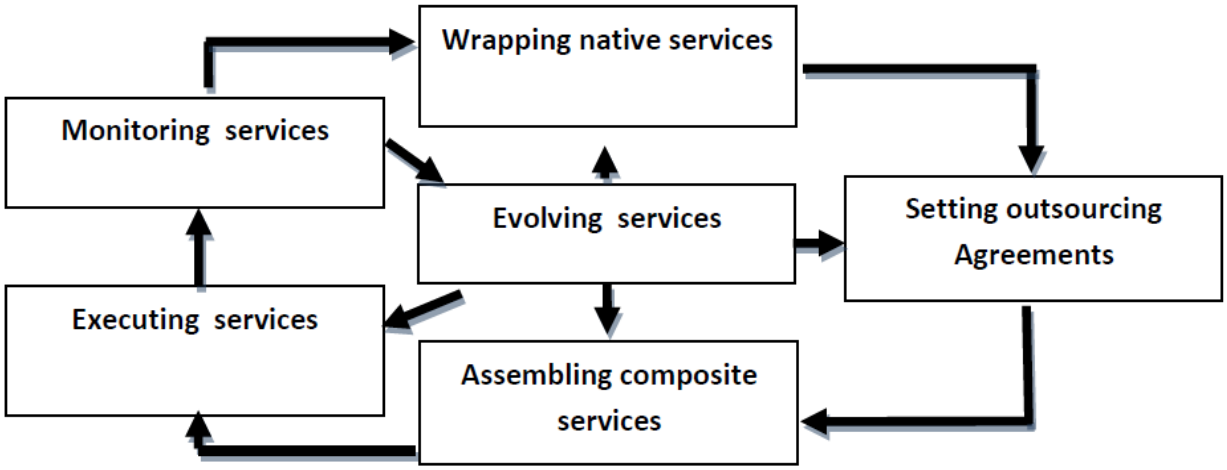


Figure 1.4 : Cycle de vie d'une composition de services (B. Benatallah, 2005)

5.3. Nature de la composition des services

La composition permet de combiner des services selon certaines règles, pour répondre la requête demandée par un utilisateur, la description de la composition est réalisée d'une manière statique ou dynamique en fonction du degré d'automatisation plus moins différent.

La composition en fonction du degré de participation de l'utilisateur dans la définition du schéma de composition des service web (M.Chalbabi, novembre 2006) :

- ✓ **La composition manuelle :** consiste que l'utilisateur qui gère la composition à la main à l'aide un éditeur de texte simple sans utilisé des outils spécialisés.

Ce type de composition exige de l'utilisateur qu'il ait le profil programmeur.

- ✓ **La composition automatique :** la composition totalement automatisée sans qu'aucune intervention de l'utilisateur.

Dans ce type en prend en charge tout le processus de composition et le réalise automatiquement.

- ✓ **La composition semi-automatique :** cette technique compris entre la composition manuelle et la composition automatique, elle permet à l'utilisateur

de garder des certaines suggestions sémantique pour aider à la sélection des services web dans le processus de composition à l'aide des outils dédiés.

La composition en fonction de la sélection des services web et la gestion de flot :

- ✓ **La composition statique des services web :** Les services web sont composés durant la conception du service composite, les composants sont choisis et reliés un ensemble avant d'être compilés.

Si l'utilisateur demande un autre service ou modification d'un service, ce changement nécessite d'apporter une modification en niveau de la conception du système.

- ✓ **La composition dynamique des services web :**

Les services web sont sélectionnés et composés au moment de l'exécution de la requête à l'aide des annuaires des services web, tout en respectant des contraintes de l'utilisateur.

6. Les principales approches de composition de sw

6.1. Orchestration de service web

L'orchestration de services Web résulte un nouveau service Web dit service Web composé, qui peut être défini comme l'agrégation de plusieurs autres services Web atomiques ou composés. Ce service composé contrôle la collaboration entre les services Web engagés dans la composition, tel qu'un chef d'orchestre **(**DUM ,10**)**.

Pour des compositions de services simples, l'orchestration est faite dans le code (Java, C#...) résidant dans le composite et pour des orchestrations complexes, un outil est utilisé pour :

- Créer un modèle visuel d'une séquence.
- Générer le code qui exécute cette séquence dans un environnement d'exécution dédié.

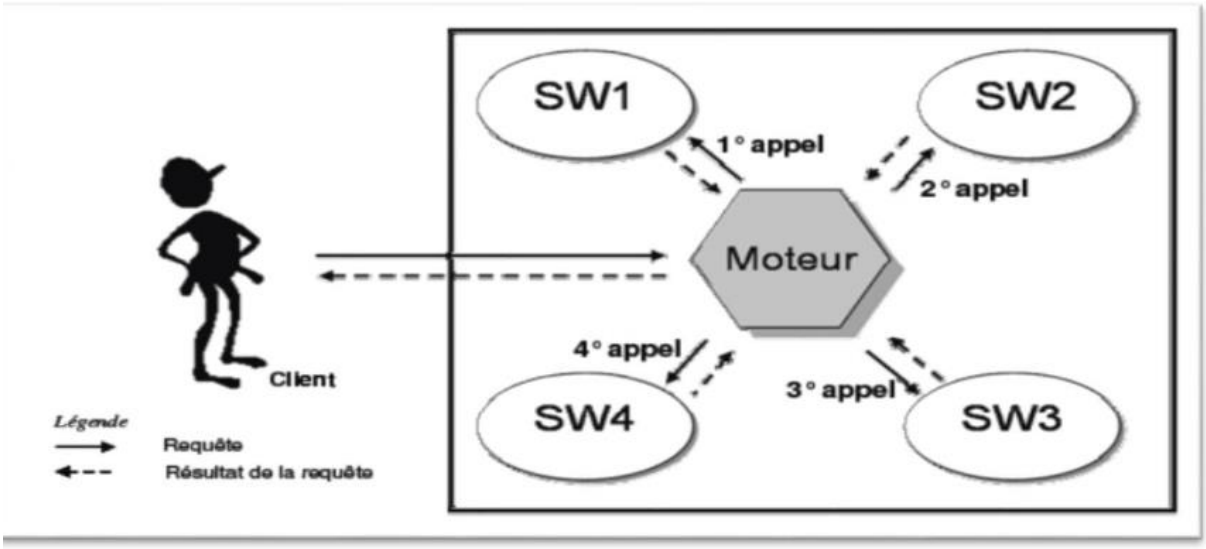


Figure 1.5 : Vue générale de l'orchestration

6.2. Chorégraphie de services web

La chorégraphie de services Web est une généralisation de l'orchestration qui consiste à concevoir une coordination décentralisée des services Web. Dans une chorégraphie, les interactions de type pair-à-pair (P2P) sont décrites dans un langage de description de chorégraphie (CDL). Les services suivent alors le scénario global de composition sans point de contrôle central (DUM ,10).

La chorégraphie est aussi appelée composition dynamique basée sur deux approches :

- Basée sur les messages
- Basée sur les composants fonctionnels

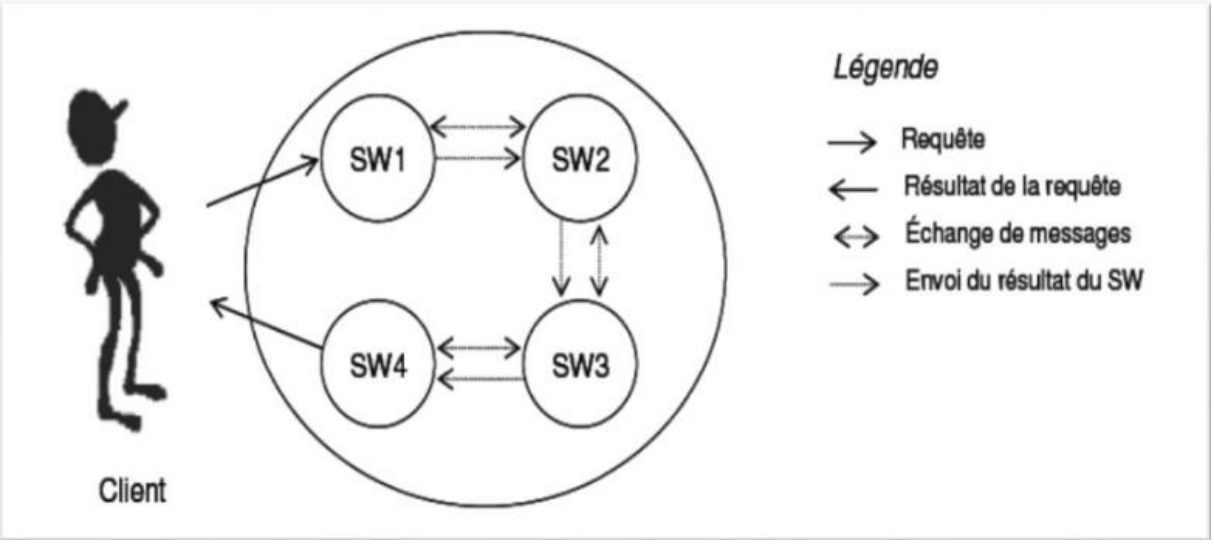


Figure 1.6 : Vue générale de l'exécution d'une composition de services Web de type chorégraphie

7. Conclusion

Les Web services sont Est une technologie importante, ce chapitre a été dédié a la présentation de cette technologie, son aspect non fonctionnel et en dernier lieu la composition des services web.

Nous présenterons dans le chapitre suivant un état de l'art sur le problème de sélection des services web composites« QoS-aware service composition »qui est l'un des problématiques les plus importants de l'architecture orientée service.

Chapitre 2

Sélection des services Web composites basée sur QoS

La plus grande problématique de la vie c'est de ne pas connaitre ce qu'on ne connait pas; Et en cherchant à connaitre ce qu'on ne connait pas, on devient étudiant pour toujours.

Landry Makana

1. Introduction

Avec la croissance explosive du nombre de services publiés sur internet, il est difficile de sélectionner des services web satisfaisants parmi les services web candidats qui offrent des fonctionnalités similaires. La qualité de service (QoS) est considérée comme le critère non fonctionnel le plus important pour la sélection du service.

Dans ce chapitre, nous présentons l'état de l'art qui introduit un problème spécifique émergeant qui est celui de sélectionner le meilleur ensemble de services en respectant les contraintes de QoS définies par les utilisateurs. Pour cela, nous décrivons les différentes stratégies de sélection et nous citons quelques approches utilisées pour résoudre ce problème.

2. Définition de sélection service web

La sélection de service web consiste à choisir les meilleurs services parmi un ensemble de services découverts en tenant en considération les descriptions des clients qui sont déclarées à l'aide d'une interface qui identifie leurs préférences pour qualifie le service . Nous distinguons deux manières de sélection : sélection statique et sélection dynamique.

	Sélection statique	Sélection dynamique
La découverte et la sélection	Manuellement en se basant sur les protocoles de découverte telle qu'UDDI	Dynamiquement en se basant sur des représentations compréhensibles par la machine telle qu'OWLS (Web Ontology Language for Services)
Les propriétés non fonctionnelles	L'utilisateur qui les décrit.	des propriétés du service mesurées ou estimées durant son exécution (Clement, 2004).
Le service sélectionné	Une fois un service sélectionné on ne peut pas le modifier(le remplacer) par la suite si des nouvelles exigences apparaissent.	Le service sélectionné est choisi une seule fois, puis lorsque de nouvelles exigences apparaissent, un autre service peut être sélectionné. (D. Roman, 2005)

Tableau 2.1.Tableau comparatif entre la sélection statique et la sélection dynamique.

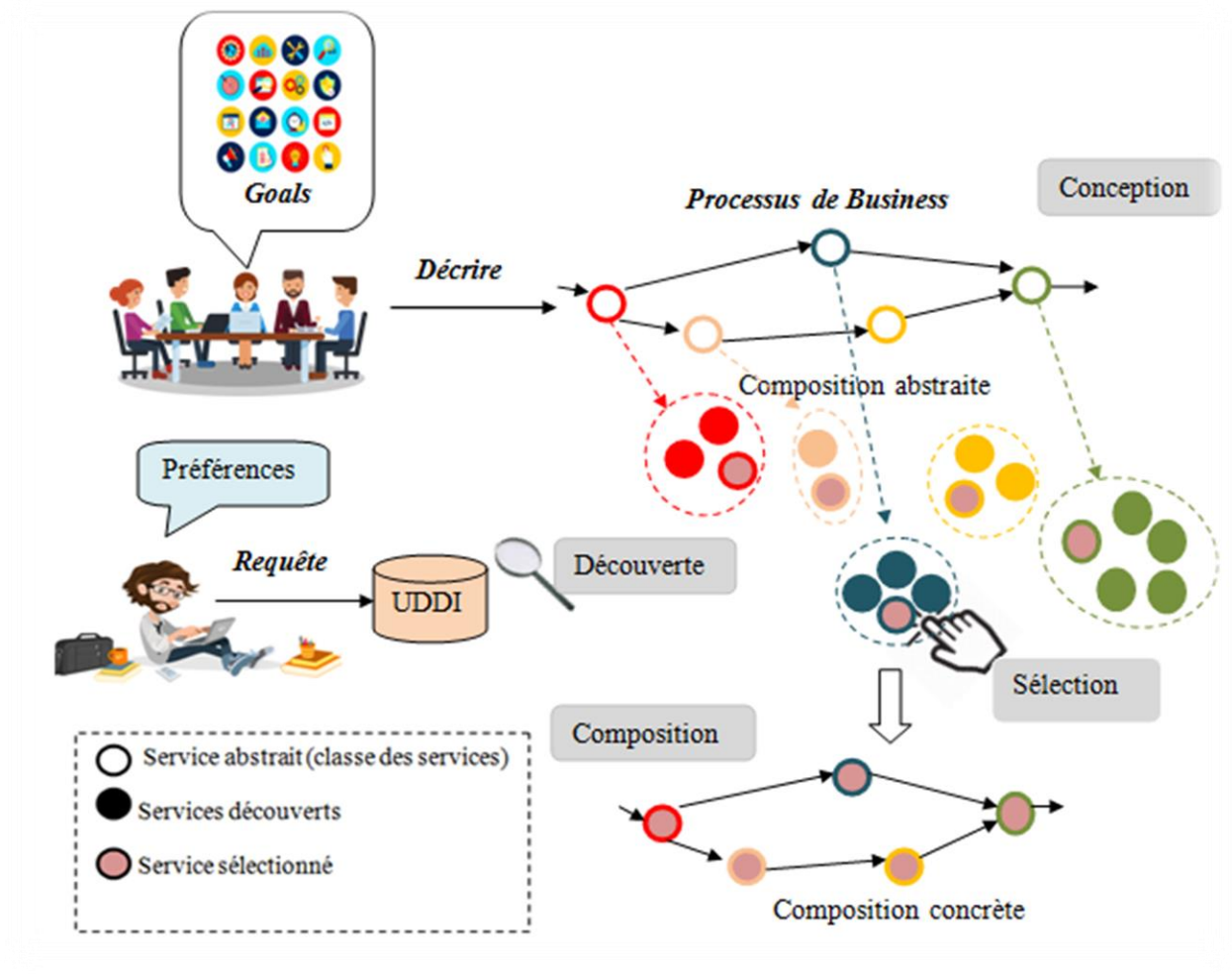


Figure 2.1:Sélection des services web composites.

3. La sélection basée sur la QoS

Dans une sélection de services web basée sur la Qos deux cas se présentent (Jaeger, 2006):

- ✓ Si la réponse à la requête d'un client exige la sélection d'un seul service web non composite, alors la sélection est très simple.
  Le candidat qui présente la meilleure Qos sera désigné pour répondre à la demande du client.
- ✓ Si la réponse à la requête d'un client exige la combinaison de plusieurs services existants, alors la sélection dans ce cas sera plus complexe du fait qu'il faut choisir la combinaison des services composants qui répond mieux aux besoins des clients.

La deuxième éventualité émet une hypothèse de sélection des service web composites suivant la description non fonctionnelle. La modélisation de cette hypothèse dépend de la nature de la stratégie de sélection.

4. Les stratégies de la sélection

Dans la littérature, il existe deux stratégies de sélection de services web (Zeng L. B., 2004) : Une stratégie de sélection locale et une stratégie de sélection globale. Chaque stratégie est définie en fonction de la nature des contraintes de QoS imposées.

En fait, ces contraintes n'ont pas un caractère obligatoire, mais permettent de définir les limites de fonctionnement d'un service web composant ou composite.

4.1. La stratégie de sélection locale (B. Benatallah, 2005) (Jaeger, 2006) :

A pour objectif de choisir le meilleur service web pour chaque tâche individuelle à part entière en considérant des contraintes de QoS relatives à chaque tâche plutôt qu'en considérant des contraintes de QoS globales exprimées pour l'ensemble des tâches. Il s'agit de sélectionner pour chaque tâche un service web apte à l'exécuter en prenant en compte les contraintes locales et d'autres préférences imposées pour chaque tâche.

Par exemple, la durée d'exécution d'une tâche devra avoir un temps de réponse qui ne dépasse pas quelques minutes, le service web exécutant une tâche doit être disponible à 100%, etc.

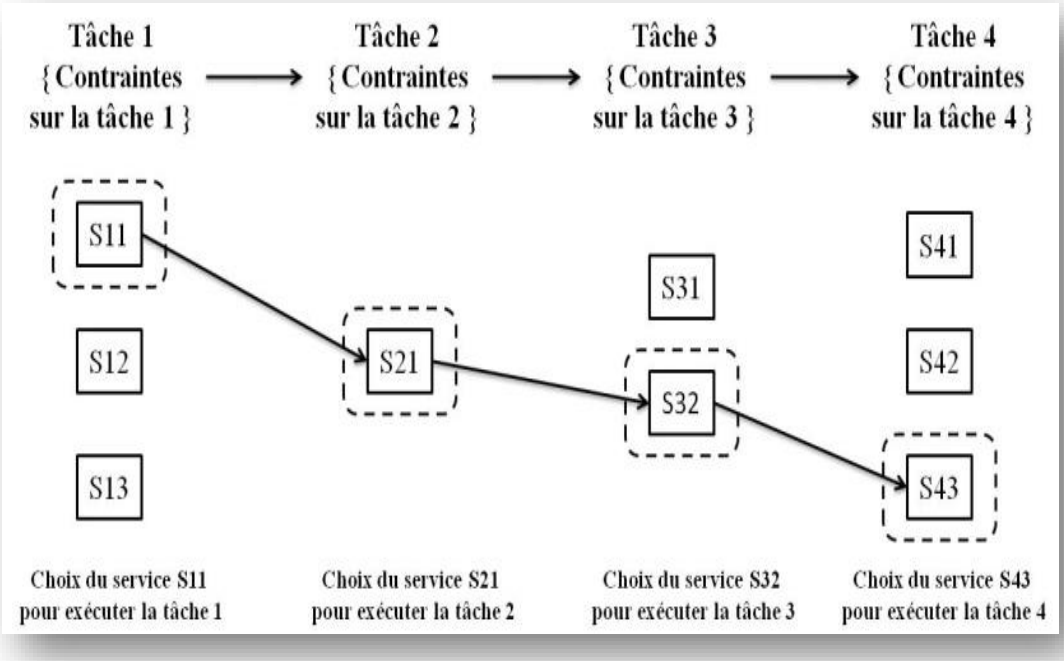


Figure 2.2 : Choix de service web pour chaque tâche.

4.2. La stratégie de sélection globale (Yu T. a., 2004) (Zeng L. B., 2003)

A pour but de choisir la combinaison de services web qui garantit la meilleure qualité globale en tenant compte des contraintes de QoS et des préférences globales assignées pour l'ensemble des tâches.

il s'agit de sélectionner la combinaison de services qui offre la meilleure QoS et qui respecte les contraintes et d'autres préférences globales imposées pour l'ensemble des tâches de l'application comme par exemple imposer que la durée d'exécution de bout en bout doit être inférieure à une échéance.

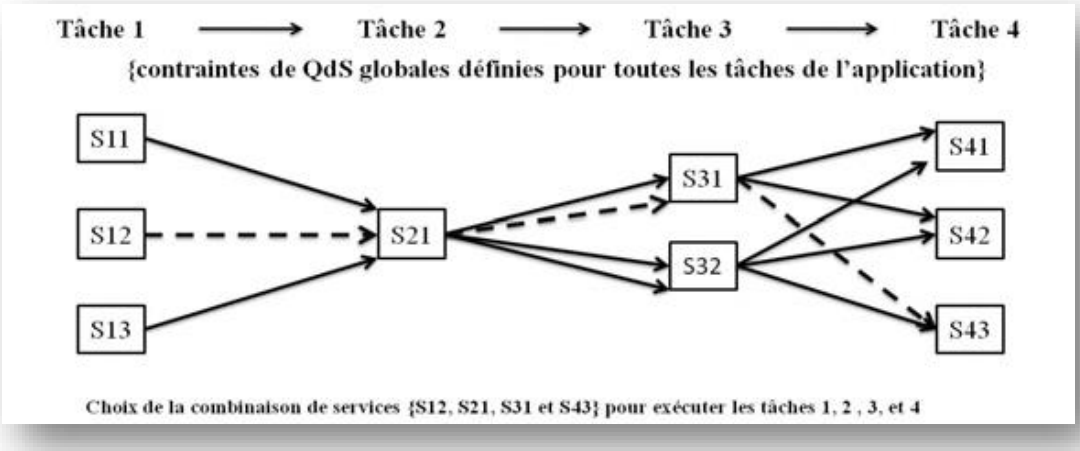


Figure 2.3 : Choix d'une composition de services web.

5. Exemple présentatif du problème sélection des services web composites basé sur qualité de service (QoSSWC)

D'abord pour atteindre un service web qui nous intéresse est une chose et pour atteindre le service le plus pertinent c'est totalement autre chose.

Pour satisfaire nos besoins parfois, il est nécessaire de fusionner un ensemble des services atomiques (simples) en un service composite, et cette dernière combinaison sera réalisée selon des métriques fonctionnels/non fonctionnelles., trouver ou choisir les meilleurs services parmi les services découverts est une tâche complexe et difficile à réaliser, c'est le problème « QoS-aware service composition » où la sélection des services web composites constitue l'un des problèmes les plus importants de l'architecture orientée service (SOA).

De ce fait, nous proposons cet exemple qui présente le service composite d'une inscription dans une université à l'étranger :

Afin de continuer des études à l'étranger une étudiante doit compléter certaines étapes qui suivent la préinscription au programme d'étude voulu à l'université choisie

Ces étapes sont des services proposés à l'étudiante ou elle va sélectionner celles qui lui conviennent le mieux selon ses moyens et préférence.

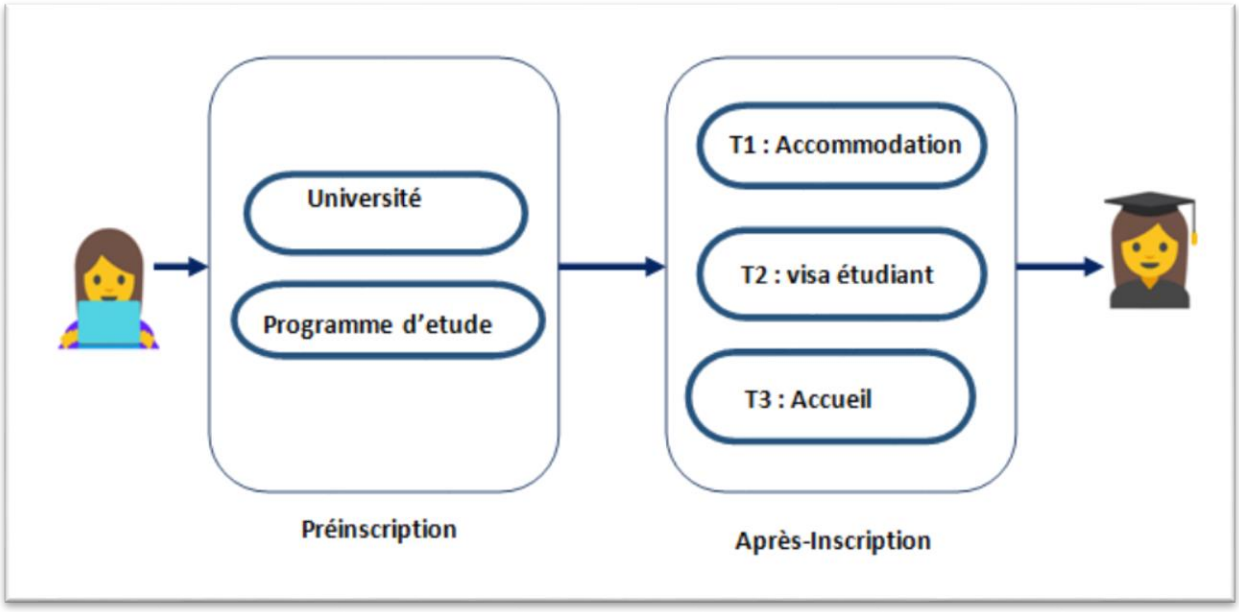


Figure 2.4 : Scénarios de motivation.

Le choix de la bonne composition des services est le but à atteindre par la sélection, la composition commence par le service d'accommodation qui est faite juste après l'inscription.

Les documents fournies de **la tache 1** (accommodation) seront utilisés pour l'obtention du visa (la tache 2), la dernière tâche st l'accueil à l'aéroport, plusieurs services similaire sont offert pour chaque tâche par conséquence la sélection sera faite selon l'attribut QoS qui diffère d'un service a un autre.

Le Tableau ci-dessus contient les services disponibles dans chaque tache avec leurs QoS qui vont nous aider à choisir un service pour chaque tache afin d'obtenir une solution quasi-optimal. Nous avons trois services candidats par tâche et 3 propriétés *QoS* : $q1$, $q2$, $q3$ dans notre exemple (correspondent respectivement à la disponibilité, temps d'exécution et le cout).

L'utilisateur peut choisir des contraintes locales par exemple :

Le service le cout minimal dans chaque tâche, et le prix total de la composition ne dépasse pas un chiffre fixé (contrainte globale).

Les Propriétés *QoS* globales pour les services composites dépendent de :

• La structure de la composition (parallèle, séquence, etc.)

• Le service sélectionné pour chaque tâche.

Task	Services	Q_1(%)	Q_2 (ms)	Q_3($)
T1	S_{11}	90	150	350
	S_{12}	65	340	145
	S_{13}	72	250	250
T2	S_{21}	82	250	156
	S_{22}	59	320	230
	S_{23}	98	150	350
T3	S_{31}	99	200	150
	S_{32}	70	90	175
	S_{33}	85	100	200

Tableau 2.2: Les services candidats offert après inscription à l'université.

6. Formulation de problème

Le problème QoSWSC appartient à la catégorie des problèmes NP-hard, il est souvent résolut comme un problème mono-objectif ou multi-multi-objectif pour qui la solution se résume dans la maximisation des contraintes QoS locales et globales ou seulement globales selon la méthode de résolution, une manière de formuler le problème QoSWSC est la suivante :

Soient :

• $C = \{S_1, \ldots, S_n\}$, une composition abstraite qui représente la requête de

L'utilisateur, c.-à-d. les n classes de services S_i à consommer.

• $gc = \{gc_1, .., gc_n\}$, un ensemble de contraintes globales définies par l'utilisateur

Qui concerne les QoS de la composition.

• $Sw = \{s_{i1}, ..., r\}$ une composition concrète, c.-à-d. nous remplaçons chaque classe

S_i par un service concret S_{ij} 2 S_i

• $q(s_{ij}) = \{q_1, .., q_m\}$ vecteur des qualités du service S_{ij} .

• $Q(C) = \{Q_1(C), .., Qm(C)\}$ vecteur des qualités du service de la composition C.

• $U(C)$ la fonction objectif à minimiser ou maximiser. Elle peut être :

– multi-objectif : dans ce cas, $U(C) = \{U_1...U_k\}$ où U_k représente la $k_{ième}$ fonction à optimiser.

– mono-objectif : dans ce cas, les *m* attributs de QoS sont agrégés en une seule valeur.

7.État de l'art des travaux sur le problème QoSSWC

Le but de la sélection des services web est de trouvé le service ou les services qui satisfit les besoins de l'utilisateur et pour cela plusieurs travaux ont vu le jour pour résoudre ce problème, ces travaux sont classifie en 3 grandes classes distincte selon leur stratégie, paramètre utilisé et résultat obtenue.

Les classes sont montrer dans la figure suivante (source : figure V.2. (FethAllah, 2014):

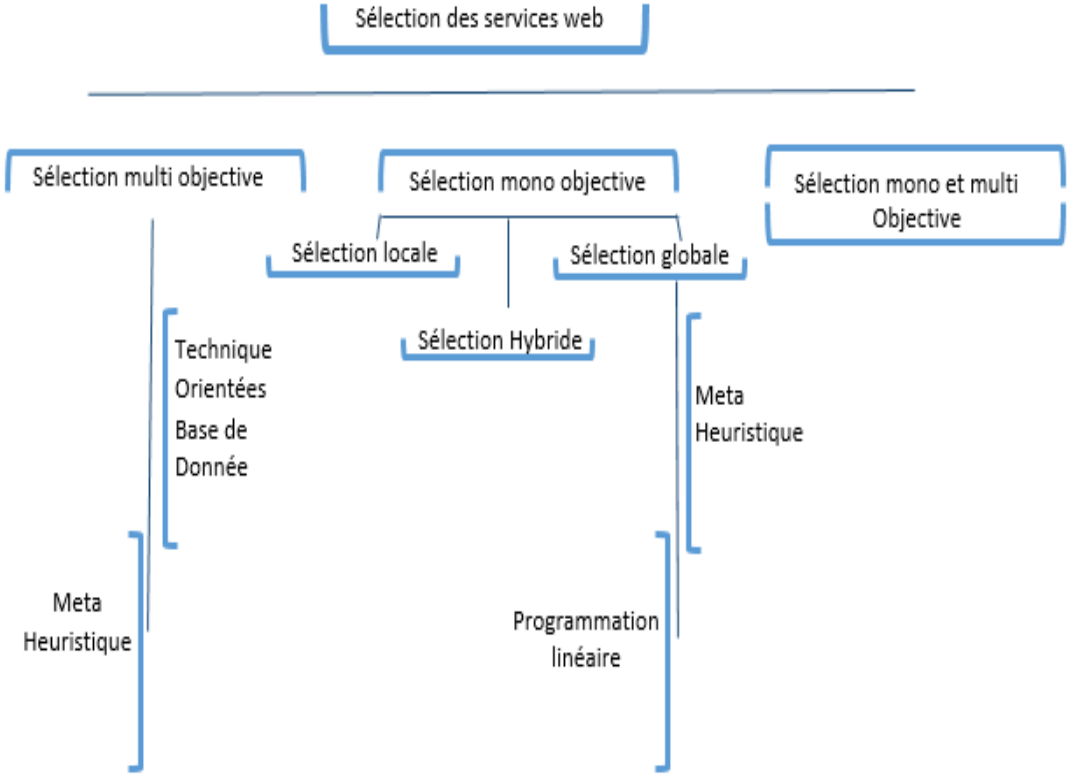


Figure 2.5 : Les classes de sélection des services web. (FethAllah, 2014)

Que ce soit une sélection mono objective, multi objective ou hybride (mono et multi objective),

Dans cet état de l'art nous nous concentrant sur les travaux d'optimisation de ce problème(QoSWSC), de premier rang 4 catégories peuvent être plus dynamique dans la résolution du problème de genre :

1. Technique de base de données.
2. Exacte.
3. Heuristique (approximative).
4. Meta-heuristique (approximative).

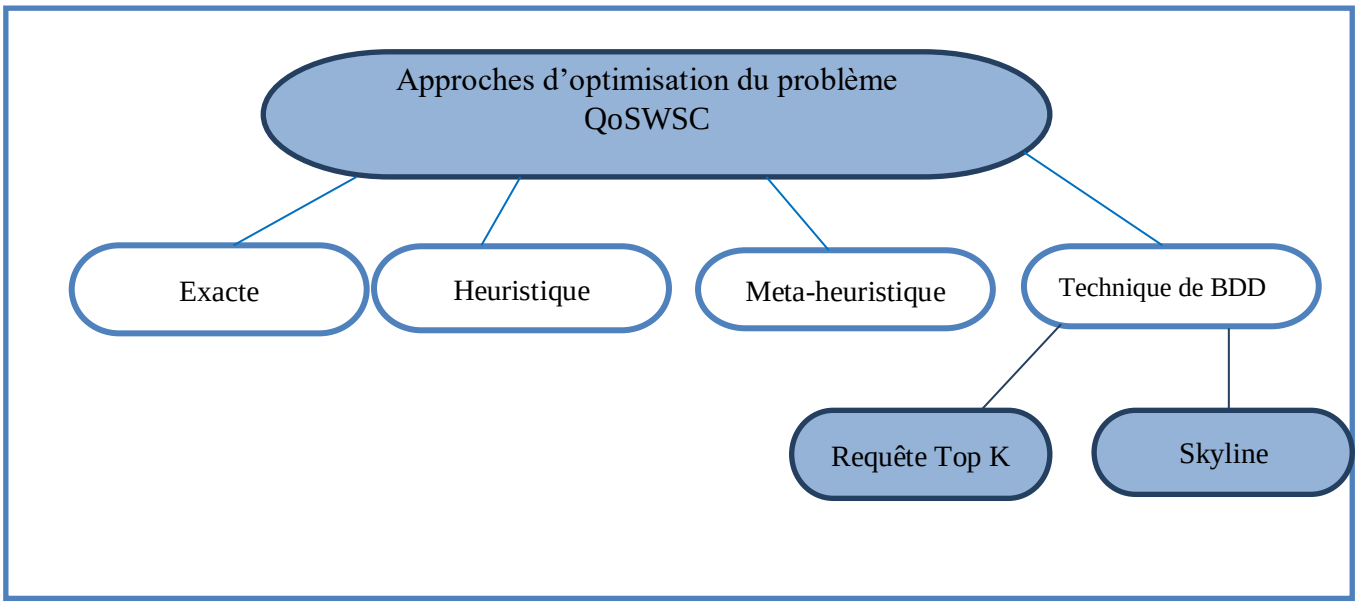


Figure 2.6:Approches de résolution de problème de sélection de SW (Halfoui, 2017).

7.1. Technique de base de données

La sélection des services web est traité comme un problème multi-objectif avec cette stratégie, qu'il soit traité en local ou en global (sans le réduire à une fonction mono-objectif). Elle repose surtout sur la notion de Pareto dominance et sur le principe des requêtes Skyline ou Top-k. ont utilisant des algorithmes pareil a Divide and conquer ou autres.

7.2. Approches Exactes

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

La figure suivante montre quelques méthodes exactes :

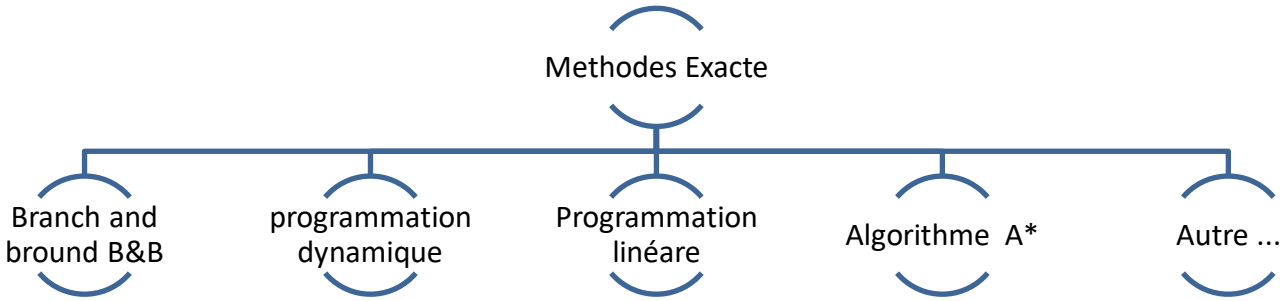


Figure 2.7 : Quelque méthode de résolution exacte.

Le Tableau qui suit contient quelques exemples des travaux utilisant des approches exactes concernant la sélection des services web avec explication.

Travaux	Explication
(Yu T. a.-J., 2005a) et (Liu, 2012)	présentent une méthode en considérant de multiples critères QoS en utilisant l'algorithme branch and bround et traitent différents types de work-flows.
(Yu T. a.-J., 2005b)	proposent une méthode qui permet de satisfaire les contraintes globales, et ont considéré le problème comme un problème de sac à dos à choix multiples. Ils ont utilisé l'algorithme branch and bround pour retourner le résultat.
(Rodriguez-Mier, 2010)	utilisent l'algorithme A*

Tableau 2.3:Quelques travaux utilisant des approches exactes concernant la sélection des services web.

L'inconvénient de ces approches exactes est la taille du problème traité, l'augmentation des variables (les tâches) aboutit à l'explosion de l'espace de recherche.

7.3. Approche heuristique (Approximative)

Une méthode heuristique est une méthode de résolution de problème qui ne s'appuie pas sur une analyse détaillée ou exhaustive du problème. Elle consiste à fonctionner par approches successives en s'appuyant, par exemple, sur des similitudes avec des problèmes déjà traités afin d'éliminer progressivement les alternatives et ne conserver qu'une série limitée de solutions pour tendre vers celle qui est optimale.

Les heuristiques prennent beaucoup plus moins de temps pour trouver la solution optimale par rapport au Méthode exacte.

Travaux	Explication
(Moustafa, 2013)et (Feng, 2013)	ils utilisent les algorithmes d'apprentissage par renforcement pour trouver l'ensemble de solutions Pareto optimales qui satisfait les facteurs QoS multiples et les exigences de l'utilisateur.
(Li, 2012)	adoptent la stratégie gagnant-gagnant (win-win stratégie) et proposent un algorithme qui porte le nom de la stratégie elle même.
(Lecue, 2009), (Klein, 2011)	utilisent l'algorithme hill climbing(Méthode de Descente) pour réduire la complexité de temps de calcul et comparent leur méthode avec celle utilisant LIP.

Tableau 2.4: Quelques travaux utilisant des approches heuristiques concernant la sélection des services web.

7.4. Approche Méta-heuristique (Halfoui, 2017)

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

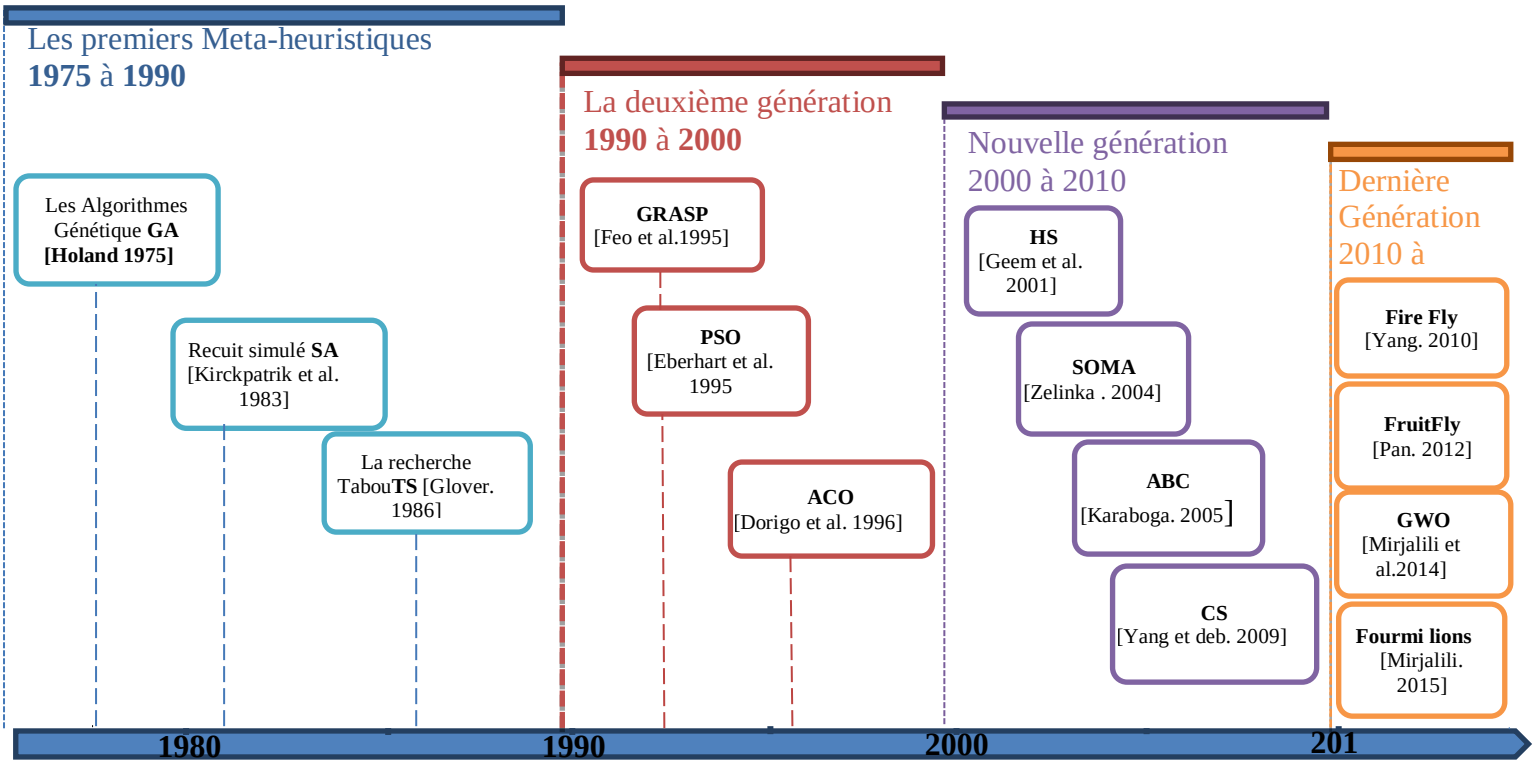


Figure 2.8 : La chronologie des principales Méta-heuristiques. (Halfoui, 2017)

Les Meta-heuristique se partage en deux classes :

Les méta-heuristique a base de solution unique

Cette catégorie initialise la recherche avec une solution unique puis elle l'améliore au cours d'une série d'itération en e basent sur la notion du voisinage. La solution initiale subit des modifications selon son voisinage afin d'améliorer progressivement sa qualité.

Nous trouvons dans ces classe deux familles d'algorithmes de recherche locale :

- ✓ pour une optimisation locale : recherche locale simple (la descente).
- ✓ pour une optimisation globale : Recuit simulé.

Travaux	Explication
(Ko, 2008)	proposent un algorithme de composition de service Web basé sur la satisfaction de contraintes qui combine la recherche Tabou et la méta-heuristique recuit simulé.
(Rosenberg, 2010)	la méta-heuristique recuit simulé.
(Hadjila, 2012)	la composition des services est modélisée sous forme d'un workflow. Les auteurs ne considèrent que la structure séquentielle et utilisent la recherche Tabou pour optimiser la fonction mono-objective qui considère 5 paramètres QoS.

Tableau 2.5:Quelques travaux utilisant des approches méta-heuristiques a base de solution unique.

Les méta-heuristiques à base de population de solution

Cette catégorie initialise la recherche avec ensemble de solutions afin d'obtenir la meilleure (solution optimal) qui est la solution du problème traité. L'utilisation d'un ensemble de solutions augmente la possibilité d'arriver une solution de bonne qualité.

Catégories de cette class :

- ✓ les algorithmes évolutionnaires.
- ✓ les algorithmes génétiques.
- ✓ algorithmes à base d'intelligence par essaims : Colonies de fourmis, recherche coucou.

Travaux	Explication
(Wang, 2008)	les auteurs proposent un algorithme génétique en utilisant une optimisation multi-objectif, ils définissent une fonction objectif pour chaque paramètre QoS considéré qui sont au nombre de 3.
(Ming, 2007)	montrent que l'optimisation en essaims particulaires PSO est plus rapide et plus sciable par rapport aux algorithmes génétiques (version standard sans hybridation), dans ce travail les auteurs modifient les services concrets des particules en changeant la vélocité avec des opérateurs tels que l'addition, la soustraction, la multiplication.
(Merzoug, 2014)	utilisent l'optimisation par harmonie pour résoudre le problème QoSWSC.

Tableau 2.6:Quelques travaux utilisant des approches méta-heuristiques à base de population de solution.

8. Conclusion

Ce chapitre était une présentation et formulation du problème de sélection des services web, avec quelques travaux de résolution et un exemple de motivation. Une partie état de l'art consacré aux 4 catégories d'approches de sélection de services web (Exacte, heuristique, méta-heuristique, technologie de base de données), avec quelques travaux faits dans le domaine et qui sont toujours encours d'amélioration.

Dans le chapitre suivant, nous allons se concentrer sur un méta heuristique qui est la base de notre thème à savoir la recherche coucou (cuckoo search).

Chapitre 3

Sélection des services Web composites basée sur l'algorithme de cuckoo search

« We cannot solve our problemes with some thinking we used when we created them »

Albert Einstein

1. Introduction

La croissance continue des services web qui partagent les mêmes fonctionnalités et le fait qu'un seul service n'est jamais suffisant mettent l'utilisateur face au défi de choisir la meilleure composition des services web qui satisfait ses besoins. Ceci nécessite une conception efficace pour obtenir une solution optimale avec minimisation dans laps de temps et l'espace de recherche parcouru.

Etant donné que la difficulté des problèmes d'optimisation NP hard résident dans le fait qu'il n'y pas des formules pour les résoudre de façon exacte. En effet, il est impossible de parcourir et tester toutes les possibilités pour trouver la meilleure solution et le nombre de possibilités sera augmenté exponentiellement juxtaposé avec la taille du problème, de cela les méthodes classiques sont très limitées pour résoudre de tels problèmes notamment notre problème de sélection des services web composite (**SWC**) car un grand ensemble de solutions doit être exploré dans une durée de temps très longue.

De surcroit les recherches montrent que les méthodes bios inspirées (les méta-heuristiques) avec leurs déférentes stratégies et paramètres peuvent résoudre ce genre de problème avec un succès réel, l'une de ces méthodes récentes est la recherche de coucou.

A la lumière de ce qui précède dans ce chapitre nous adaptons l'algorithme de coucou pour résoudre le problème de sélection de la solution quasi optimale dans une composition du service Web, ainsi nous décrivons un prototype implémentant l'environnement de sélection des services web composites ***« Sélection des services web basée sur les Méta-heuristiques (Cuckoosearch) »*** ensuite nous montrons les différentes expérimentations menées, et finalement nous discutons les résultats obtenus.

2. Scénario

La composition des services web consiste à construire des nouveaux services en intégrant un ensemble des services existants. En raison de la croissance rapide des fournisseurs de services web sur internet il devient difficile de choisir un service approprié qui satisfait les exigences non fonctionnels des utilisateurs lors de la composition des services web, de ce fait la sélection des services web composites sera basée sur les qualités des services telle que : le temps de réponse, la fiabilité , l'intégrité, la confidentialité, etc. Afin de distinguer entre les services qui offrent des fonctionnalités similaires.

Nous concéderons « *un service de prise de commande en ligne* » ,lors d'une prise de commande il faut tout d'abord que le client soit connecté à l'internet, ensuite, il visite certaines boutiques et consulte leurs fiches techniques. Afin d'avoir une liste de souhait il va faire une commande où il peut ajouter les articles qu'il envisage d'acheter au panier. Puis il choisit un mode de payement (par carte bancaire, par paypale, par chèque …),après la validation de sa commande il effectué le service de la livraison en choisissant les options qui convient à ses exigences.

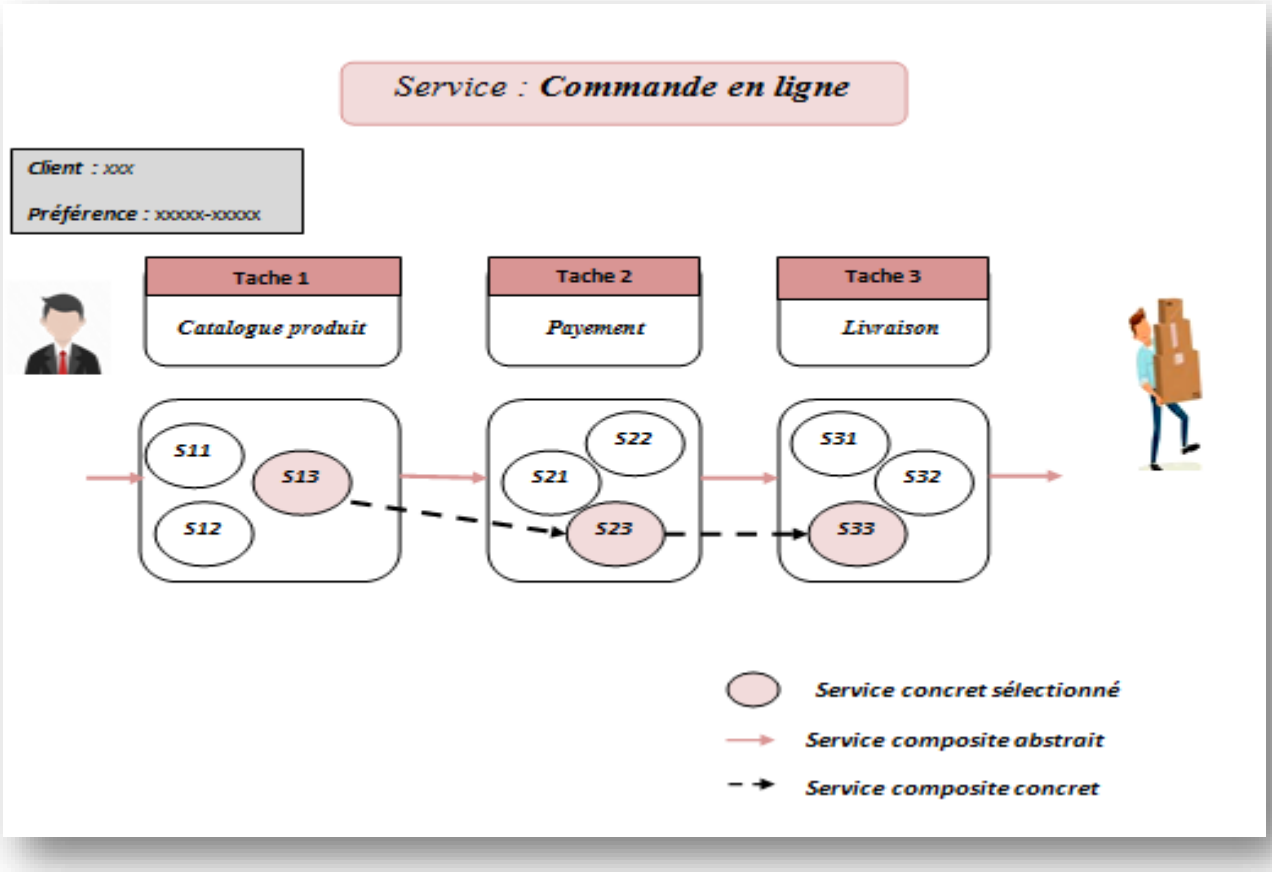


Figure 3.1: Exemple illustratif de la sélection de service web composite.

Dans exemple (**Figure 3.1**) le client va effectuer une commande en ligne. Effectivement, il ne peut pas consulter les produits avant de les acheter ni rembourser son argents. Après de ce fait, il faut renseigner la fiabilité (***QoS1***) du service sur lequel il souhaite passer une commande d'un part, et qu'il soit attentif à la mention de la disponibilité (***Qos2***) d'autre part.

3. Les préliminaires du problème de QoSWSC

3.1. Les propriétés QoS d'une composition

Les propriétés fonctionnelles du service décrivent ce que le service peut faire et les propriétés non fonctionnelles décrivent comment le service peut le faire, ces derniers sont les critères les plus importants dans la sélection où ils aident l'utilisateur à choisir le service le plus adéquat dans une composition. Généralement, les propriétés non fonctionnelles du service Web comprennent la performance, la réutilisation, la maintenance, la sécurité, la fiabilité, disponibilité, etc. et ces derniers peuvent être évalués d'une manière qualitative ainsi d'une manière quantitative, souvent les attributs quantitatifs sont les plus utilisés.
Dans la section ci-dessous, nous décrivons brièvement divers paramètres QoS:

1. **Fiabilité:** Ratio du nombre des opérations réussies par l'utilisateur pour le nombre total d'invocations.

$$\boldsymbol{Fiability} = \frac{\boldsymbol{Ns}}{\boldsymbol{Ni}} \ (\%)$$

Où :

Ns : Le nombre d'invocations réussies.

Ni : Le nombre total d'invocations.

2. **Disponibilité:** la disponibilité correspond au Le pourcentage d'accessibilité du service, Elle est mesuré en pourcentage (%).
3. **Coût:** Montant d'argent que l'utilisateur doit payer pour consommer un service Web, Le coût peut être permanent pour chaque invocation ou proportionnel à chaque méthode de service Web qui est utilisée. Les fournisseurs peuvent également demander des frais plus élevés pour les services hébergés sur un matériel meilleur / plus rapide.
4. **Temps de réponse :** c'est la période entre la transmission et la réception d'une requête.

$$\Delta_{\boldsymbol{T}} = \ \boldsymbol{Tr} - \boldsymbol{Ts} \ (\boldsymbol{ms})$$

Où :

$\Delta_{\boldsymbol{T}}$: Le temps de réponse.

Tr : Le temps de réception de la réponse par le client

Ts : Le temps d'envoi de la requête par le client.

5. **Réputation :** La réputation reflète une perception commune d'autres services web ou client envers ce service. Il regroupe les notes du service donné par d'autres principes. Généralement, une réputation serait établie à partir les feedbacks des clients.

Ici, nous avons identifié ces différents paramètres QoS dans notre méthode car ces paramètres ont un grand impact sur l'assistance aux demandeurs pour des sélections raisonnables. En outre, ces paramètres sont les qualités fondamentales pour remplir les objectifs du service Web. Ceux-ci peuvent être divisés en deux groupes :

- ✓ **Le premier groupe** des valeurs d'attribut QoS négatifs ont besoin être minimisé, c'est-à-dire plus la valeur du temps de réponse et le coût inférieur seront la qualité.
- ✓ **Le deuxième groupe** des attributs QoS positifs qui ont besoin être maximisé tel que disponibilité, fiabilité.

Pour résoudre un problème d'objectif de maximisation, il faut multiplier les valeurs négatives par (-1) pour faire face à l'hétérogénéité des critères de QoS.

3.2. Les fonctions d'agrégation des propriétés QoS

Après avoir testé et évalué les propriétés individuelles, elles doivent être combinées pour obtenir la qualité composite agrégée pour le service Web. À cette fin, Le tableau qui suit montre divers méthodes d'agrégation appropriées pour calculer les qualités composite du service.

QoS	séquentiel	Parallèle	Boucle	Conditionnel
Temps de réponse	$\sum_{j=1}^{n} q(Sj)$	$max_{j=1\ q}^{n} q(Sj)$	$\sum_{j=1}^{n} q(si)$	$max_{j=1\ q}^{n} q(Sj)$
Cout	$\sum_{j=1}^{n} q(Sj)$	$\sum_{j=1}^{n} q(si)$	$\sum_{j=1}^{n} q(si)$	$max_{j=1\ q}^{n} q(Sj)$
Disponibilité	$\prod_{j=1}^{n} q(Sj)$	$\prod_{j=1}^{n} q(Sj)$	$\prod_{j=1}^{n} q(Sj)$	$min_{j=1\ q}^{n} q(Sj)$
Fiabilité	$min_{j=1\ q}^{n} q(Sj)$	$min_{j=1\ q}^{n} q(Sj)$	q(Sj)	$min_{j=1\ q}^{n} q(Sj)$

Tableau 3.1 : Les fonctions d'agrégation des propriétés QoS. (Alrifai, 2010)

Dans notre travail, nous avons utilisé le modèle séquentiel car une séquence exprime une composition d'un workflow abstrait, un service s'exécute lorsque le précédent termine son exécution.

3.3. La fonction objective

La fonction objective U (c) est définie pour cartographier les valeurs des attributs QoS en un nombre réel pour faciliter le classement et la sélection des services Web en tenant compte les contraintes globales et les besoins d'utilisateur. Cette fonction est définie comme suit:

$$\mathbf{U}(\mathbf{c}) = \sum_{\mathbf{i=1}}^{\mathbf{d}} \mathbf{w_i} \frac{\mathbf{Qi(CSW)} - \mathbf{Qi_{min}}}{\mathbf{Qi_{max}} - \mathbf{Qi_{min}}} - \mathbf{P}(\mathbf{c})$$

Tel que $\mathbf{w_i} \in \mathbf{R_0^+}$ et $\sum_{i=1}^{d} \boldsymbol{w_i} = \mathbf{1}$, dont $\boldsymbol{d}$ c'est le nombre des propriétés QoS dans notre cas $\boldsymbol{d}$ **=5** et $\boldsymbol{w_i}$représente le poids qui exprime l'importance de **Qi**, étant donné que le mécanisme de sélection devrait permettre aux utilisateurs de trier les services selon les préférences individuelles et de mettre en évidence la relation entre les différentes propriétés. Par exemple, un utilisateur donne la première priorité au temps de réponse et à la sécurité, puis coût. Donc, le poids aide l'utilisateur à classer le service en fonction de sa priorité. Le poids donné sera entre 0 et 1, nous considérons dans notre travail que l'utilisateur donne une importance égale pour les cinq critères alors les valeurs des poids seront égaux $\mathbf{w_i}$ =0.2. $\mathbf{Qi_{max}}$La valeur maximum du critère **Qi** et $\mathbf{Qi_{min}}$la valeur minimum du critère **Qi** pour les compositions des services web générées, qui peuvent être calculées à l'aide les fonctions d'agrégation définies dans le **Tableau 3.1 .**

$$\boldsymbol{P(c)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{d}} \boldsymbol{D_i}^2$$

Si $(\boldsymbol{Q_i} \geq \boldsymbol{C_i})\ alors \boldsymbol{D_i}$ **= 0**

Sinon $\boldsymbol{D_i} = (\boldsymbol{C_i} - \boldsymbol{Q_i})$

$\boldsymbol{C_i}$Présente les contraintes QoS globales qui sont exprimé la borne inferieure ou supérieur qui doit satisfaire les contraintes des QoS de la composition.

$\boldsymbol{P(c)}$C'est la fonction de pénalité qui estappliquéauxcontraintesà desproblèmes d'optimisation (maximisation ou minimisation) en introduisant une pénalité artificielle lorsqu'une ou plusieurs critères de la composition ne respectent pas les contraintes.

4. L'algorithme de Cuckoo search

4.1. Définition recherche Coucou (CS)

Recherche Coucou (CS) est un algorithme efficace basé sur l'intelligence par essaim, des développements importants ont été réalisés depuis son introduction en 2009.

L'Algorithme CS a était proposé et développé par Xin-She Yang et Suash (Yang, 2010), c'est une très récente méta-heuristique à base d'une population de solution, CS est inspiré du comportement de reproduction parasitaire de quelques espèces de coucous.

4.2. Comportement de coucou

Ce parasitisme de reproduction agressive agit comme suit :

La femelle coucou pond ses œufs dans le nid de d'autres oiseaux hôtes d'une façon aléatoire, et le reste dépond de l'hôte, s'il détecte l'œuf coucou dans son nid, dans ce cas soit iljette l'œuf ou bien il abandonne totalement son nid pour reconstruire un autre ailleurs.

L'algorithme est développé de la même façon.

La recherche sur le coucou idéalise un tel comportement de reproduction, et peut donc être appliqué pour différents problèmes d'optimisation.

4.3. Principe de Cuckoo search

L'algorithme de Cuckoo Search est basé sur trois règles idéalisées:

- Chaque coucou choisi un nid de manière aléatoire et pond dans ce dernier seulement un œuf à la fois.
- Les meilleurs nids avec des œufs (solutions) de haute qualité sont conservés pour les prochaines générations.
- Le nombre de nids hôtes est fixé. L'œuf pondu par un coucou peut être découvert par l'espèce hôte avec une probabilité Pa∈ [0,1]. Dans ce cas, l'oiseau hôte (i) soit quitte le nid et en construit un autre. (i) soit détruit l'œuf du coucou. Pour simplifier l'implémentation, cette dernière hypothèse peut être approchée par le remplacement d'une fraction Pa de n nids par des nouveaux.

4.4. Le vol de Lévy

Dans la nature, les animaux recherchent de la nourriture de manière aléatoire ou quasi-aléatoire Généralement, le chemin d'alimentation d'un animal est effectivement une marche aléatoire parce que le prochain déplacement est basé à la fois sur l'emplacement / l'état actuel et sur la probabilité de transition vers l'emplacement suivant. La direction choisie dépend implicitement d'une probabilité, qui peut être modélisée mathématiquement.

Un vol Lévy est une marche aléatoire dans laquelle les étapes sont définies en termes de longueur d'étape, qui a une certaine distribution de probabilité, les directions des étapes étant isotropes et aléatoires.

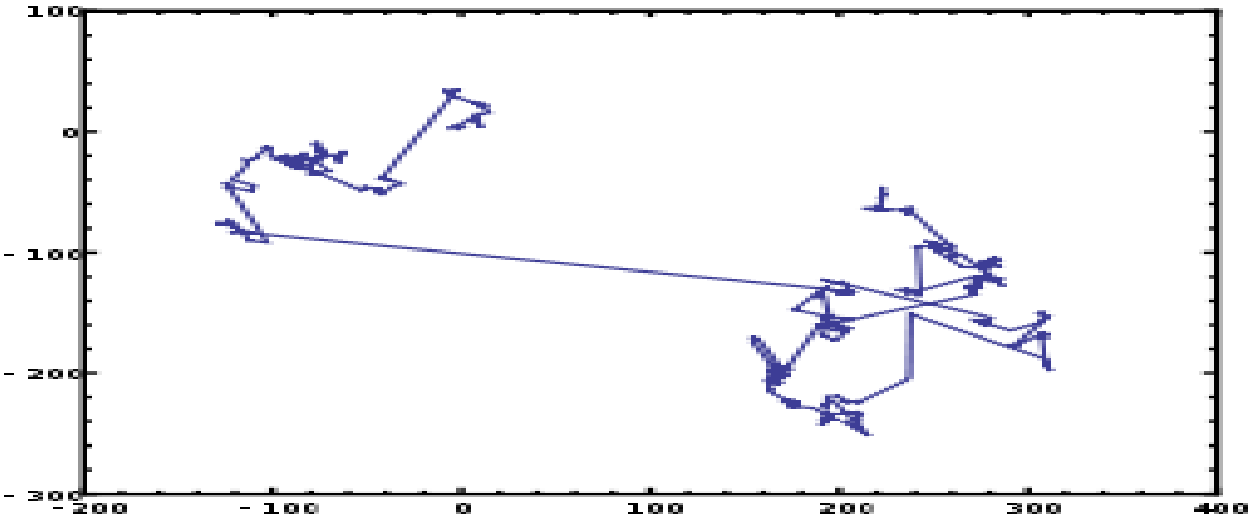


Figure 3.2 :Exemple de vol de Lévy.

5. Adaptation de la méta-heuristique Cuckoo search au problème QoSWSC

Dans notre cas (Sélection des Service Web Composés SSWC), chaque œuf dans un nid représente une composition et chaque coucou peut pondre un seul œuf (qui représente une composition), le but est d'utiliser la meilleure composition pour remplacer une moins bonne composition dans un nid. Bien que l'algorithme puisse être étendu au cas le plus complexe où chaque nid contient plusieurs œufs représentant un ensemble de compositions (solutions), nous employons ici la version la plus simple où chaque nid ne contient qu'un seul œuf (une seule composition). Dans ce cas, il n'y a plus de distinction entre œuf, nid ou coucou, et chaque nid correspond à un œuf qui représente aussi un coucou qui est modélisé par une composition.

Nous proposons ici une version simplifiée de l'algorithme de CS

Algorithme CS_SSWC : Recherche Coucou pour la Sélection des Service web Composés

Entrées :

Un workflow : l x m // *l* : le nombre de classes de services, m : le nombre de services par classe.

N : le nombre de nids

MaxIt : le nombre d'itérations

Pa : la probabilité le nombre de nid à remplacer

Cg : contrainte globales

Début

It = 1 ;

1. Générer la population initiale : créer aléatoirement N nids, et N coucou // c à d nous avons N compositions ;
2. Evaluer chaque Coucou par la fonction U () ;

Tant que (it<MaxIt) faire

Pour chaque coucou Ci (i= 1..N) **faire**

3. Créer un œuf Pi en utilisant un pas aléatoire ; // création d'une nouvelle composition
4. Calculer la fitness de Pi ;
5. Remplacer Ci par Pi si U (Pi) > U (Ci);

Fin Pour

6. Une fraction Pa des plus mauvais nids est abandonné et d'autres sont générés à la place ;
7. Trouver la meilleure composition Ci. Si elle est meilleure que celle de l'itération précédente la remplacer ;

It=It+1 ;

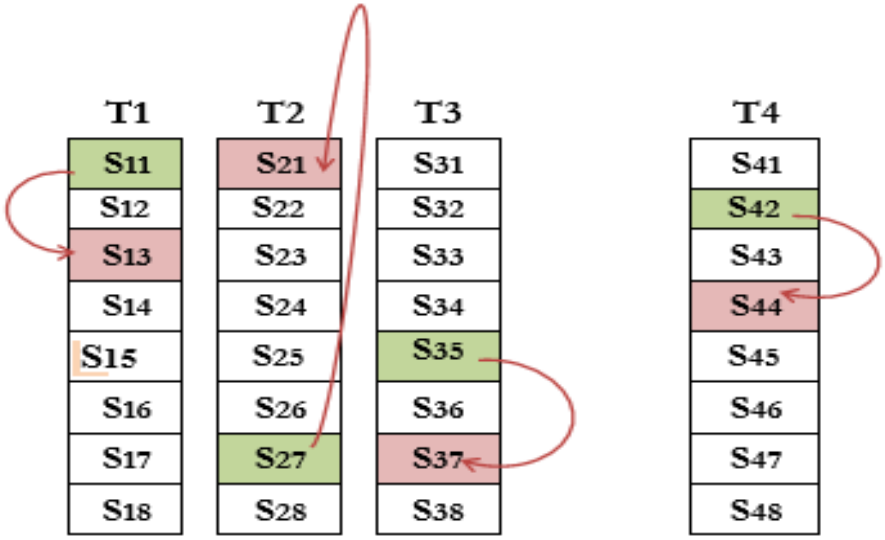
Fin Tant que

Fin.

Nous n'utilisons pas dans notre algorithme le vol de Lévy. Le vol de Lévy permet de créer un déplacement aléatoire. Ce dernier, n'est pas adapté à notre problème d'optimisation nous le remplaçons par une génération d'un pas aléatoire (ligne 3 de l'algorithme) et d'un déplacement comme celui utilisé dans SOMA (Halfaoui, 2017), comme suit :

Nous générons un pas aléatoire ente [1 et le nombre de service par classe], prenons par exemple la composition C1.Sachant que le nombre de classe est 4, le nombre de services par classe est égale à 8 et que le pas aléatoire généré est 2, nous aurons le poussin P1, S11 (1+2) sera remplacé par S13 S27 par S21,….

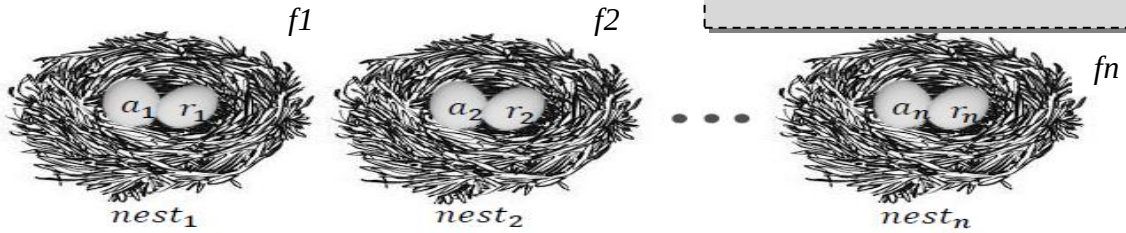
C1	S11	S27	S35	S42
P1	S13	S21	S37	S44



6. Exemple explicatif

ai : la classe de service.
ri : le service.
fn : Fitness.

Etape 1 : créer aléatoirement n nids (nest n).

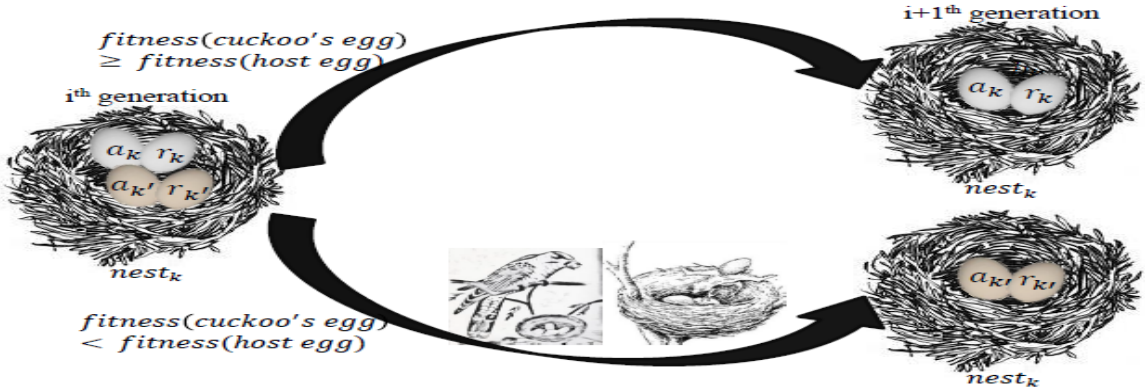


Etape 2 : Créer un œuf $a_{k'}r_{k'}$ en utilisant un pas aléatoire.

$$a_{k'}r_{k'}=a_k r_k+randomwalk$$

Randomwalk : pas aléatoire

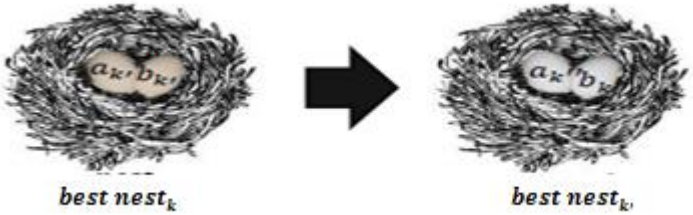
Etape 3 : Calculer la fitness de $a_{k'}r_{k'}$ et comparer entre$f(a_k r_k)$
Et $f(a_{k'}r_{k'})$ si$f(a_{k'}r_{k'}) \geq f(a_k r_k)$remplacer $a_k r_k$par $a_{k'}r_{k'}$.



Etape 4 : avec une fraction Pa éliminer les plus mauvais nids et générer de nouveaux nids à la place

.

Etape 5 : Trouver la meilleure composition *bestnest*, si elle est meilleure que celle de l'itération précédente la remplacer.



Répéter les étapes **2 à 5** jusqu'à ce que le critère d'arrêt soit satisfait.

Implémentation & Expérimentations

1. Introduction

dans ce qui suit nous présentons et décrivons notre prototype implémentant l'environnement de sélection d'une composition de services web, commençant par la description des différents composants de notre système avec leurs rôles, passants au corpus utilisés pour évaluer l'environnement , et terminant avec les expérimentations appliquées sur le système en discutant chaque résultat obtenu.

2. Présentation de l'environnement de développement

Nous avons implémenté et développé notre prototype sous Netbeans IDE 8.2 avec java (JDK 1.8.0).

Nous avons utilisé java comme langue de programmation pour notre problème.

Netbeans : un environnement de développement intégré (EDI) *open source* développé par Sun en juin 2000 sous licence CDDL (Common Developpent and Distribution License).

3. Outils de système et paramètres (Hadjila, 2012)

Les paramètres d'entrée sont présentés dans L'interface de notre prototype montré dans la figure suivante :

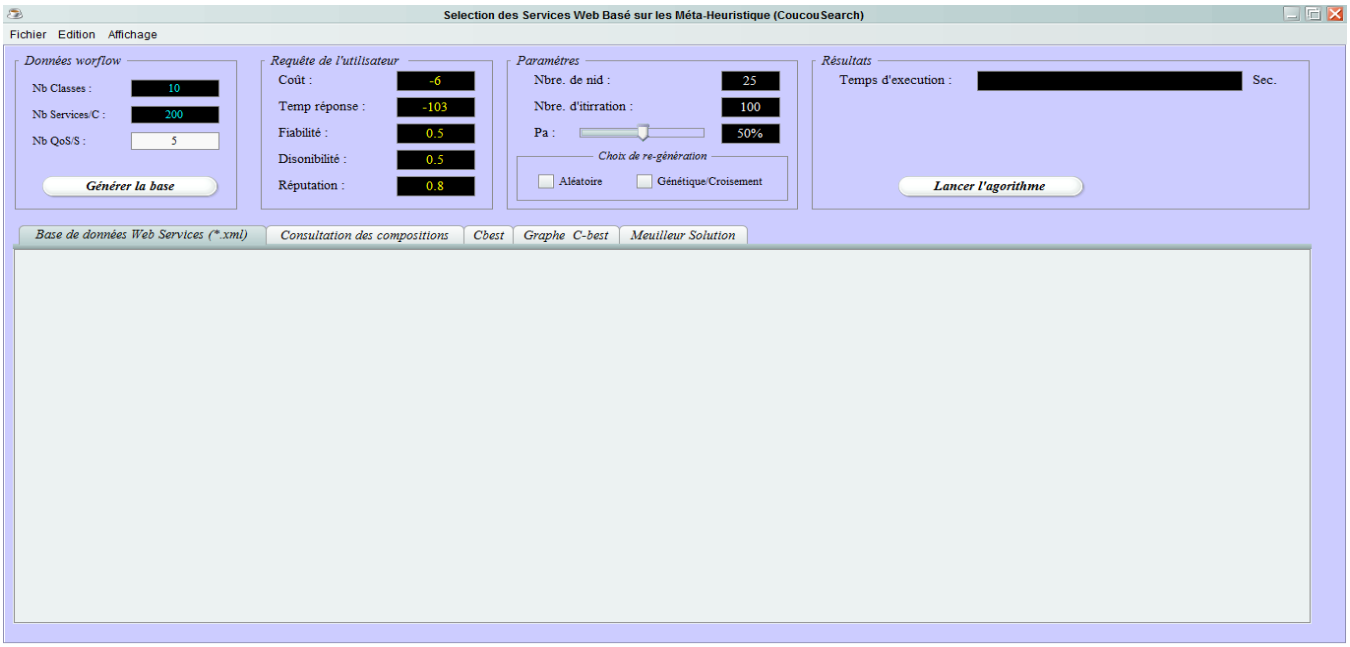


Figure 3.3 : Interface principale.

4. Interface (paramètre et résultat)

Nous avons construit notre interface d'une manière simple afin de facilité la manipulation des paramètres, et que les résultats soient compréhensible.

4.1 Générer et charger la base

Nos tests sont effectués sur une base de données synthétique dans un format XML

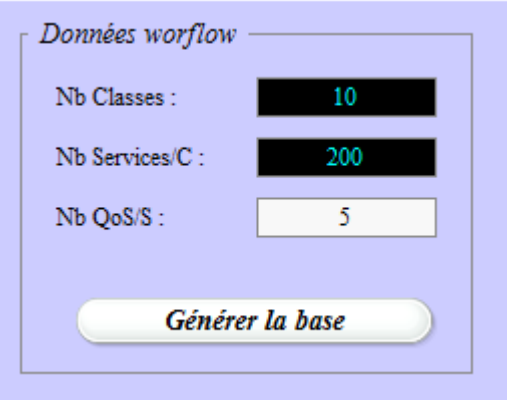


Figure 3.4: Données workflow.

A partir de cette dernière nous pouvons charger une base aléatoirement avec un nombre de classes et de services diminué selon nos désirs.

Nous chargerons la base que nous avons générer pour effectuer les tests voulu, et l'affichage sera comme suit :

Classes	Sévices	Qos
Classe 1	Service 1	les qualités de services
	Service 2	
	Service m	
Classe 2		
- - Classe n		

4.2.Requête de l'utilisateur et composition optimale

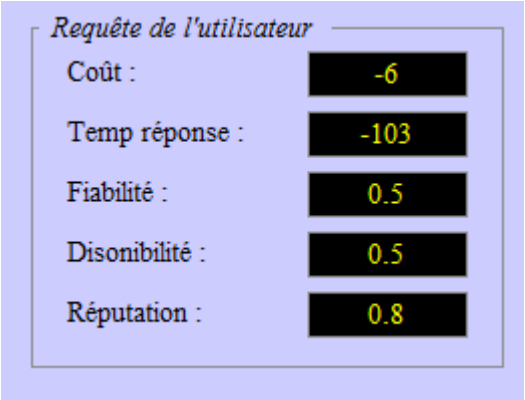
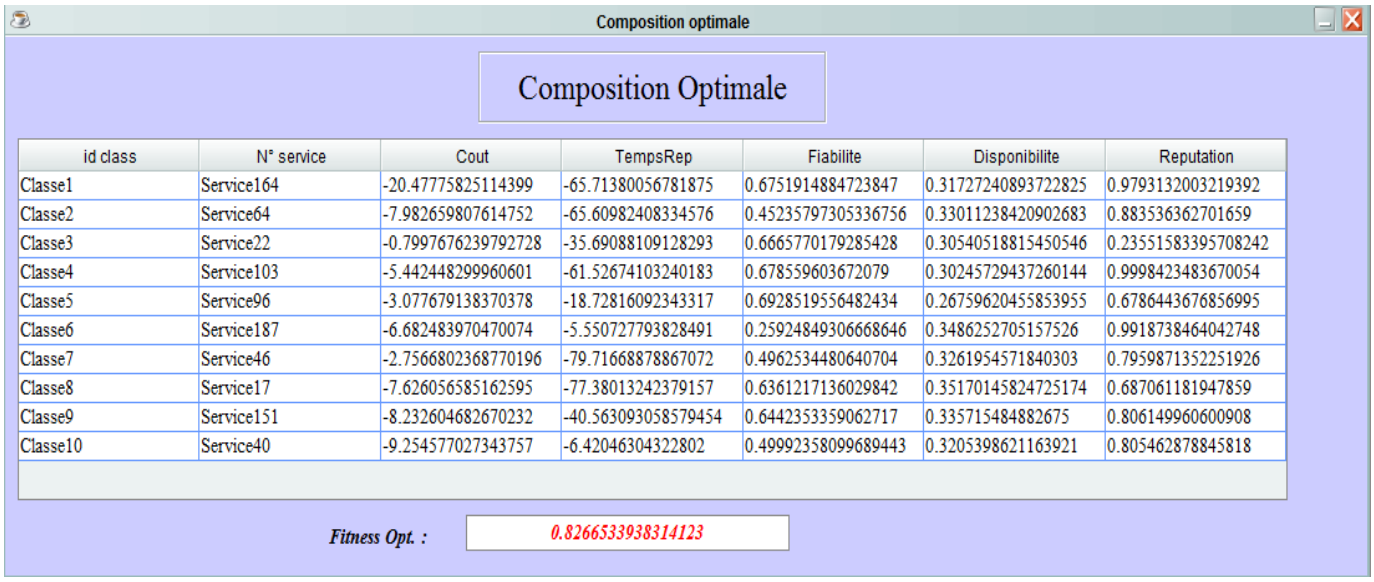


Figure 3.5: Les critères globaux des qualités de services.

L'intervalle de chaque qualité est montré dans le tableau suivant :

Afin de comparer la solution quasi optimale atteinte par l'exécution de la méthode coucou avec la meilleure solution existante dans la base générée, Nous calculons la composition Optimale de note base.



id class	N° service	Cout	TempsRep	Fiabilite	Disponibilite	Reputation
Classe1	Service164	-20.47775825114399	-65.71380056781875	0.6751914884723847	0.31727240893722825	0.9793132003219392
Classe2	Service64	-7.982659807614752	-65.60982408334576	0.45235797305336756	0.33011238420902683	0.883536362701659
Classe3	Service22	-0.7997676239792728	-35.69088109128293	0.6665770179285428	0.30540518815450546	0.23551583395708242
Classe4	Service103	-5.442448299960601	-61.52674103240183	0.678559603672079	0.30245729437260144	0.9998423483670054
Classe5	Service96	-3.077679138370378	-18.72816092343317	0.6928519556482434	0.26759620455853955	0.6786443676856995
Classe6	Service187	-6.682483970470074	-5.550727793828491	0.25924849306668646	0.3486252705157526	0.9918738464042748
Classe7	Service46	-2.7566802368770196	-79.71668878867072	0.4962534480640704	0.3261954571840303	0.7959871352251926
Classe8	Service17	-7.626056585162595	-77.38013242379157	0.6361217136029842	0.35170145824725174	0.687061181947859
Classe9	Service151	-8.232604682670232	-40.563093058579454	0.6442353359062717	0.335715484882675	0.806149960600908
Classe10	Service40	-9.254577027343757	-6.42046304322802	0.49992358099689443	0.3205398621163921	0.805462878845818

Figure 3.6: Composition optimale généré à partir de la base.

4.3.Lancement d'algorithme et affichage

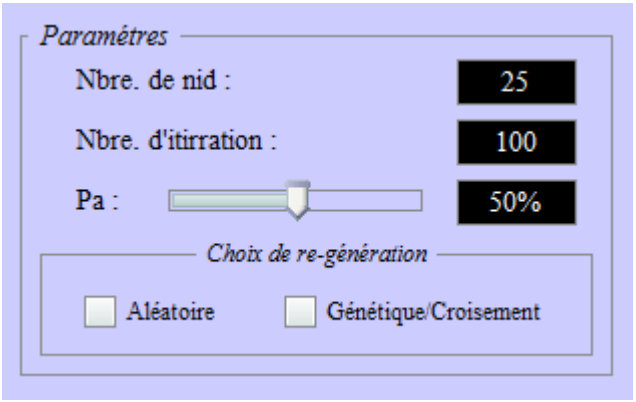
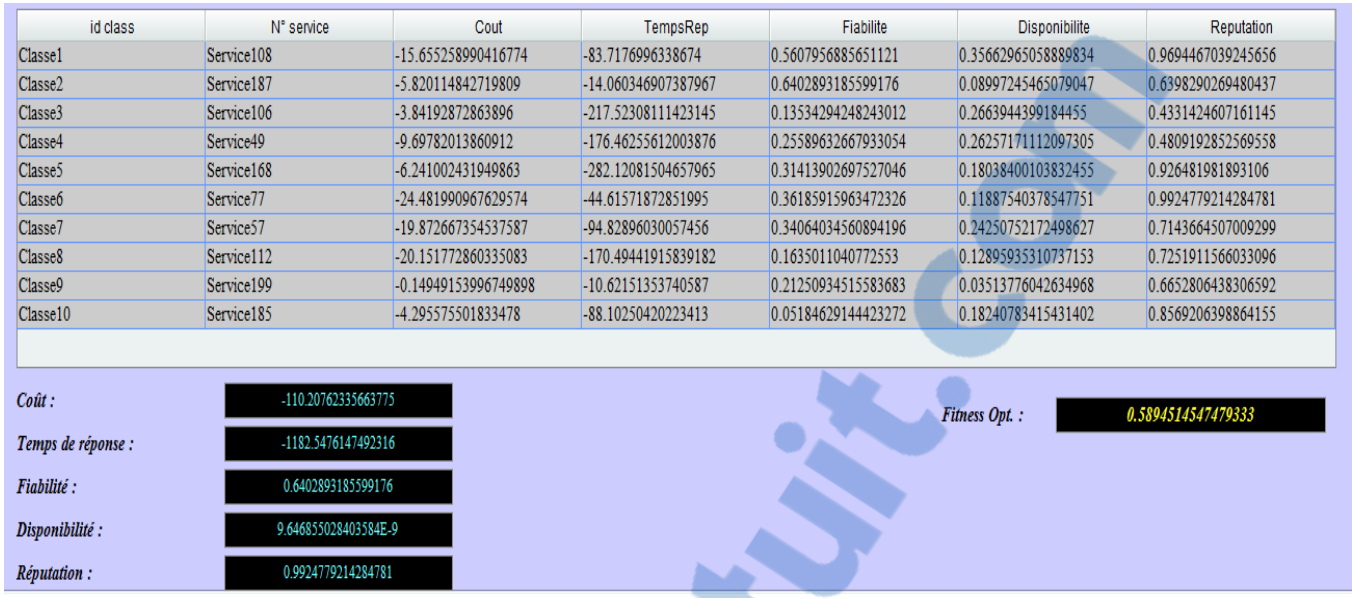


Figure 3.7: paramètre de l'algorithme coucou.

La figure ci-dessus représente les paramètres utilisés dans l'algorithme coucou qui ont un impact sur le résultat atteinte.



id class	N° service	Cout	TempsRep	Fiabilite	Disponibilite	Reputation
Classe1	Service108	-15.655258990416774	-83.7176996338674	0.5607956885651121	0.35662965058889834	0.9694467039245656
Classe2	Service187	-5.820114842719809	-14.060346907387967	0.6402893185599176	0.08997245465079047	0.6398290269480437
Classe3	Service106	-3.84192872863896	-217.52308111423145	0.13534294248243012	0.2663944399184455	0.4331424607161145
Classe4	Service49	-9.69782013860912	-176.46255612003876	0.25589632667933054	0.26257171112097305	0.4809192852569558
Classe5	Service168	-6.241002431949863	-282.12081504657965	0.31413902697527046	0.18038400103832455	0.926481981893106
Classe6	Service77	-24.481990967629574	-44.61571872851995	0.36185915963472326	0.11887540378547751	0.9924779214284781
Classe7	Service57	-19.872667354537587	-94.82896030057456	0.34064034560894196	0.24250752172498627	0.7143664507009299
Classe8	Service112	-20.151772860335083	-170.49441915839182	0.1635011040772553	0.12895935310737153	0.7251911566033096
Classe9	Service199	-0.149491539967498 98	-10.621513537405 87	0.21250934515583683	0.03513776042634968	0.6652806438306592
Classe10	Service185	-4.295575501833478	-88.10250420223413	0.05184629144423272	0.18240783415431402	0.8569206398864155

Coût : -110.20762335663775

Temps de réponse : -1182.5476147492316

Fiabilité : 0.6402893185599176

Disponibilité : 9.646855028403584E-9

Réputation : 0.9924779214284781

Fitness Opt. : 0.5894514547479333

Figure 3.8: la solution optimale selon l'algorithme de coucou.

5. Expérimentations

Nos expérimentations sont menées sur une machine dotée de :

Critère QoS	Intervalle
Coût	∈[0,30€]
Temps de repense	∈[0,300ms]
Fiabilité	∈[0,5-1,0]
Disponibilité	∈[0,5-1.0]
Réputation	∈[0,5]

Nous présentons les résultats obtenus après plusieurs simulations, nous nous concentrons sur l'optimalité et le temps d'exécution.

Les paramètres changeables selon les expérimentations sont :

Le Nombre d'itérations va de 50 à 400.

Le Nombre de nids va de 5 à 100.

Le choix de la régénération des compositions Aléatoire ou génétique (croisement).

La taille de la base : classes (3 à 10), services (50 à 200).

Dans le premiers teste nous calculons le taux d'optimalité par rapport au nombre d'itérations, et le temps d'exécution en parallèle sur la base totale (10 classes, 200 services) et avec un nombre de nids fixé à 25 les résultats sont présenté dans les graphes suivant :

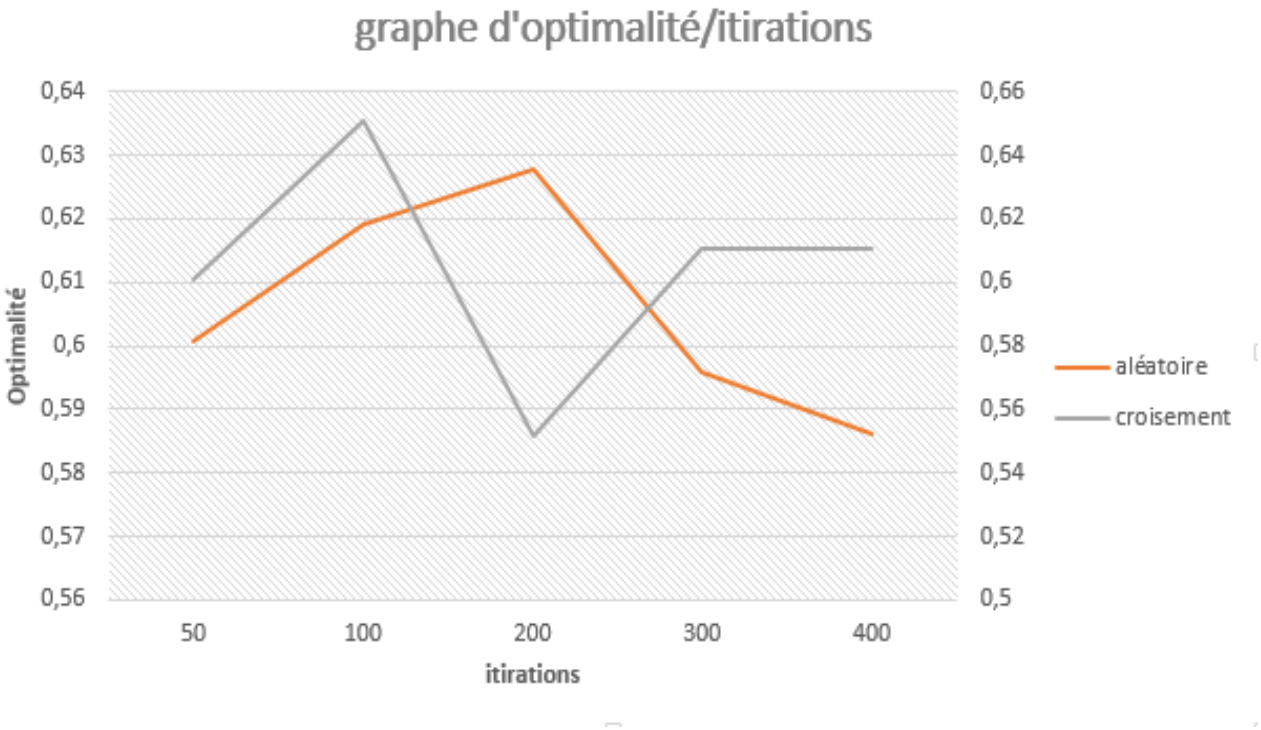


Figure 3.9: Courbe d'optimalité croisement vs Aléatoire/ itération.

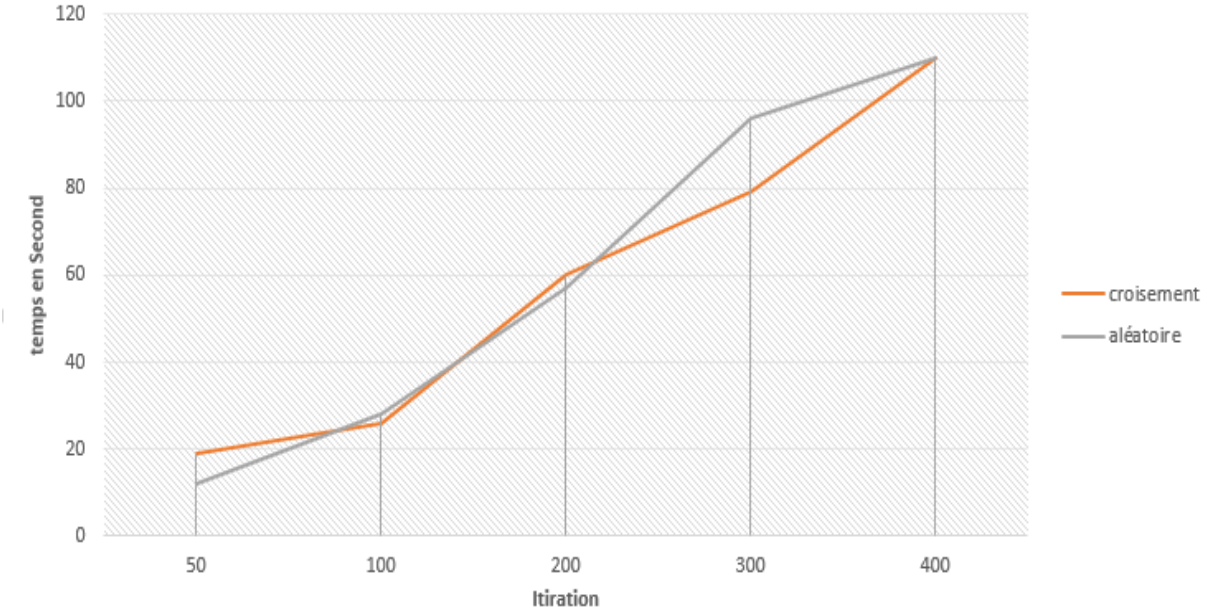


Figure 3.10: courbe de temps d'exécution Génétique Vs Aléatoire.

Nous pouvons remarquer, selon le choix du (re)génération la solution optimale est atteinte après 100 itérations pour la méthode croisement, et 200 itérations pour la méthode aléatoire.

Aussi, la régénération des compositions avec la méthode génétique donne plus meilleur résultat que l'aléatoire, il n'y a pas une grande différence au côté temps d'exécution, dans les deux choix le temps augmente proportionnellement avec le nombre d'itération.

Dans le deuxième teste nous changeons le nombre de nids pour un nombre d'itération fixé à 100, les résultats sont

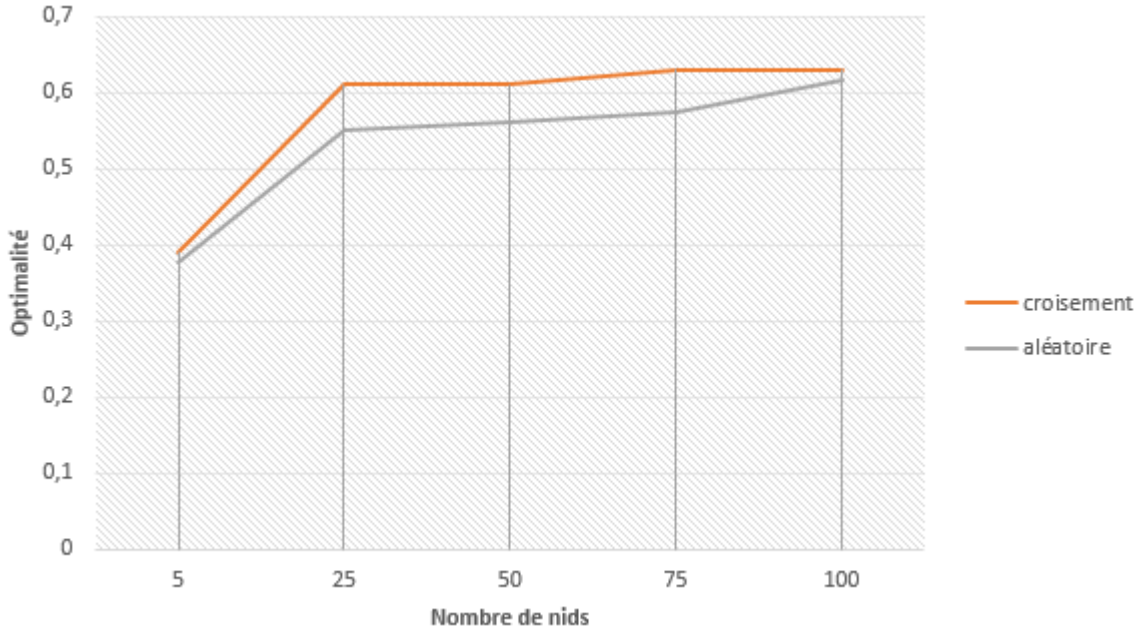


Figure 3.11: Courbe d'optimalité Croisement Vs Aléatoire /Nombre de nids.

En résumé, les changements menées sur le nombre de nids à beaucoup plus d'impact quand le choix de la régénération est avec la méthode génétique, où l'optimalité est réalisé pour un nombre de nids (population) minimum 5 nids ou maximum 100 nids.

Un autre teste reste a effectué sur le Pa c'est le pourcentage de changement sur la population dans chaque itération est qui a résulter la courbe suivante

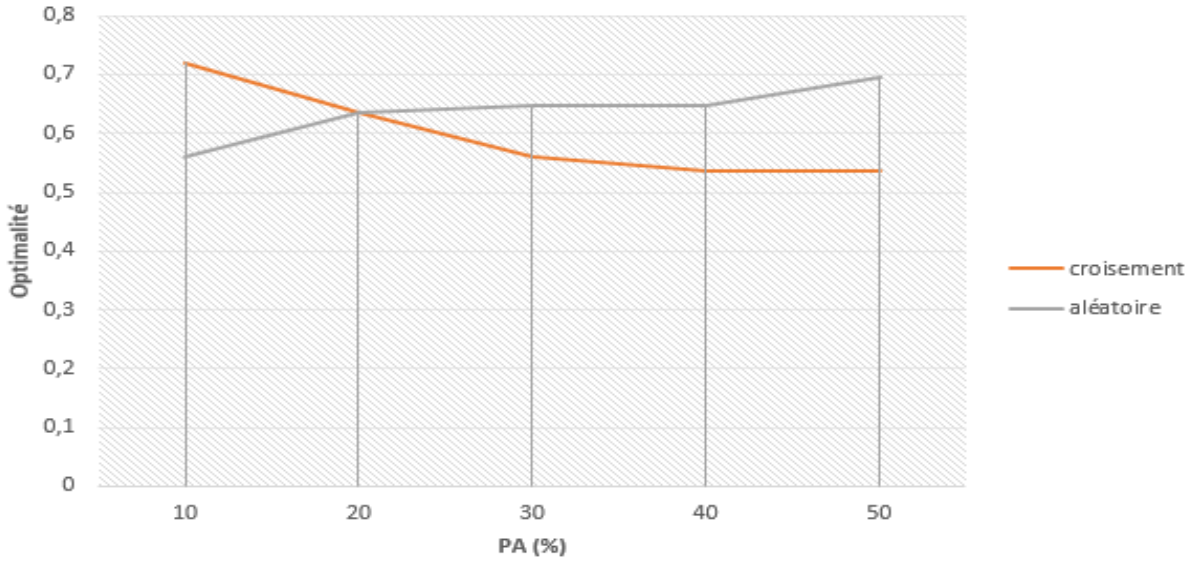


Figure 3.12 : Courbe d'optimalité sur le Pa (%).

Les résultats démontrent une interversion entre les méthodes de régénération, ou nous pouvons clairement voire qu'avec la méthode de croisement si le Pa augmente l'optimisation diminue, contrairement à la régénération aléatoire l'optimisation est meilleur quand le pa est grands (50%).

Selon les résultats obtenus nous pouvons remarqués que la résolution avec Cuckoo est prometteuse.

	Coût	Temps de réponse	Fiabilité	Disponibilité	Réputation	C-best
SW1	-38.476	-258.735	0.676	0.003	0.855	0.644
SW2	-52.191	-643.120	0.662	1.257	0.680	0.544
SW3	-84.872	-718.784	0.611	1.469	0.969	0.531
SW4	-138.021	-1374.99	0.665	2.236	0.977	0.53

Tableau 3.2: Résultats d'optimalité.

Le tableau ci-dessus démontre des résultats obtenus après des modifications sur la taille de la base utilisé pour mener les tests où :

SW1 : une base de 3Classes/50 services.

SW2 : une base de 5Classes/100 services.

SW3 : une base de 7Classes/150services.

SW4 : une base de 10Classes/200services.

Nous conclurons que la taille de la base joue un rôle important dans l'obtention d'une meilleur optimalité, nous remarquons que la fitness de meilleure composition augmente relativement avec la réputation et la fiabilité.

6. Conclusion

Ce chapitre est consacrée à formaliser le **QOSWSC**, en premier section nous définissons les préliminaires de ce problème, en mettant l'accent sur la fonction objectif afin d'évaluer les meilleures résultats à l'aide de la méthode Méta-heuristique recherche de coucou, qui est présentée en détail à partir de la définition (comportement, pseudo, paramètre) La deuxième section nous a permis de tester la capacité de résolution d'un problème de sélection avec l'algorithme de coucou, en se concentrant sur l'optimalité atteinte par ce dernier, après l'implémentation de coucou, nous menons quelque itérations.

Enfin les résultats obtenus par l'exécution de coucou monteront une capacité satisfaisant en terme de temps d'exécution, ce qui est un point important dans l'environnement des SW, outre il répond une manière plus efficace en tenant compte les besoins des utilisateurs d'une part et les contraintes globales de QoS d'autre part, notre approche garantit une optimalité relativement bonne cette dernière sera testée dans le future avec d'autres algorithme.

Conclusion Générale

Conclusion générale

La Sélection de la composition contenant l'ensemble des services web qui satisferont les besoins d'utilisateur et qui ont une meilleure qualité, est l'objective de notre travail.

Pour cela nous avons commencé par présenter notre problématique qui été détaillée d'un chapitre à l'autre jusqu'à l'obtention d'une solution, dans le premier chapitre nous avons présenté les services web et les compositions générées à partir de ces derniers.

Puis nous sommes passées à la sélection dans le deuxième chapitre avec un exemple explicatif où nous avons déduit que la qualité de services est un point principal dans la sélection où la plus part des méthodes de sélection s'effectue sur la QoS, et nous avons finale chapitre par un état de l'art sur les méta-heuristiques qui sont utilisées dans la résolution de notre problème qui est classé comme un problème complexe ou bien NP-hard, nous avons aussi présenté quelques approches et travaux qui ont utilisés ces méthodes dans la résolution.

Dans le dernier chapitre, nous avons détaillé la méta-heuristique« la recherche de coucou », et nous l'avons adaptée à notre problème afin d'atteindre les meilleurs résultats possibles. Nous avons aussi présenté la fonction objective expliquée à travers un exemple. Nous avons terminé notre travaille avec l'implémentation de notre prototype et quelques expérimentations.

Nous avons discuté les résultats afin de montrer l'efficacité de notre méthode dans la résolution de ce genre de problème, en matière d'optimalité et de temps.

Les résultats obtenues montrent que l'algorithme proposé arrive à trouver des solutions de bonnes qualité mais nous prévoyons dans les travaux futures de :

- Améliorer l'algorithme Coucou en ajoutant une heuristique dans la génération de la population initiale afin de choisir une population initiale de meilleure qualité.
- d'intégrer et d'adapter le vol de Lévy ou une autre technique pour la génération d'un nouveau coucou, car pour l'instant nous le générons à travers un pas aléatoire.

Bibliographie

(s.d.). Récupéré sur www.openclassrooms.com/courses/les-service-web

Alrifai, M. S. (2010). *Selecting skyline.*

B. Benatallah, R. D. (2005). *Book chapter in: Service-Oriented Software Engineering: Challenges and Practices.* Idea Group Inc (IGI).

Clement, A. H. (2004). *UDDI v.3.0.2. OASISSpecification.*

D. Roman, U. K. (2005). *"Web Service Modeling Ontology,".*

F. Casati, a. M.-C. (2002). *Event-Based Interaction Management for Composite.* Information Systems Frontiers.

Feng, L.-i. O. (2013). Qos optimization for web services composition based on reinforcement. *International Journal of Innovative Computing, Information* .

FethAllah, H. (2014). *Composition et interopération des services web sémantiques.* Tlemcen.

Gardarin, G. (28 Nov 2002). *Xml, des bases de données aux services web (in french).* Dunod.

Hadjila, F. a. (2012). Qos-aware service selection . Alger. Algerie.

Halfaoui, A. H. (2017). Qos-aware webservice selection based on self-organising migrating algorithm and fuzzy dominance. *-aware webservice selection based on self-organising migrating algorithm and fuzzy dominance.*

Halfoui. (2016-2017). *La sélection des services web dans.* Tlemcen.

Jaeger, M. a. (2006). In proceedings of the 8th IEEE International Conference on E Commerce Technology and the 3rd IEEE International Conference on Enterprise Computing, E-Commercer,E-Services. San francisco.

KHANOUCHE, M. E. (2016 – 2017). Sélection et composition de services sensibles à l'énergie et à la qualité. BEJAIA.

Klein, A. I. (2011). Efficient heuristic approach with improved time complexity for qos-aware service composition. *IEEE International Conference*, (pp. pages 436–443).

Ko, J. M.-H. (2008). Quality-of-service oriented web service composition algorithm and planning architecture. *jornal of Systems and Software* .

la toupie. (s.d.). Récupéré sur http://www.toupie.org/Dictionnaire/Heuristique.htm

Lecue, .. M. (2009). Towards scalability of quality driven semantic web service composition. *International Conference*, (pp. pages 469–476).

Li, Y.-Q. a. (2012). an approach of qos-guaranteed web service composition based on a win-win strategy. *19th International Conference on.*

Liu, M. M. (2012). *A quality of service (qos)-aware execution plan selection approach for a service composition.*

M.Chalbabi. (novembre 2006). *Découverte de Services Web Sémantiques* .

Merzoug, M. C. (2014). Qos-aware web service selection based on harmony search. *ISKO-Maghreb* , pages 1–6. IEEE.

Ming, C. a.-w. (2007). An approach for web services composition based on qos and discrete particle swarm optimization. *Eighth ACIS International Conference on Software Engineering, Artificial Intelligence.*

Moustafa, A. a. (2013). Multi-objective service composition using reinforcement learning. *In International Conference on Service-Oriented Computing.*

Rodriguez-Mier, P. M. (2010). *Composition of web services through genetic programming.*

Rosenberg, F. M. (2010). Metaheuristic optimization of larges-cale qos-aware service compositions. *Services Computing* , (pp. pages 97–104.).

Wang, J. a. (2008). Optimal web service selection. *based on multi-objective genetic algorithm.*, (pp. 553–556 IEEE).
Xiang, X. (Apr. 2007). *Service-oriented architecture for integration of bioinformatic data and application.* Indiana: Graduate Program in Computer Science and Engineering.
Yang, X.-S. (2010). *Nature-Insprid Meta-heuristic Algoritms.* United Kindgdom: Luniver Press.
Yu, T. a. (2004). Service Selection Algorithms for Web Services with Endto-end QoS Constraints. . *In Proceedings of the 2004 IEEE Conference on Electronic Commerce (CEC04).* San Diego,California.
Yu, T. a.-J. (2005a). *Service selection algorithms for composing complex services with multiple qos constraints.* . International Conference.
Yu, T. a.-J. (2005b). *Service selection algorithms forweb services with end-to-end qos constraints.* Information systems and e-business.
Zeng, L. B. (2004). *QoS-Aware Middleware for Web Services Composition.* . IEEE transactions.
Zeng, L. B. (2003). Quality Driven Web Services Composition. *In proceedings of the 12th International World Wide Web Conference (WWW2003).* Budapest, Hungar.

Résumé

Entre un énorme nombre de services web offerts sur Internet qui ont la même fonctionnalité, et le fait qu'un seul service web ne peut jamais satisfaire les exigences de l'utilisateur, il est donc nécessaire de choisir une composition de services qui répondent au mieux aux besoins exigés, afin d'atteindre cet objectif des méthodes de sélection sont développés et améliorées pour faciliter le travail de l'utilisateur et minimiser le temps de sa recherche.

Ces méthodes avec leurs différentes stratégies souvent utilisent le critère de qualité qui distingue chaque service pour la sélection, cette qualité de service (QoS) nous aide à déterminer les meilleurs services parmi d'autres.

Nous nous intéressons, dans notre travail à la sélection des services web basée sur la qualité de service d'un côté, et d'un autre côté aux méthodes de résolutions de problèmes d'optimisation qui est le cas de notre problème de sélection connu comme un problème NP-hard.

La méthode de résolution proposée est une Méta-heuristique récente nommée la recherche de coucou (cuckoo search), nous adapterons cette méthode sur notre problème avec les améliorations nécessaires pour atteindre la solution optimal.

Mots clés : Sélection des services web, qualité de service, méta-heuristique, recherche coucou, optimisation combinatoire.

Abstract

a huge number of web services offered on the Internet that have the same functionality, and the fact that a single web service can never satisfy the user's requirements, it is therefore necessary to choose a service composition that best Needs in order to achieve this objective selection methods are developed and improved to facilitate the user's work and minimize the time of his research.

These methods with their different strategies often use the quality criterion that distinguishes each service for the selection, this quality of service (QoS) helps us to determine the best services among others.

In our work we are interested in the selection of web services based on quality of service on the one hand and, on the other hand, on the methods of solving optimization problems which is the case of our selection problem known as A NP-hard problem.

The proposed resolution method is a recent Meta-heuristic called cuckoo search, we will adapt this method on our problem with the improvements needed to achieve the optimal solution.

Keywords: Selection of web services, quality of service, meta-heuristics, cuckoo search, combinatorial optimization

ملخص

إن عدد كبير من خدمات الويب المعروضة على شبكة الإنترنت التي لها نفس الوظيفة، ونظرا أن خدمة ويب واحدة لا يمكنها تلبية متطلبات المستخدم بعض الأحيان ، ولذلك فمن الضروري اختيار خدمة الويب المركبة التي تلبي الاحتياجات ف أتمم وجه ، من أجل تحقيق هذا الاختيار تم تطوير الطرق وتحسينها لتسهيل عمل المستخدم وتقليل الوقت من أبحاثه.

هذه الأساليب مع استراتيجياتها المختلفة وغالبا ما تستخدم معيار الجودة التي تميز كل خدمة وجودة الخدمة مما (QoS)يساعدنا على تحديد أفضل الخدمات وغيرها.

في عملنا هذا أعرنا الاهتمام على اختيار خدمات الويب استنادا إلى نوعية من الجانب الخدمي وعلى الطرق حل المشاكل بطرق مثلى من ناحية أخرى ،هذا هو الحال مع معضلة التي نحن في إطار معالجتها المعروفة ب -NP) . hard)

الطريقة المقترحة هي Méta-heuristique التي تم اكتشافها مؤخرا البحث الوقواق .(Cuckoo search)

استخدمنا هذه الطريقة لحل مشكلتنا مع التحسينات اللازمة لإيجاد الحل الأمثل.

كلمات البحث: اختيار خدمات ويب، جودة الخدمة،الفوقية الاستدلال ،البحث الوقواق، التحسين التركيبي.