Remerciements

Avant tout ; nous remercions Allah de nous avoir aidé à faire ce travail

Nous remercions également nos enseignants pour la qualité de l'enseignement qu'ils nous prodigué au cours de ces 5 années passées à l'université Moulay Taher SAIDA.

Nous remercions tout particulièrement Mr. Mekour l'encadreur qui nous laissé une large part d'autonomie dans ce travail, et qui nous guidés vers des pistes de réflexions riches et porteuses.

Nous souhaiterions aussi remercier tout le corps administratif, ainsi que toutes les personnes qui souhaiteraient voir un jour notre université au meilleur rang.

Nous remercions enfin l'ensemble des nous proches qui nous aidé et motivé durant ce cursus , nous les remercions pour l'aide qu'ils nous apporté dans la réalisation de ce travail.

Merci à tous

. . . .

Dédicaces

Je tien a dédier ce modeste travail à tous ceux qui m'ont encouragé durant toute la période de réalisation de ce travail.

En particulier :

Mon très cher père, ma très chère maman qui ma toujours soutenu

***A** mon frère Oussama ; Ma sœur khaoula*

***A** Chers amies*

Rim, Chahinaz, Marwa, Amira, Soumia

***A** ma Binôme Bouchra*

***A** toutes ma famille (Tahri)*

***A** tous mes amis*

***A** tous ceux que j'aime et ceux qui m'aiment*

Tahri fatima Zohra

Et merci

. . . .

Dédicaces

Je tien a dédier ce modeste travail à tous ceux qui m'ont encouragé durant toute la période de réalisation de ce travail.

En particulier :

Mon très cher père Ahmed, ma très chère maman Naima qui m'a toujours soutenu, Mes chères sœurs HADJER, FATIMA, MERIEM, AICHA, RAJAA, HALLA

Ma grand-mère

Ma meilleure amie

CHAIMAA

***A** toutes ma famille (LOUAZANI et YAHI)*

***A** tous mes amis*

***A** ma Binôme Fatima*

***A** tous ceux que j'aime et ceux qui m'aiment*

Louazani Imane Bouchra

Et merci

. . . .

Résumé

Depuis l'apparition de l'Architecture Orientée Services (SOA) et son implémentation avec la technologie des services Web, de nombreux services Web, avec des fonctionnalités similaires sont fournis par des fournisseurs concurrents, un problème qui apparaît à la mise en oeuvre de cette technologie est que le processus de sélection devient assez complexe. Notre travail consiste à développer une application basée sur une approche d'optimisation multi-objective évolutionnaires (GA,NSGA2). L'objectif est de sélectionner un ensemble de services atomiques pour composer un nouveau service pour satisfaire les contraintes standards (minimiser le temps et le coût, et maximiser la disponibilité, la fiabilité et réputation) en fonction de QoWS.

mots clés : SOA,GA,NSGA2, service web, sélection de service, composition de services *....*

Abstract

Since the appearance of Service Oriented Architecture (SOA) and its implementation with web services technology, many web services, with similar functionality are provided by from competing suppliers, a problem that arises in the implementation of this technology is that the selection process becomes quite complex. Our job is to develop an application based on an optimization approach multi-objective evolutionary). The goal is to select a set of atomic services to compose a new service to meet standard constraints (minimize time and cost, and maximize availability, reliability and reputation) based on QoWS.

Keywords :SOA, web service, service selection, service composition, GA,NSGA2 ....

Table des matières

Table des figures

Listes des abréviations

SOA/AOS : Service-oriented architecture.

HTTP : Hypertext Transfer Protocol.

XML : eXtensible Markup Language.

QoS : Quality of Service.

W3C : World Wide Web Consortium.

WSDL : Web Service Definition Language.

UDDI : Universal Description, Discovery, and Integration.

SOAP : Simple Object Access Protocol.

GA/AG : Genetic Algorithme/Algorithme Génétique.

NSGA2 : non dominated sorting génétic algorithm 2

AE : algorithmes évolutionnaires

WSOA : web services oriented architecture

W3C : World Wide Web Consortium.

B2B : Busniss to Busniss.

B2C : Busniss to Consumer.

SMTP : Simple Mail Transfer Protocol.

FTP : File Transfer Protocol.

BEEP : Blocks Extensible Exchange Protocol.

API : Application programming interface.

PHP : Hypertext Preprocessor.

IBM : International Business Machines.

Introduction générale

Aujourd'hui, Le web est devenu une source d'informations dynamique dans lequel l'information est produite sur demande. Dans ce contexte les services web sont devenus le facteur le plus attirant car ils constituent un moyen rapide de diffuser des informations entre les clients, les fournisseurs, les partenaires commerciaux et leurs différentes plates-formes. Les services web reposent sur le modèle SOA (Service Architecture) [43] ,Il s'agit d'une technologie permettant à des applications de dialoguer à distance via Internet, et ceci indépendamment des plates-formes et des langages sur lesquelles elles reposent. Pour ce faire, les services Web s'appuient sur un ensemble de protocoles Internet très répandus XML (langage de balisage extensible), http (protocole de transport d'hyper texte), afin de communiquer. Cette communication est basée sur le principe de demandes et réponses, effectuées avec des messages XML.
L'objectif de ce travail est de sélectionner de façon optimale (ou proche de l'optimale) une combinaison de service web à base de leur qualité de services.
La composition de service [44]est de but de créer de nouvelles fonctionnalités en Combinant des fonctionnalités offertes par d'autres services existants, composés ou non en vue d'apporter une valeur ajoutée. Un Web service est dit composite lorsque son exécution implique des interactions avec d'autres Web services, et des échanges de messages entre eux afin de faire appel à leurs fonctionnalités. La composition de Web services spécifie quels services ont besoin d'être invoqués, dans quel ordre et comment gérer les conditions d'interaction.

Nous avons structuré ce mémoire en quatre chapitres : dans Le premier chapitre, nous présentons quelques définitions, les services web et ses technologies, nous avons cité quelques avantages et inconvénients des services web, l'intérêt

avec les caractéristiques du service web . Le deuxième chapitre parle de la notion de composition de services web et de la sélection selon la qualité de service QoS. Et selon la troisième chapitre on a traité la composition de services web en adaptant les algorithmes évolutionnaires (AE). Et la fin avec le dernier chapitre qui est dédié à l'implémentation de chaque algorithme sous forme d'une application.

1

Les services web

1.1 Introduction

L'information dépend de plus en plus sur des technologies Internet et avec la grande utilisation du web, les chercheurs ont développé des logiciels pour garantie et simplifier la communication entre les machines et les applications connectées via le réseau, ces logiciels sont appelés les services web "web services". Dans ce chapitre, nous présentons les concepts de base de l'architecture orientée service (AOS).

1.2 Historique

La légende raconte que c'est Bill Gates, alors président de Microsoft le premier qui a utilisé le terme Web Services. Il l'aurait fait le 12 Juillet 2000 au cours de (Microsoft Professional Developers Conference) à Orlando. Même si cette légende est controversée, il est plus certains en revanche que c'est chez Microsoft que ces mots ont été utilisés la première fois en les associant à SOAP, XML, WSDL et UDDI. Evidemment, tout ne s'est pas fait en jour, et la naissance des Services Web et des technologies qui les accompagnent remontent à un peu plus loin que cette date. En réalité, l'histoire commence en 1975 lorsque l'informatique souffrait encore de peu de standardisation et que les constructeurs et éditeurs se rendent compte de la nécessité d'uniformiser les échanges de données. Ils firent alors les vieux pieux de "l'interopérabilité " afin de standardiser la communication entre application au travers d'un réseau. C'est la naissance l'EDI (Electronic Data Interchange) ou (Echange de Données Informatisées), l'ancêtre des Web Services [1].

1.3 L'architecture SOA

1.3.1 Définition

Une architecture orientée services notée SOA pour (Services Oriented Architecture) est une architecture logicielle s'appuyant sur un ensemble de services simples. L'objectif d'une architecture orientée services est donc de décomposer une fonctionnalité en un ensemble de fonctions basiques, appelées services, fournies par des composants et de décrire finement le schéma d'interaction entre ces services. L'idée de base est de ne plus créer d'activités commerciales autour des applications afin de s'assurer qu'une architecture logicielle globale est intégrée dans des services compatibles avec les processus métier de l'entreprise. Lorsque l'architecture SOA s'appuie sur des web services, on parle alors de WSOA, pour(Web Services Oriented Architecture) [4].

1.4 Fonctionnement de SOA

L'utilisation de l'architecture SOA (Service Oriented Architecture) permet de rendre les applications complètement distribuées. La création d'une application qui utilise l'architecture SOA à grande échelle d'entreprise est une tâche difficile qui nécessite une connaissance approfondie. L'architecture orientée services (Service Oriented Architecture) , ou SOA constitue un style d'architecture basée sur le principe de séparation de l'activité métier en une série de services. Un service web SOA défaillant ou malveillant pourrait fournir à nos applications distribuées des données erronées et influencer sur les décisions finales prises à partir des données collectées. L'étude de la sûreté de fonctionnement de l'architecture orientée services[5] .

1.5 Définition service web

On peut trouver plusieurs définitions des services Web tel que :

1.5.1 Citation 1 :W3C

Selon la définition du W3C (World Wide Web Consortium), un Web service ou (service Web) est une application appelable via Internet par une autre application d'un autre site Internet permettant l'échange de données (de manière textuelle) afin que l'application appelante puisse intégrer le résultat de l'échange à ses propres analyses. Les requêtes et les réponses sont soumises à des standards et normalisées à chacun de leurs échanges[2].

1.5.2 Citation 2 : Dico du Net

Une technologie permettant à des applications de dialoguer à distance via Internet indépendamment des plates-formes et des langages sur lesquels elles reposent[1].

1.5.3 Citation 3 :

Un service Web est une application modulaire que vous pouvez décrire, publier, localiser et invoquer sur le Web. Un service Web remplit des fonctions, qui peuvent aller de simples demandes à des processus opérationnels compliqués. Une fois qu'un service Web est déployé, d'autres applications ou d'autres services Web peuvent découvrir et invoquer le service déployé [3].
En d'autres termes, un service Web est tout simplement un programme accessible au moyen d'Internet, qui utilise un système de messagerie standard XML[1].

1.6 L'architecture d'un service web

Pour comprendre le fonctionnement d'une architecture de services Web, il faut commencer par revoir certains principes. Si l'on reprend la définition de Mark Colan, Web Service and XML Chief Advocate chez IBM, les Web Services sont des "applications modulaires basées sur Internet qui exécutent des tâches précises et qui respectent un format spécifique". Ce sont donc des unités logiques applicatives qui sont accessibles grâce au protocole Internet. Une définition conceptuelle du terme service Web mettrait en avant les qualités d'une fonctionnalité commerciale présentée par une entité hétérogène quelconque sur Internet afin de fournir un moyen d'user de ce service à distance. Pour l'aspect opérationnel, les services Web ne sont que des applications modulaires qui peuvent être présentées, publiées, situées et invoquées dans un réseau et ce automatiquement. Ainsi, les applications peuvent faire appel à des fonctionnalités situées sur d'autres machines dans d'autres applications. Au final, on peut affirmer que le but initial d'un service Web est de rendre possible l'utilisation d'un composant applicatif de façon distribuée. L'apport majeur de ce modèle d'échange de données est d'introduire ces services comme des "boîtes noires". En effet, les requêtes-réponses d'un service Web sont administrées dans le contenu de messages dont on sait la forme grâce à des interfaces clairement présentées et sur lesquelles l'implémentation interne du traitement et le langage employé ne jouent pas au niveau de l'architecture. Grâce à cela on obtient un haut niveau de modularité et d'interopérabilité. Ce modèle de message permet donc d'oublier la structure, le langage ou encore la plate-forme qui va porter le service : il suffit juste que le message suive une architecture donnée pour qu'il puisse être analysé. Il s'agit maintenant d'identifier chaque acteur de ses Web

services et de comprendre comment ils interagissent les uns avec les autres. Les trois éléments les plus importants des services Web sont les fournisseurs de service, les annuaires de services et les consommateurs de service. Le fournisseur (ou serveur) crée le service Web et publie toutes ces caractéristiques dans l'annuaire de service. L'annuaire rend disponible les interfaces d'accès aux services et donnant le contrat et l'architecture employée pour permettre les interactions. Le consommateur (ou client) quant à lui, accède à l'annuaire pour rechercher les services Web dont il a besoin et avec lui les normalisations à obtenir. Il peut ainsi envoyer ses requêtes au service désiré et obtenir les réponses qu'il pourra analysé. Cette architecture fonctionne de la manière suivante :

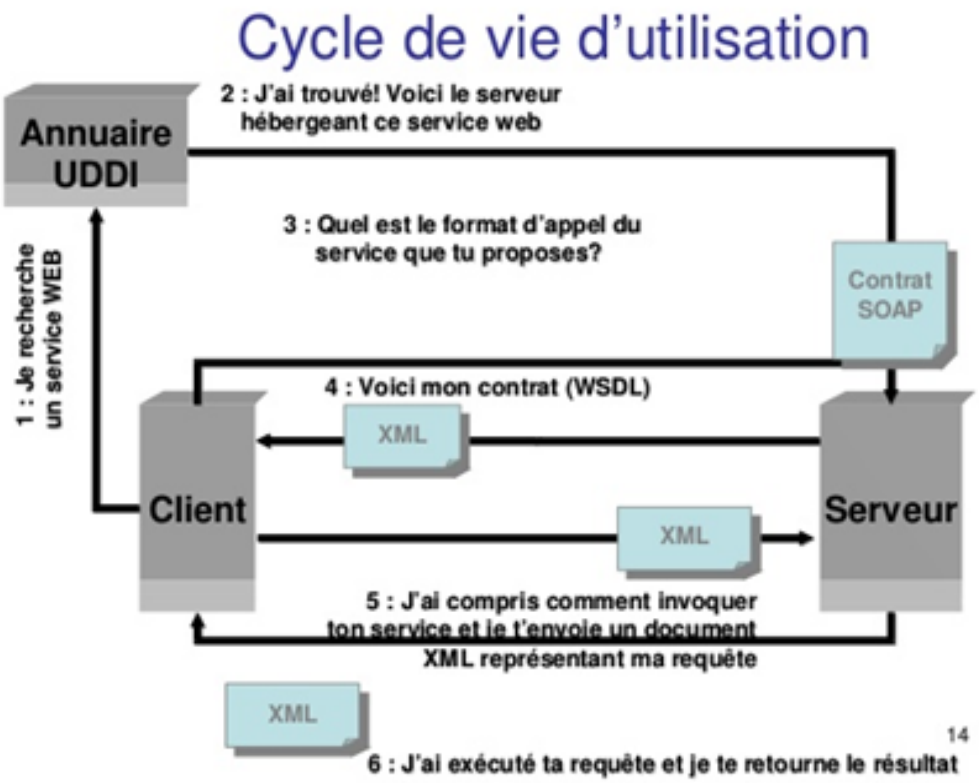


FIGURE 1.1 : L'architecture d'un service web.

Explication :

1. Le client envoie une requête à l'annuaire de Service pour trouver le service

Web dont il a besoin.

2. L'annuaire cherche pour le client, trouve le service Web approprié et renvoie une réponse au client en lui indiquant quel serveur détient ce qu'il recherche.
3. Le client envoie une deuxième requête au serveur pour obtenir le contrat de normalisation de ses données.
4. Le serveur envoie sa réponse sous la forme établie par WSDL en langage XML.
5. Le client peut maintenant rédiger sa requête pour traiter les données dont il a besoin.
6. Le serveur fait les calculs nécessaires suite à la requête du client, et renvoie sa réponse sous la même forme normalisée [2].

1.7 Fonctionnement d'un service web

Services web non publics :

— Services web non publiés dans un annuaire UDDI.
— Services web dont le point d'accès est connu des utilisateurs du service.
— Généralement des SW intranet, des SW de type B2B (Business to Business)[6].

FIGURE 1.2 : Fonctionnement d'un service web à accès non public.

Services Web publics :

— Services web publiés dans un annuaire UDDI.

— Généralement des SW de type B2C (Business to Consumer), ex : agence de voyage,...etc.

— Le SW et l'annuaire UDDI qui le publie peuvent ne pas résider sur la même machine[6] .

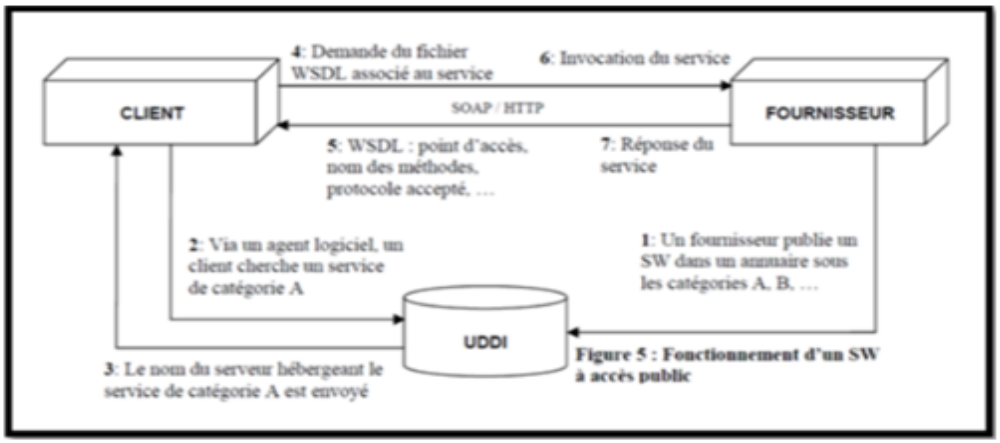


FIGURE 1.3 : Fonctionnement d'un service web à accès public.

1.8 L'intérêt d'un Service Web :

Les services Web fournissent un lien entre applications. Ainsi, des applications utilisant des technologies différentes peuvent envoyer et recevoir des données au travers de protocoles compréhensibles par tout le monde. Les services Web sont normalisés car ils utilisent les standards XML et HTTP pour transférer des données et ils sont compatibles avec de nombreux autres environnements de développement. Ils sont donc indépendants des plates-formes. C'est dans ce contexte qu'un intérêt très particulier a été attribué à la conception des services Web puisqu'ils permettent aux entreprises d'offrir des applications

accessibles à distance par d'autres entreprises. Cela s'explique par le fait que les services Web n'imposent pas de modèles de programmation spécifiques. En d'autres termes, les services Web ne sont pas concernés par la façon dont les messages sont produits ou consommés par des programmes. Cela permet aux vendeurs d'outils de développement d'offrir différentes méthodes et interfaces de programmation au-dessus de n'importe quel langage de programmation, sans être contraints par des standards comme c'est le cas de la plate-forme CORBA qui définit des ponts spécifiques entre le langage de définition IDl et différentes langages de programmation. Ainsi, les fournisseurs d'outils de développement peuvent facilement différencier leurs produits avec ceux de leurs concurrents en offrant différent niveaux de sophistication. Les services Web représentent donc la façon la plus efficace de partager des méthodes et des fonctionnalités. De plus, ils réduisent le temps de réalisation en permettant de tirer directement parti de services existants[8].

1.9 Les caractéristiques d'un service Web

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

Un service Web possède les caractéristiques suivantes :

— Il est accessible via le réseau.
— Dispose d'une interface publique (ensemble d'opérations) décrite en XML.

— Ses descriptions (fonctionnalités, comment l'invoquer et où le trouver ?) sont stockées dans un annuaire.
— Il communique en utilisant des messages XML, ces messages sont transportés par des protocoles Internet (généralement HTTP, mais rien n'empêche d'utiliser d'autres protocoles de transfert tels : SMTP, FTP, BEEP...)
— L'intégration d'application en implémentant des services Web produit des systèmes faiblement couplés, le demandeur du service ne connaît pas forcément le fournisseur.
— Ce dernier peut disparaître sans perturber l'application cliente qui trouvera un autre fournisseur en cherchant dans l'annuaire[8].

1.10 Architecture des composants des services web

1.10.1 SOAP

Pourquoi SOAP ?

Dans le monde d'aujourd'hui, il existe un grand nombre d'applications basées sur différents langages de programmation. Par exemple, il pourrait y avoir une application Web conçue en Java, une autre en .Net et une autre en PHP.

L'échange de données entre applications est crucial dans le monde en réseau d'aujourd'hui. Mais l'échange de données entre ces applications hétérogènes serait complexe. Il en sera de même de la complexité du code pour réaliser cet échange de données.

L'une des méthodes utilisées pour lutter contre cette complexité consiste à utiliser XML (Extensible Markup Language) comme langage intermédiaire

pour l'échange de données entre applications. Chaque langage de programmation peut comprendre le langage de balisage XML. Par conséquent, XML a été utilisé comme support sous-jacent pour l'échange de données. Mais il n'y a pas de spécification standard sur l'utilisation de XML dans tous les langages de programmation pour l'échange de données. C'est là que SOAP entre en jeu.

SOAP a été conçu pour fonctionner avec XML sur HTTP et avoir une sorte de spécification qui pourrait être utilisée dans toutes les applications[11].

1.10.2 Définition SOAP

SOAP est un format de messages qui permet de transmettre en XML les appels de procédures distantes. SOAP permet aussi d'envoyer en XML un document entier d'un ordinateur à un autre. SOAP a été défini par l'IETF en Décembre 1999. Après qu'IBM ait rejoint le mouvement, SOAP a été spécifié par le W3C (World Wide Web Consortium) en Mai 2000. La version 1.2 est actuellement en cours de recommandation[10].

1.10.3 La structure d'un message soap

SOAP définit un format pour l'envoi des messages. Les messages SOAP sont structuré en un document XML et comporte 2 éléments obligatoires : Une enveloppe et un corps (une entête facultative)[9].

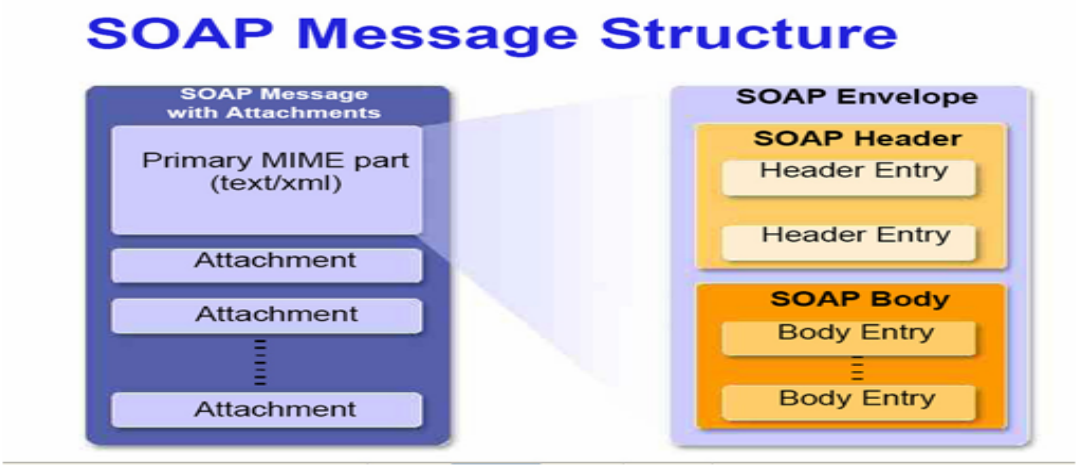


FIGURE 1.4 : La structure d'un message soap.

Le schéma ci-dessus montre la structure d'un message SOAP.

Explication :

- Une enveloppe qui définit le contenu du message
- Un en-tête optionnel qui contient les informations d'en-tête (autorisations et transactions par exemple)
- Un corps contenant les informations sur l'appel et la réponse
- Des attachements optionnels.

1.10.4 Exemples d'un message SOAP

Soit un dialogue RPC encodé par SOAP qui contient un message de requête et un message de réponse. Considérons la méthode d'un service simple qui double la valeur d'un entier donné.

Signature de la Methode

```
int doubleAnInteger ( int numberToDouble );
```

Voici la requête :

Requête

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<SOAP-ENV:Envelope
 SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
 xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
 xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance"
 xmlns:xsd="http://www.w3.org/1999/XMLSchema">
      <SOAP-ENV:Body>
              <ns1:doubleAnInteger
               xmlns:ns1="urn:MySoapServices">
                      <param1 xsi:type="xsd:int">123</param1>
              </ns1:doubleAnInteger>
      </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Dans la requête ci-dessus, on retrouve bien les deux éléments obligatoires caractéristiques d'un message SOAP. Le tag de l'enveloppe dans le message de requête contient également des définitions de "namespaces". On trouve ensuite le tag SOAP Body qui encapsule le tag de méthode qui porte le nom de la méthode elle même (ou le même nom suivi de "response" dans le cas du message de réponse[9].

1.10.5 Principe de fonctionnement de SOAP

Explication :

Coté client :

Le client envoie des messages au serveur correspondant à des requêtes SOAP-XML enveloppés dans des requêtes HTTP. De même, les réponses des serveurs sont des réponses HTTP qui renferment des réponses SOAP-XML. Dans le processus client, l'appel de service est converti en une requête SOAP qui est ensuite enveloppé dans une HTTP[9].

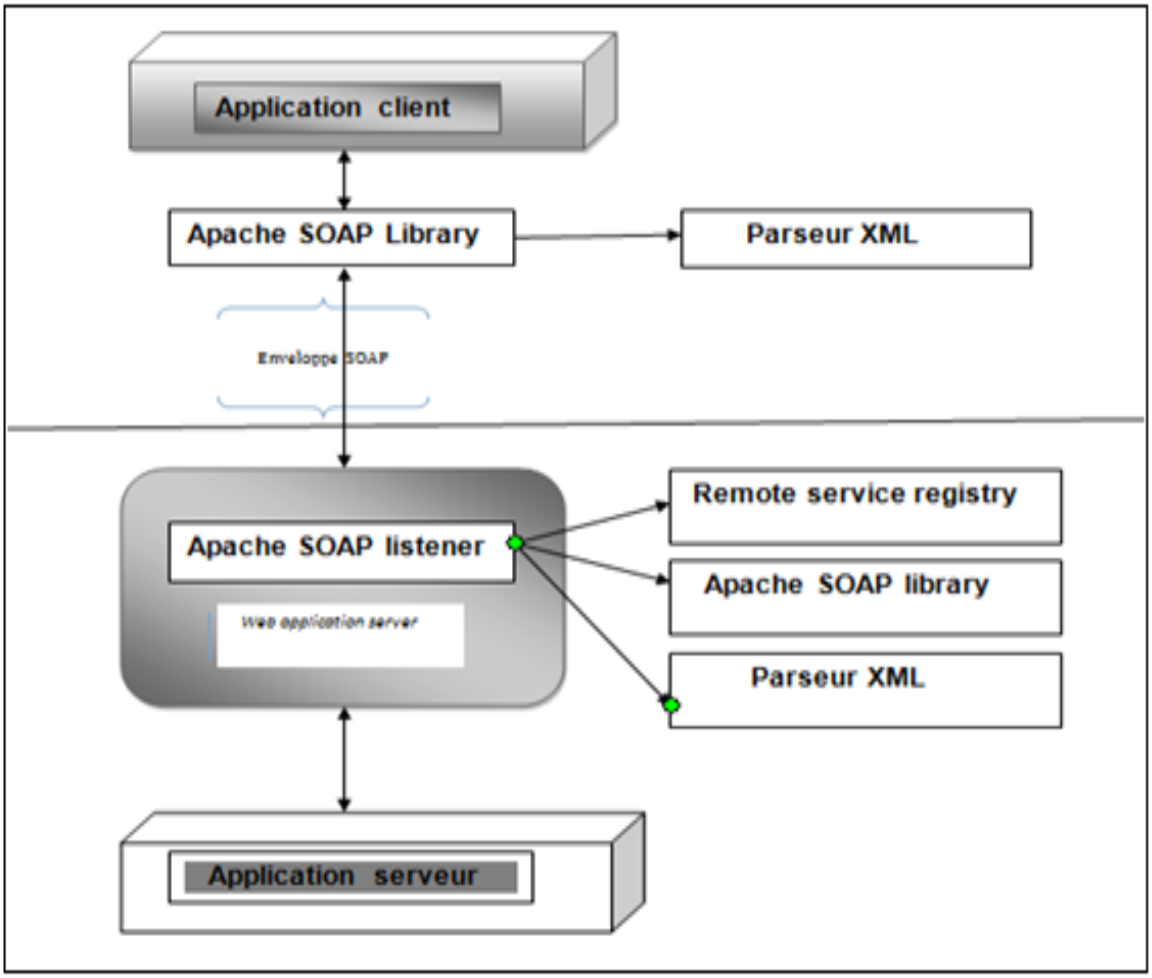


FIGURE 1.5 : Principe de fonctionnement de SOAP.

Coté serveur :

C'est légèrement plus complexe car il requiert un processus listener correspondant au processus serveur en attente de connexion cliente. Le listener est souvent implémenté au travers d'un servlet qui s'exécute qui a pour tache d'extraire le message XML-SOPA de la requête HTTP, de le désérialiser c'est à dire de séparer le nom de la méthode et les paramètres fournis puis invoquer la méthode du service en conséquence. Le résultat de la méthode est alors sérialisé, encodé HTTP et renvoyé au demandeur[9].

1.11 WSDL (Web Services Description Language)

1.11.1 Définition

WSDL[12] (Web Services Description Language) est un fichier XML qui indique essentiellement à l'application cliente ce que fait le service Web. Le fichier WSDL est utilisé pour décrire en bref ce que fait le service Web et donne au client toutes les informations nécessaires pour se connecter au service Web et utiliser toutes les fonctionnalités fournies par le service Web. WSDL 1.1[13] a été soumis en tant que note W3C par Ariba, IBM et Microsoft pour décrire les services pour l'activité XML du W3C sur les protocoles XML en mars 2001.

WSDL 1.1 n'a pas été approuvé par le W3C (World Wide Web Consortium), mais il vient de publier un projet de version 2.0 qui sera une recommandation (une norme officielle), et donc approuvé par le W3C.

1.11.2 Structure d'un document WDSL

Un document WSDL définit les services comme des ensembles de points de terminaison réseau ou ports. Dans WSDL, la définition abstraite des points de terminaison et des messages est séparée de leur déploiement réseau concret ou de leurs liaisons de format de données. Cela permet de réutiliser des définitions abstraites : des messages, qui sont des descriptions abstraites des données échangées, et des types de ports qui sont des collections abstraites d'opérations. Le protocole concret et les spécifications de format de données pour un type de port particulier constituent une liaison réutilisable. Un port est défini en associant une adresse réseau à une liaison réutilisable, et une collection de ports définit un service. Par conséquent, un document WSDL utilise les éléments suivants dans la définition des services réseau[14].

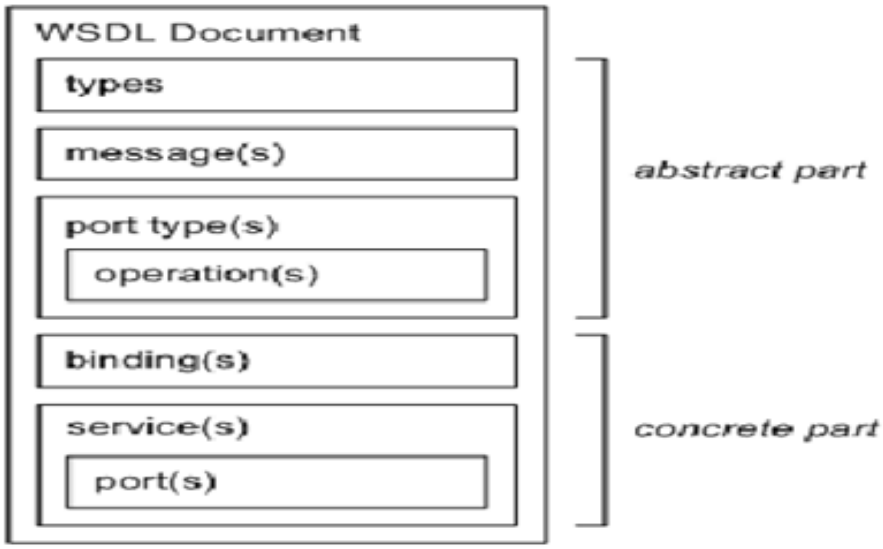


FIGURE 1.6 : La structure d'un document WSDL..

Types : un conteneur pour les définitions de types de données utilisant un système de type (tel que XSD).

Message : une définition abstraite et typée des données communiquées.

Opération : une description abstraite d'une action prise en charge par le service.

Type de port : un ensemble abstrait d'opérations prises en charge par un ou plusieurs points de terminaison.

binding : un protocole concret et une spécification de format de données pour un type de port particulier.

Port : un point de terminaison unique défini comme une combinaison d'une liaison et d'une adresse réseau.

Service : : une collection de points de terminaison associés.

1.12 UDDI (Universal Description, Discovery and Integration)

1.12.1 Définition

Un registre UDDI (Universal Decription Discovery and Integration) est un annuaire basé sur XML qui permet de publier des services et facilite leur découverte par d'autres services en définissant comment ils interagissent. Un scénario d'utilisation possible est donc la publication d'un fournisseur de service (donc de son WSDL) auprès d'un registre en créant tout d'abord une entreprise et une catégorie de service. Un client demande ensuite à un registre UDDI la localisation d'un service qui correspond au service venant d'être ajouté à l'annuaire. Le WSDL du service demandé est alors reçu par le client qui peut communiquer avec le fournisseur de services en SOAP[17] . La spécification UDDI offre également une API aux applications clientes, pour consulter et extraire des données concernant un service et / ou son fournisseur[16].

1.12.2 Les registres UDDI

Le registre UDDI implémente la spécification UDDI. UDDI est un annuaire distribué basé sur le Web qui permet aux entreprises de se répertorier sur Internet (ou Intranet) et de se découvrir, de la même manière que les pages jaunes et blanches d'un annuaire téléphonique traditionnel. Le registre UDDI est à la fois un annuaire professionnel de pages blanches et une bibliothèque de spécifications techniques. Le registre est conçu pour stocker des informations sur les entreprises et les services et contient des références à une documentation détaillée[15].

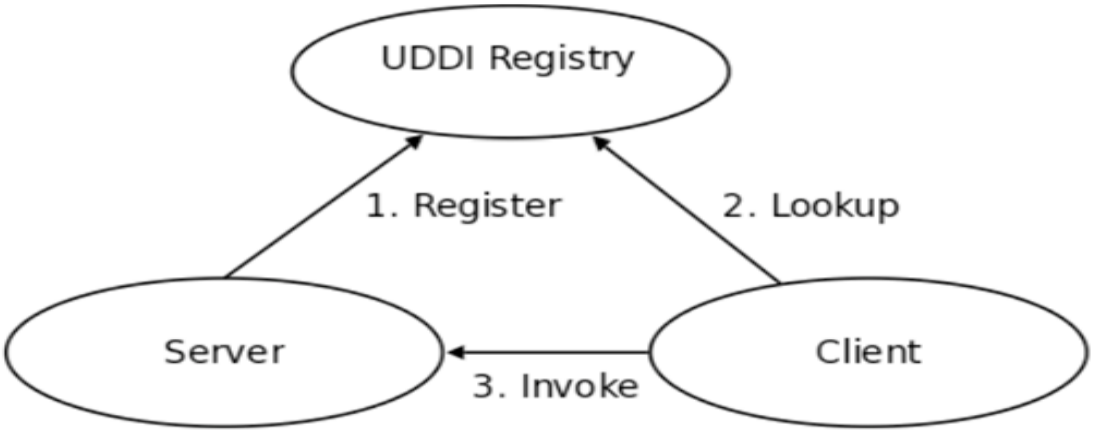


FIGURE 1.7 : Modèle d'invocation à l'aide du registre UDDI.

Explication :

— À l'étape 1 de la figure "Modèle d'appel à l'aide du registre UDDI", il est montré comment une entreprise publie des services dans le registre UDDI.

— À l'étape 2, un client recherche le service dans le registre et reçoit des informations de liaison de service.

— Enfin, à l'étape 3, le client utilise ensuite les informations de liaison pour appeler le service.

Les API UDDI sont basées sur SOAP pour des raisons d'interopérabilité. Dans cet exemple, nous avons trois API spécifiées dans la spécification UDDI v3, Security, Publication and Inquiry. La spécification UDDI v3 définit 9 API :

— UDDI_Security_PortType : définit l'API pour obtenir un jeton de sécurité. Avec un jeton de sécurité valide, un éditeur peut publier dans le registre. Un jeton de sécurité peut être utilisé pour toute la session.

— UDDI_Publication_PortType : définit l'API pour publier des informations commerciales et de service dans le registre UDDI.

— UDDI_Inquiry_PortType : définit l'API pour interroger le registre UDDI. En règle générale, cette API ne nécessite pas de jeton de sécurité.

— UDDI_CustodyTransfer_PortType : cette API peut être utilisée pour transférer la garde d'une entreprise d'un nud UDDI à un autre.

— UDDI_Subscription_PortType : définit l'API pour s'inscrire aux mises à jour sur une entreprise de service particulière.

— UDDI_SubscriptionListener_PortType : définit l'API qu'un client doit implémenter pour recevoir des notifications d'abonnement d'un nud UDDI.

— UDDI_Replication_PortType : définit l'API pour répliquer les données de registre entre les nuds UDDI.

— UDDI_ValueSetValidation_PortType : par nuds pour permettre aux fournisseurs externes de validation d'ensemble de valeurs. Services Web pour évaluer si (keyedReferences) ou (keyedReferenceGroups) sont valides.

— UDDI_ValueSetCaching_PortType : les nuds UDDI peuvent effectuer eux-mêmes la validation des références d'éditeur à l'aide des valeurs mises en cache obtenues à partir d'un tel service Web

1.13 Les avantages et les inconvénients des Services Web

1.13.1 Avantage

Le principal avantage des services Web réside dans le fait que la communication peut s'effectuer sur diverses plateformes. Le client et le serveur n'ont pas besoin d'avoir grand-chose en commun pour que la communication fonctionne. Pour permettre cette dernière, les services Web ont recours à des formats standardisés compris par tous les systèmes[7].

1.13.2 Inconvénient

L'un des inconvénients provient toutefois de ces formats. XML est un format

plutôt encombrant qui entraîne de grands paquets de données. Dans le cas de connexions réseau lentes, cela peut être source de problèmes. Une autre possibilité pour connecter deux systèmes ensemble via Internet consiste à utiliser des API Web. Il s'agit d'interfaces pouvant également être consultées via Internet. Elles sont généralement plus rapides, mais fixent des objectifs beaucoup plus clairs pour le client et le serveur ce qui limite l'interopérabilité[7].

1.14 Conclusion

Dans ce chapitre nous avons présenté une vision générale sur l'architecture oriente service SOAP cette architecture qui nous présente le concept de service Web, il représente actuellement l'ensemble de standards les plus connus pour la réalisation des applications Internet. Cette technologie repose sur des standards XML très populaires. Nous avons aussi introduis brièvement la notion de Web sémantique et leurs utilités.

2

Composition des services web

2.1 Introduction

La composition de services est un sujet de grand intérêt autant pour le monde de la recherche que pour le monde industriel. De nombreux travaux de recherche visent à développer des modèles de composition de services et à fournir les outils nécessaires pour la composition. L'ensemble du processus de composition de services doit être considéré plus largement que le problème de combinaison des services. Il comprend également le problème de la description des buts de l'utilisateur, de l'acquisition des données nécessaires de l'utilisateur, de la conception et l'exécution du meilleur service composite possible, et de la présentation des résultats à l'utilisateur.

2.2 Définition de la composition de services web

La composition des services Web est le processus de construction de nouveaux services Web valeur ajoutée, à partir de deux ou plusieurs services Web déjà présents et publiés sur le Web.
Un service Web est dit composé ou composite lorsque son exécution implique des interactions avec d'autres services Web, et des changements des messages entre eux afin de faire appel à leurs fonctionnalités. La composition de services Web spécifie quels services ont besoin d'être invoqués, dans quel ordre et comment gérer les conditions d'interaction[19]. En d'autres termes, la composition des Web services est la combinaison des Web services existants pour former de nouveaux services. L'objectif principal de la composition de Web service est la possibilité de créer de nouvelles fonctionnalités d'un nouveau Web service, en combinant des fonctionnalités offertes par d'autres Web services existants. Elle implique la capacité de sélectionner, de coordonner, d'interagir, et de faire

inter-opérer des Web services existant comme il est indiqué dans la figure 2.1.

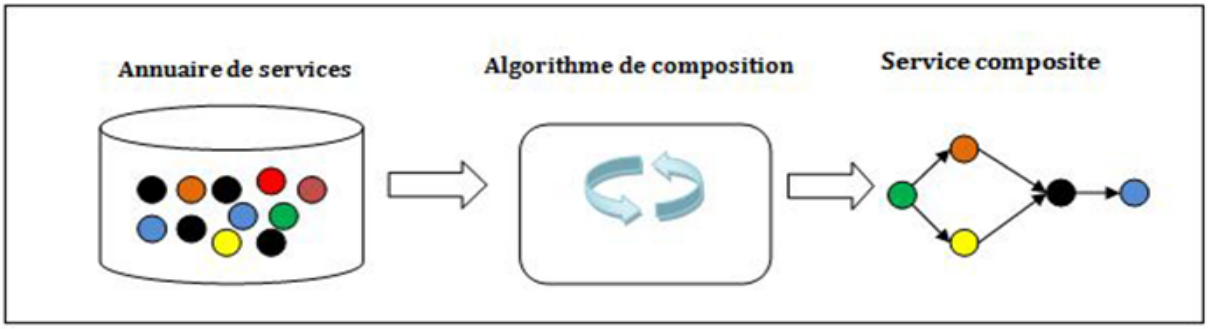


FIGURE 2.1 : Composition de services web.

2.3 Cycle de vie d'une composition de service web :

La composition des services web comporte une séquences d'étapes de construction tel que[20] :

a.L'encapsulation de services natifs : Cette activité permet de s'assurer que tout service peut être appelé lors d'une composition, indépendamment de son modèle de données, de son format de message, et de son protocole d'interaction.

b.L'établissement d'accord d'externalisation : Cette activité consiste à négocier, établir, et appliquer des obligations contractuelles entre les services.

c.L'assemblage de services composants : Cette activité permet de spécifier, à un haut niveau d'abstraction, l'ensemble des services à composer afin d'atteindre l'objectif attendu. Cet assemblage comporte une phase d'identification des services et de spécification de leurs interactions conformément aux descriptions et aux accords entre services.

d.L'exécution de services composants : Cette activité consiste en l'exécution des spécifications de la composition précédemment définies.

e.Le contrôle de l'exécution de services composites : cette activité permet de superviser l'exécution de la composition en vérifiant, par exemple, l'accès aux services, les changements de statut, les échanges de messages. Ce contrôle permet de détecter des violations de contrats, de mesurer les performances des services appelés et de prédire des exceptions.

f.L'évolutivité des services : Cette dernière activité permet de faire évoluer la composition en modifiant les altérations de l'organisation de services, en utilisant de nouveaux services, ou en prenant en compte les retours de la phase de contrôle.

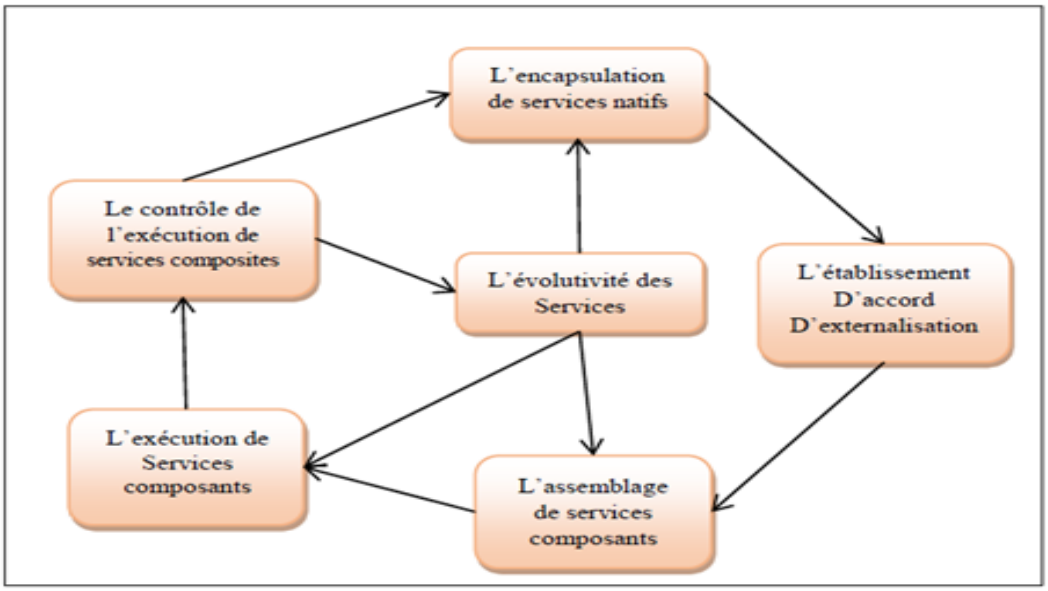


FIGURE 2.2 : Illustration du cycle de vie de d'une composition de SW. [20]

2.4 Classification de composition des services web

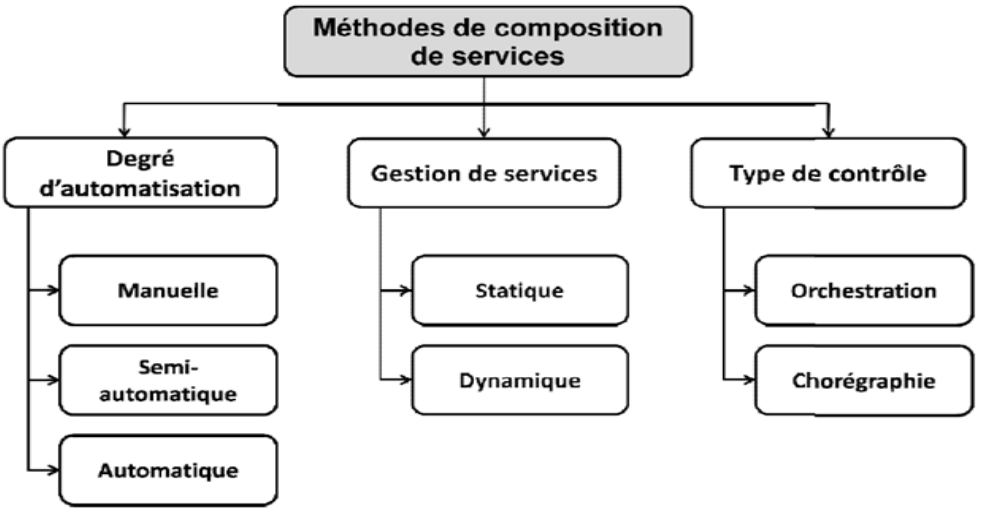


FIGURE 2.3 : Classification des méthodes de composition de services.

Les solutions proposées peuvent être classifiées selon deux axes :

1. En fonction du degré de participation de l'utilisateur dans la définition du schéma de composition, ces propositions sont manuelles, semi-automatiques ou automatiques [21] :

a.Composition manuelle : la composition manuelle des services web suppose que l'utilisateur gère la composition à la main via un éditeur de texte et sans l'aide d'outils dédiés.

b.Composition semi-automatique : les techniques de composition semi-automatiques sont un pas en avant en comparaison avec la composition manuelle, dans le sens qu'elles font des suggestions sémantiques pour aider à la sélection des services web dans le processus de composition.

c.Composition automatique : la composition totalement automatisée prend en charge tout le processus de composition et le réalise automatiquement, sans qu'aucune intervention de l'utilisateur ne soit requise.

2. Selon que, la sélection des services web et la gestion du flot soient faites ou non a priori, une autre approche sera dite statique ou dynamique :

2.4.1 Composition statique des services web :

Ce type de composition peut être appliqué dans des environnements "stables" où les services Web participants sont toujours disponibles et où le comportement du service composite est le même pour tous les clients. La composition des services web prend place durant la période de conception. Les composants sont choisis, relient entre eux et, compilés et déployés. Le service composite ainsi obtenu fonctionnera bien tant que son environnement et les services qui le composent ne changent pas ou ne changent que rarement.

On distingue deux types de composition par procédés : orchestration et chorégraphie.

a.Orchestration : Dans l'orchestration, un seul processus (appelé orchestrateur) est responsable de la composition et contrôle les interactions entre les différents services. Cet orchestrateur coordonne de manière centralisée les différentes opérations des services partenaires qui n'ont aucune connaissance de cette composition[22].

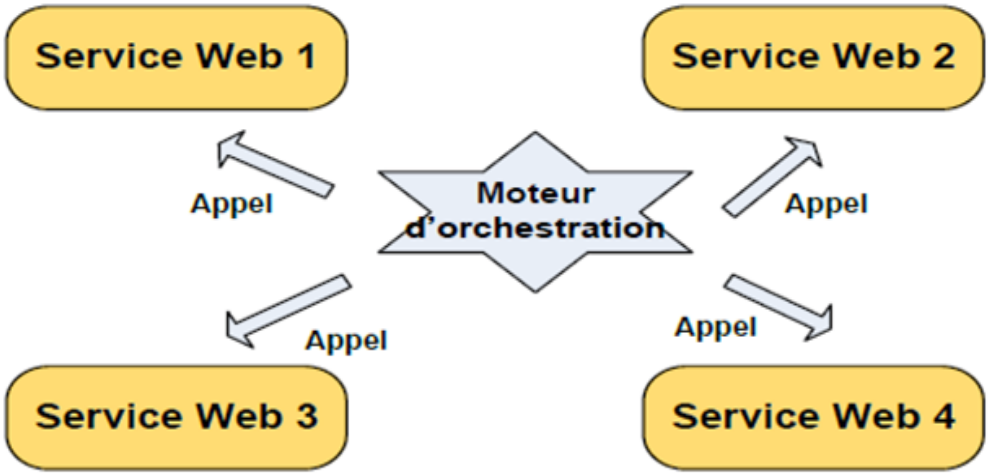


FIGURE 2.4 : Vue générale de l'orchestration
[23]

b.Chorégraphie : Chorégraphie de services Web La chorégraphie décrit une collaboration entre services pour accomplir un certain objectif. A l'inverse de l'orchestration, la chorégraphie ne repose pas sur un seul processus principal. C'est une coordination décentralisée ou chaque participant est responsable d'une partie du workflow[22].

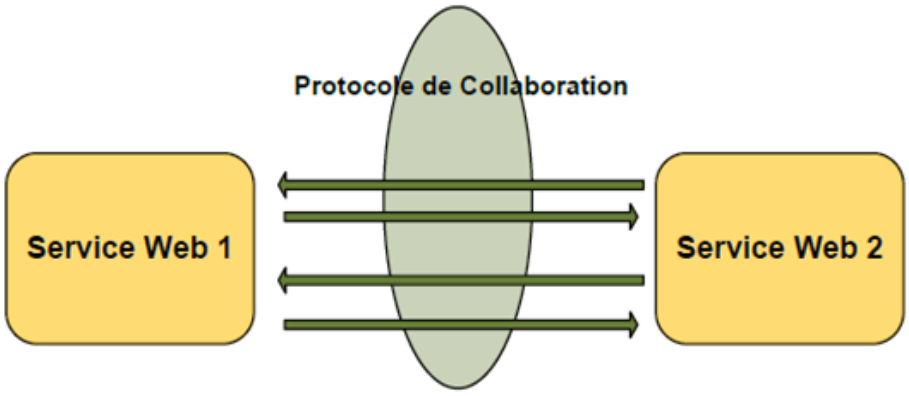


FIGURE 2.5 : Vue générale de la chorégraphie.
[23]

En résumé, la différence majeure entre l'orchestration et la chorégraphie est

que l'orchestration offre une vision centralisée de la composition, alors que la chorégraphie offre une vision globale et plus collaborative de la coordination.

2.4.2 Composition dynamique des services web :

Dans ce type de composition, les services Web à composer sont déterminés lors de l'exécution de la requête d'un client. Ils peuvent être déterminés selon les contraintes de chaque client, la disponibilité des services Web, etc. La composition dynamique apparaît la plus intéressante d'une part, elle promet d'être capable de faire face à un environnement très dynamique dans lequel des services apparaissent et disparaissent rapidement. D'autre part, elle permet de mieux satisfaire les besoins de chaque client en minimisant son intervention.

a.Approches orientés Workflow Un Workflow est une abstraction d'un processus de type business (métier), Il est composé d'un ensemble d'activités élémentaires structurées, ainsi que l'ordre d'exécution entre elles, et de dépendances entre ces activités. De la même façon les services composants ainsi que les contrôles et échanges de données entre ces services peut être vu comme un Workflow.
Les méthodes qui modélisent la composition des services Web à base de Workflows peuvent être statiques ou dynamiques. Dans le cas statique, l'utilisateur établit manuellement un modèle abstrait des tâches, en plus des dépendances. (i.e. le flux de contrôle, le flux de données). Chaque tâche contient une clause ou question qui est employée pour rechercher le service concret qui l'accomplit. Ce travail est fait pendant la conception. Durant l'exécution, le système de composition instancie chaque tâche, en cherchant un service concret.
Par contre les méthodes qui modélisent la composition des services Web à base

de Workflows dynamiques génèrent les tâches instanciées à la volée pendant l'exécution (runtime). La composition à base de Workflows dynamiques crée le modèle de tâches et l'instanciation des services de façon automatique. Dans cette vision, une composition automatique demande des Workflows capables de reconnaître les services web correspondants à chaque tâche, mais aussi de trouver d'autres services nécessaires pour la substitution dans le cas ou ceux-ci soient indisponibles à leur exécution normal[24].

b. Approches orientées intelligence artificielle La composition de services par des techniques issues de l'intelligence artificielle, et plus particulièrement par des techniques à base de règles, est la voie qui semble prometteuse pour certains auteurs comme Dans ce qui suit, nous présentons quelques axes de recherche concernant la composition par planification, par SMA et par d'autres techniques d'intelligence artificielle[25].

La planification : La planification est un des domaines de l'Intelligence Artificielle qui permet de choisir et d'organiser des actions en produisant un plan.L'exécution de plan modifie les propriétés du système en le faisant évoluer de l'état initial jusqu'au but désiré[26]. Le problème de composition des services Web peut être modélisé comme un problème de planification où les services sont vus comme des actions et la composition comme un plan. Un plan solution c'est le schéma d'exécution d'un ensemble de services (actions) à partir de l'état initial jusqu'à atteindre l'état but.

La plupart des approches de planification utilise un modèle conceptuel de transition d'état. Plus formellement, le problème de la planification correspond à un quadruplet.[27]

$\sum$ (S,A,E,Y)

ou

S = s1,s2, ... est un ensemble d'états fini.

A = a1,a2, ... est un ensemble d'actions fini.

E = e1,e2, ... est un ensemble d'événements fini.

et : S * A * E $\longrightarrow 2^s$: est la fonction de transition d'état.

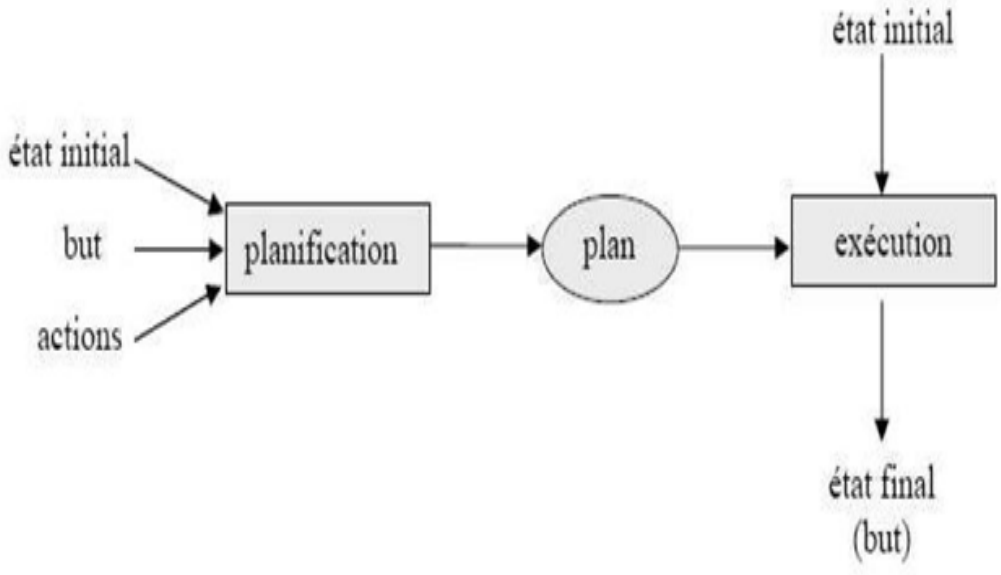


FIGURE 2.6 : Problème de planification.

Preuve de théorèmes : Dans cette approche , La requête de l'utilisateur est décrite comme un théorème que l'on souhaite prouver. Initialement, les services Web disponibles et les pré-requis de l'utilisateur sont décrits dans la logique du premier ordre. la requête est formulée comme un théorème ou la description de service est la spécification et l'objectif recherché est l'entrée. Ensuite, une preuve est générée par le générateur de preuve de théorème SNARK. Enfin, la réponse de la requête (la description de la composition de services) est extraite d'une preuve particulière en fonction de la disponibilité des services Web[28].

Calcul situationnel : Le Calcul de Situation [26][27]un langage basé sur la lo-

gique du premier ordre permettant de représenter et raisonner sur un domaine dynamique. Ce formalisme a été introduit par John McCarthy et Patrick J. Hayes en 1969.Les principaux éléments sont les actions, les fluents et les situations. Le monde est conçu comme un arbre de situations, débutant par la situation initiale s0, et évoluant jusqu'à la nouvelle situation par l'application d'une ou plusieurs actions. Une situation s donnée correspond toujours à un historique de l'ensemble d'actions réalisées sur s0 (Voir la Figure).

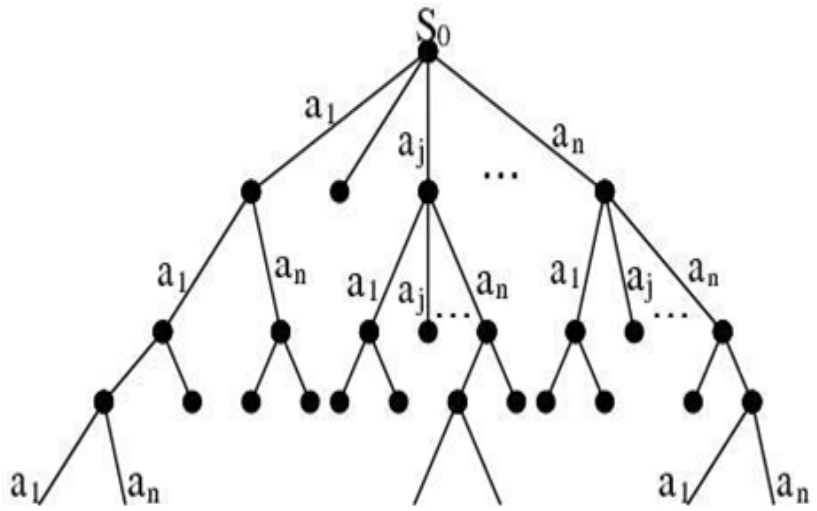


FIGURE 2.7 : L'ensemble d'actions réalisées sur s0.

Le problème de la composition des services Web est abordé de la façon suivante : la requête de l'utilisateur et les contraintes des services sont représentées en termes de prédicats du premier ordre dans le langage de calcul situationnel. Les services sont transformés en actions (primitives ou complexes) dans le même langage. Puis, à l'aide de règles de déduction et des contraintes, des modèles sont ainsi générés et sont instanciés à l'exécution à partir des préférences utilisateur, et l'ensemble de services atomiques sont reliés dans un langage de programmation procédural (if-then-else, while etc).

2.5 L'invocation de services dans une composition

L'invocation des services dans une composition sont dans un ordre bien précis. On distingue les types d'exécution suivant :

2.5.1 Exécution séquentielle :

Dans une exécution séquentielle, un service est invoque une fois que tous les Web services précédents ont été exécutes. La figure suivante montre ce type :

FIGURE 2.8 : Exécution séquentielle.

.

2.5.2 Exécution en parallèle :

Dans ce cas, les Web services s'exécutent en parallèle. Elle peut être représentée par l'opérateur AND. Dans figure suivante, le Web service 2 s'exécute en parallèle avec le Web service 3.

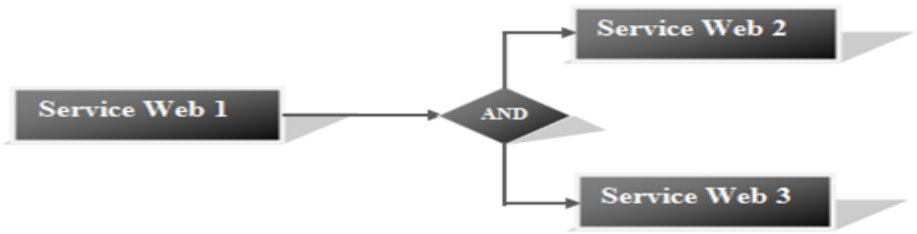


FIGURE 2.9 : Exécution en parallèle .

.

2.5.3 Exécution conditionnelle :

Un chemin est choisi parmi plusieurs, ce choix est fait a l'aide d'une décision prise au moment de l'exécution. Elle peut être représentée par l'opérateur OR. La figure suivante montre ce type :

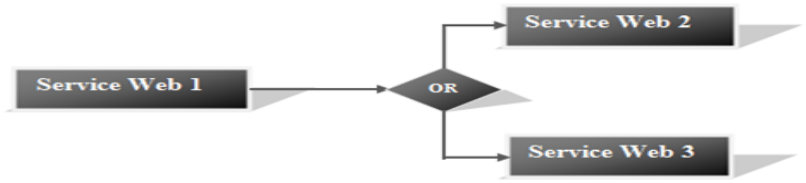


FIGURE 2.10 : Exécution conditionnelle .

.

2.6 Les stratégies de sélection

L'exécution d'un Web service composite inclut nécessairement une étape de sélection de Web services composants. La manière de sélectionner ces Web services dépend des exigences de choix imposées Dans la littérature, il existe deux stratégies de sélection de Web services : 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[29].

2.6.1 Sélection locale :

La stratégie de sélection locale a pour objectif de choisir le meilleur Web service pour chaque tâche individuelle a part entière en considérant des contraintes de QoS relatives a chaque tache plutôt qu'en considérant des contraintes de QoS globales exprimées pour l'ensemble des tâches. En d'autres termes, cela revient

a choisir pour chaque ensemble de candidats S_i un Web service $S_{i,j}$ qui vérifie au mieux les contraintes de QoS définies par l'utilisateur[30][31][32]. La figure suivante décrit la stratégie de sélection locale.

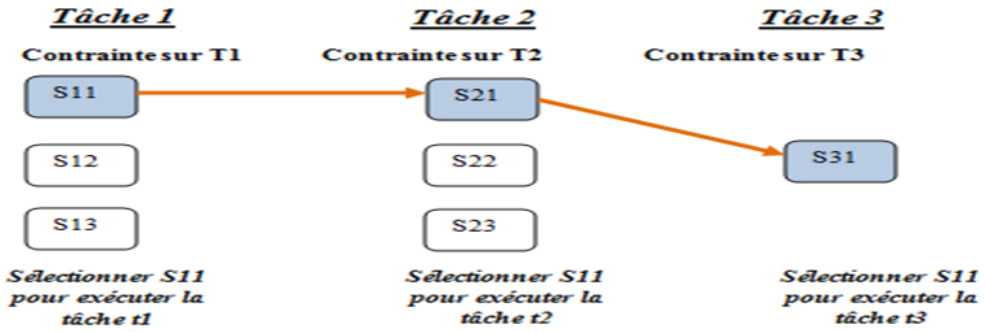


FIGURE 2.11 : Sélection locale.

2.6.2 Sélection globale :

La stratégie de sélection globale[33][34][35][36][37] a pour but de choisir la combinaison de Web services qui garantit la meilleure qualité globale en tenant compte des contraintes de QoS et des préférences globales assignées pour l'ensemble des taches. Avant d'appliquer une stratégie de sélection globale, on calcule la valeur des critères de QoS relative au service composite. Le calcul de ces critères s'appuie sur l'application de fonctions d'agrégations qui peuvent être une somme, un produit, un minimum,. . . . Ensuite, on choisit la combinaison qui offre les meilleures valeurs de QoS parmi toutes les combinaisons possibles par rapport aux contraintes et aux préférences de l'utilisateur. la figure suivante décrit la stratégie de sélection globale.

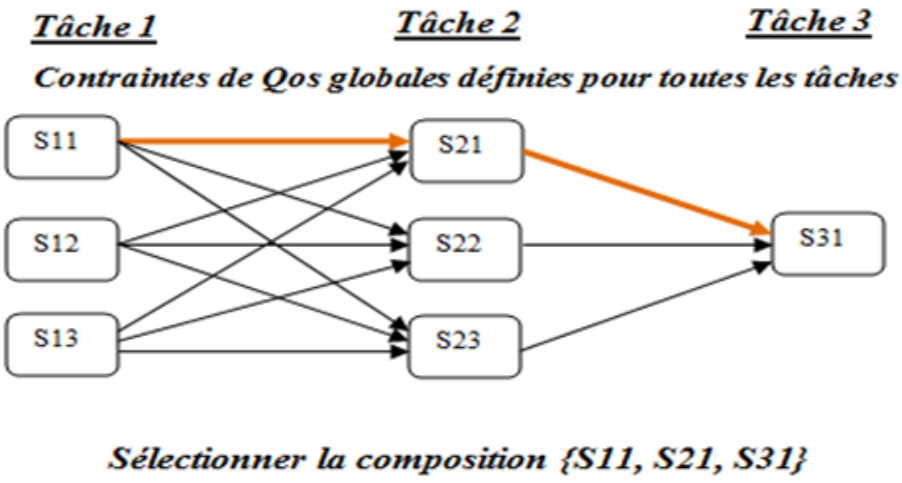


FIGURE 2.12 : Sélection globale.

2.7 Les propriétés d'un Web service

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

2.7.1 Les propriétés fonctionnelles :

Les propriétés fonctionnelles d'un service désignent les fonctionnalités que ce service peut fournir. Les propriétés fonctionnelles sont décrites dans la descrip-

tion de service en termes d'opérations, et reflètent le fonctionnement du service, comme par exemple un service de localisation[38].

2.7.2 Les propriétés non-fonctionnelles :

Les propriétés non fonctionnelles de service appelées aussi : qualités de service définissent les capacités de service a fonctionné dans de bonnes conditions en termes de disponibilité, performance, coût d'invocation, fiabilité, etc[38].

2.8 Qualité de service QoS

La qualité de service (QoS) peut être définie comme "un ensemble d'exigences dans le comportement collectif d'un ou plusieurs objectifs". Dans le contexte des technologies de l'information et multimédia, la QoS a été définie comme "l'ensemble des caractéristiques quantitatives et qualitatives d'un système multimédia, nécessaires pour atteindre la fonctionnalité requise par l'application "[39]. Nous pouvons aussi dire que la qualité de service représente l'aptitude d'un service a répondre d'une manière adéquate a des exigences, exprimées ou implicites, qui visent a satisfaire ses usagers. Ces exigences peuvent être liées a plusieurs aspects d'un service, par exemple :

1. Latence : Latence mesure le retard prévu en secondes entre le moment ou l'exécution commence et moment où l'exécution finit. La latence est calculé par :

$$\text{Latence(sol)}=\sum_{i=1}^{n} \text{lat(Ci)}$$

2. La sécurité : désigne les conditions assurant que les échanges de messages entre le client et le service soient sécurisés à savoir : la confidentialité, le cryp-

tage des messages et le contrôle d'accès.

3. Coût : le prix d'exécution est la somme d'argent qu'un demandeur de service doit payer pour exécuter une opération d'un service. Les allocataires de service annoncent directement le prix d'exécution de leurs opérations.

$$\text{coût(sol)}=\sum_{i=1}^{n}\text{coût(Ci)}$$

4. Disponibilité : La disponibilité d'un service est la probabilité que le service est accessible.

$$\text{Disp(sol)}=\log 2\prod_{i=1}^{n}\text{ disp(Ci)}$$

5. Fiabilité : La fiabilité d'un service est la probabilité qu'une demande est correctement répondue dans le délai de temps prévu maximum (ce critère calcul le taux d'erreur). La fiabilité est une mesure liée à la configuration de matériel et/ou de logiciel des services et aux connexions réseau entre les demandeurs de service et les fournisseurs.

$$\text{Fiab(sol)}=\log 2\prod_{i=1}^{n}\text{ fiab(Ci)}$$

6.Réputation : la réputation d'un service est une mesure de sa fidélité. Elle dépend principalement des expériences de l'utilisateur d'employer ce service. Les différents utilisateurs peuvent avoir le service à peu près identique de différents avis. La valeur de la réputation est définie comme rang moyen indiqué au service par des utilisateurs

$$\text{Rep(sol)}=1/2 \sum_{i=1}^{n} \text{rep(Ci)}$$

Les attributs QdS peuvent être classés en deux (02) grandes parties :

1.Attributs non mesurables : La sécurité, la réputation,.....

2. Attributs mesurables : Comme la disponibilité, la fiabilité, le Coût et le temps de réponse.

Les attributs QdS mesurables peuvent aussi être classés en deux catégories :

1. Les attributs à maximaliser : Comme la disponibilité et la fiabilité lesquels le plan résultant doit contenir les valeurs maximales.

2. Les attributs à minimiser : Comme le temps de réponse et le coût lesquels le plan résultant doit contenir les valeurs minimales

2.9 Problème d'optimisation Multiobjectif

La majorité des problèmes d'optimisation réels sont décrits a l'aide de plusieurs objectifs ou critères souvent contradictoires devant être optimises simultanément. Alors que, pour les problèmes n'incluant qu'un seul objectif, l'optimum recherché est clairement défini, celui-ci reste à formaliser pour les problèmes d'optimisation multiobjectif[40].

2.10 Problème d'optimisation bi-objectif

Lorsqu'un deux objectifs (critère) est donné, le problème d'optimisation est bi-objectif. Les problèmes bi-objectifs ont la particularité d'être beaucoup plus faciles à traiter que leur équivalent multi-objectif (3 critères ou plus). Il reste toujours un problème d'optimisation Multiobjectif.

2.11 Langages de composition de services web

2.11.1 BPEL4WS (Business Process Execution Language for Web Services) :

BEA, IBM, SAP, Siebel Systems et Microsoft ont uni leurs efforts afin de produire un langage de composition de services Web, conçu pour supporter les processus métier à travers les services Web. Ce langage, BPEL4WS (Business Process Execution Language for Web Services), est issu de la fusion de deux langages : WSFL - Web Service Flow Language d'IBM [41]et XLANG [42] de Microsoft. BPEL4WS combine les caractéristiques d'un langage de processus structuré par bloc (XLANG) avec ceux d'un langage de processus basé sur les processus métier (WSFL).
BPEL4WS [43]est basé sur XML et sur les Workflows. Ce langage distingue les processus abstraits des processus exécutables :

Le processus abstrait : ce processus spécifie les messages échangés entre les différentes parties (services Web composants) sans indiquer le comportement de chacune d'elles. parle de Business Protocol, c'est-à-dire la spécification du comportement des partenaires par rapport aux messages échangés, sans rendre public le comportement interne. Ce processus abstrait peut être relié à une composition de type chorégraphie. Les services Web communiquent alors à l'aide d'échanges de messages (partie gauche de la Figure 2.13).
La Figure 2.13 illustre la mise en oeuvre des deux types de processus (exécutable et abstrait) par BPEL4WS. La définition du processus métier est donnée par le processus exécutable. Les processus abstraits gèrent les invocations entre les différents services Web permettant l'exécution de la composition de services

définie dans le processus exécutable.

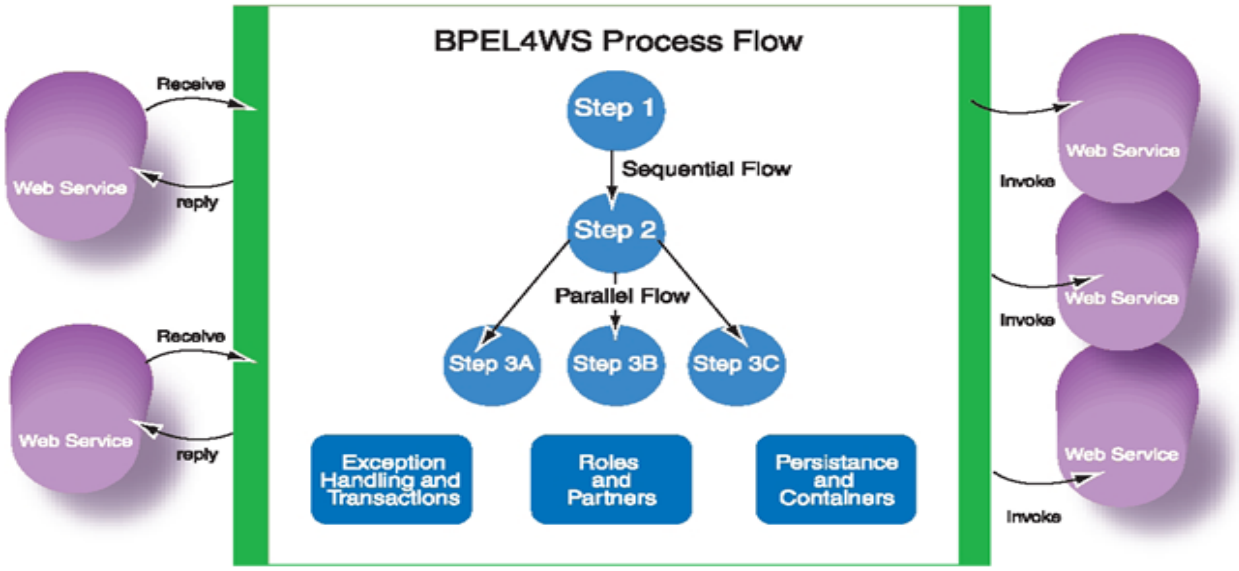


FIGURE 2.13 : Le flot de processus avec BPEL4WS.
[43]

Trois éléments permettent à BPEL4WS de gérer le flot de processus dans le processus exécutable : les transactions (Exception handling and transactions), les partenaires (Roles and Partners) et les espaces de stockage (Persistance and Containers) (voir Figure 2.13) :

Les transactions : Les transactions sont utilisées dans BPEL4WS pour gérer les erreurs et les appels d'autres services si le service appelé est indisponible ou défaillant.

Les partenaires : Les partenaires sont différents services Web invoqués dans le processus. Ils ont chacun un rôle spécifique dans un processus donné. Chaque partenaire est décrit par son nom, son rôle (en tant que service indépendant), et son rôle dans le processus.

Les espaces de stockage : Les espaces de stockage permettent la transmission des données. Le flot de processus BPEL4WS permet que ces données soient cohérentes à travers les messages échangés entre les services Web. Un message peut être un message d'appel (invoke), de réponse (reply) ou d'attente (receive).

2.12 Conclusion

L'objet de ce chapitre était de fournir une vue d'ensemble sur la composition des services Web, ainsi que ses classifications. Nous avons également présenté un ensemble de différentes approches de composition des services Web regroupés en plusieurs catégories. Pour trouver ce service composite, un processus de recherche dans un espace de recherche généralement important doit être exécuté. L'algorithme qui compose les services doit sélectionner les processus atomiques adéquats et également choisir la manière correcte de les combiner à l'aide des différentes structures de contrôle disponibles. Dans le chapitre suivant, des algorithmes méta-heuristique sont implémentés pour la composition de services Web est présenté. Les propositions tentes de minimiser le nombre de services et recherches les compositions avec le chemin d'exécution minimal.

3

Les algorithmes génétiques pour la sélection des services web

3.1 Introduction

Ces dernières années, le nombre de services Web a proliféré ; par conséquent, le nombre de services Web qui offrent les mêmes services ont été augmentés. Les différences entre les mêmes services Web sont des paramètres de qualité. Qualité de service dans les services Web se composent de divers facteurs non fonctionnels tels que coût d'exécution, temps d'exécution, disponibilité, exécution réussie taux et sécurité. Dans ce travail, le problème est formulé comme un problème d'optimisation multiobjectifs pour trouver les meilleurs Web services en fonction de qualité de services basé sur des Les algorithmes évolutionnaires.

3.2 Evolution artificielle

Les algorithmes évolutionnaires (AEs) sont des méthodes stochastiques qui simulent le processus de l'évolution naturelle dans la résolution des problèmes d'optimisation [46] .Ils reposent sur l'analogie avec l'un des principes Darwiniens les plus connus : la survie de l'individu le mieux adapté [47].Ils sont basés sur la notion de "population d'individus ", dans laquelle chaque individu représente une solution potentielle de l'espace de recherche. Ces algorithmes permettent de surmonter certains problèmes rencontrés avec les méthodes classiques. Ce sont des méthodes d'optimisation globales qui assurent la convergence vers les optima globaux malgré la présence d'optima locaux, et indépendamment de la répartition de la population initiale. Ils peuvent également être utilisés pour des problèmes discrets et discontinus. En outre, en tant qu'algorithme base de population, leur parallélisassions est aisée : il suffit de distribuer l'évaluation des fonctions coût sur autant de processeurs qu'il y a d'individus dans la population. Ils sont fréquemment utilisés à cause de leur robustesse et de leur souplesse, qui

leurs permettent d'aborder les problèmes les plus raides. Historiquement, les algorithmes évolutionnaires sont utilisés depuis les années soixante.

3.2.1 Catégories d'algorithmes évolutionnaires.

- Les algorithmes génétiques
- La programmation génétique
- Evolution Strategies
- Differential Evolution
- Evolutionary programming
- Grammatical Evolution
- Gene expression programming
- L'algorithme Learning Classifier System
- non dominated sorting genetic
- Strength Pareto Evolutionary Algorithm

Dans le reste de cette section, on présente la description du principe de fonctionnement d'un algorithme génétiques et non dominated sorting genetic algorithm :

3.3 Principe d'un algorithme génétique standard

Les AG [49] sont des algorithmes itératifs de recherche globale, fondés sur une analogie avec le monde biologique. En effet, pour un problème donné, une solution est un individu et un ensemble de solutions correspond à une population d'individus. Chaque individu peut être appelé chromosome, Les algorithmes génétiques simulent le processus d'évolution d'une population. Au sein d'une population, les individus les moins adaptés à l'environnement disparaissent.

Les autres survivent et transmettent leurs caractéristiques aux générations suivantes. On applique des opérateurs simulant les interventions sur les individus tels que le croisement et la mutation pour arriver à une population de solutions de mieux en mieux adaptée au problème. Cette adaptation est évaluée grâce à une fonction fitness. Un algorithme génétique est défini par les quatre éléments de base suivants :

- Individu/Chromosome Qui représente une solution potentielle du problème.
- Population : Un ensemble de chromosomes ou des points de l'espace de recherche.
- Environnement : L'espace de recherche.
- Fonction fitness : La fonction que nous cherchons à maximiser ou minimiser.

La figure suivante illustre l'architecture générale de cet algorithme :

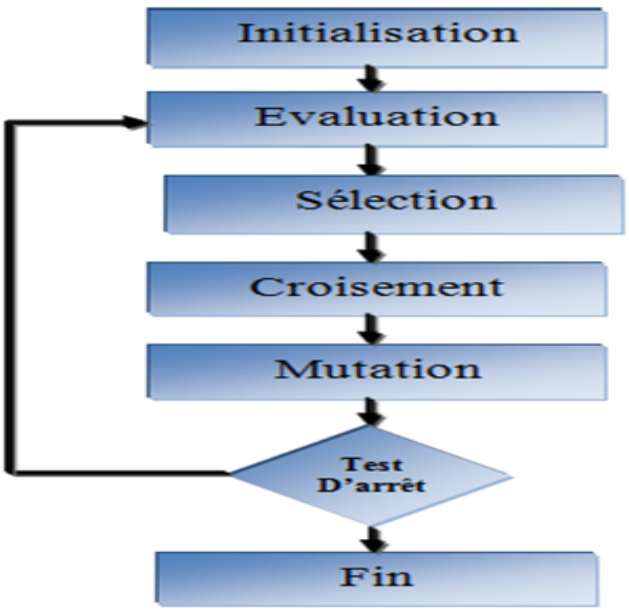


FIGURE 3.1 : L'architecture générale d'un AG [50].

Un AG est basée sur les étapes suivantes :

A. Initialisation : Une population p de N chromosomes est tirée aléatoirement ;

B. Evaluation : Chaque chromosome est évalue par la fonction fitness ;

C. Sélection : Création d'une nouvelle population de N/2 chromosomes par l'utilisation d'une méthode de sélection, nous trouvons dans la littérature plusieurs méthodes de sélection :

1.Sélection par roulette : Chaque individu a une chance d'être sélectionnée proportionnellement a sa fonction d'évaluation, donc plus les individus sont adaptés au problème, plus ils ont des chances d'être sélectionnées.

2.Sélection par tournoi : La sélection par tournoi est la méthode principalement utilisée, elle consiste à choisir un nombre de programmes aléatoirement de la population, et le gagnant de ce tournoi va être choisi.

3.La méthode élitiste : Cette méthode consiste à sélectionner les N individus dont on a Besoin pour la nouvelle Génération P' en prenant les N meilleurs individus de la population P après l'avoir triée de manière décroissante selon la fitness de ses individus.

D. Croisement : Le croisement est la transposition informatique du mécanisme qui permet, dans la nature, la production de chromosomes qui héritent partiellement des caractéristiques des parents. Son rôle fondamental est de permettre la recombinaison des informations présentes dans le patrimoine génétique de la population. Il concerne deux individus parents choisis au hasard pour construire deux enfants possédant des caractéristiques issues des leurs parents. L'opérateur de croisement opère avec une probabilité pc elle dépend de la forme de la fonction de fitness. . Plus elle

est élevée, plus la population subit des changements importants. Les opérateurs de croisement utilisés sont l'opérateur de croisement à un point et à multipoints et le croisement uniforme.

1. **L'opérateur de croisement a un point :** Il s'agit de choisir, au hasard, un point de croisement pour chaque couple de chromosomes et d'effectuer un échange des ensembles segmenté se trouvant de part et d'autre de ce point entre les deux parents. La figure suivante illustre un croisement a un point

FIGURE 3.2 : Opérateur de croisement a un point.

2. **L'opérateur de croisement en multipoints :** Dans ce cas, plusieurs points de croisement sont sélectionnés et il y a un échange des différentes parties segmenté cernées par ces points, entre les parents. La figure suivante illustre un croisement a multipoints.

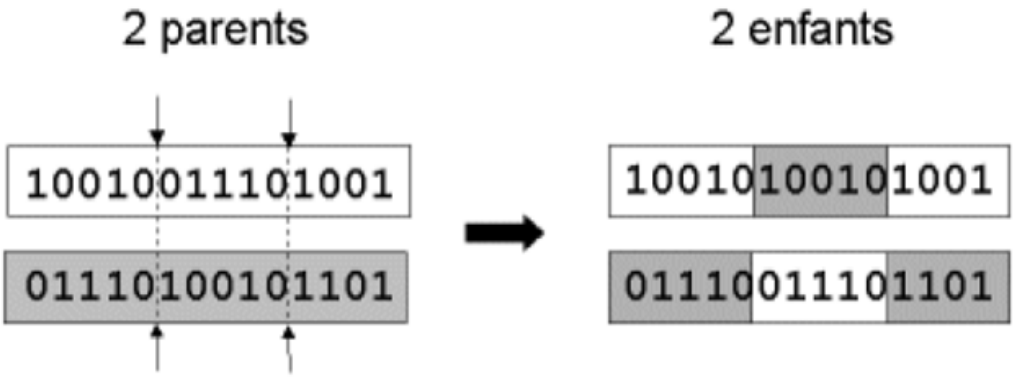


FIGURE 3.3 : Opérateur de croisement en deux points.

3.Croisement uniforme : Il opère à l'aide d'un tirage aléatoire, pour décider de la transmission de la valeur de segment à l'un ou l'autre des descendants. La figure suivante illustre le croisement uniforme.

Parent 1	1	0	0	0	1	1	1	1	1	0
Parent 2	1	1	1	1	0	0	0	0	0	1
Enfant 1	1	1	1	0	1	1	0	0	1	1
Enfant 2	1	0	0	1	0	0	1	1	0	0

FIGURE 3.4 : Opérateur de croisement uniforme.

E. Mutation : Le rôle de cet opérateur est de modifier aléatoirement, avec une certaine probabilité, la valeur d'un composant de l'individu choisi aléatoirement. Ce taux est généralement faible puisqu'un taux élevé risque de conduire à une solution sous-optimale. La figure suivante illustre un exemple de mutation

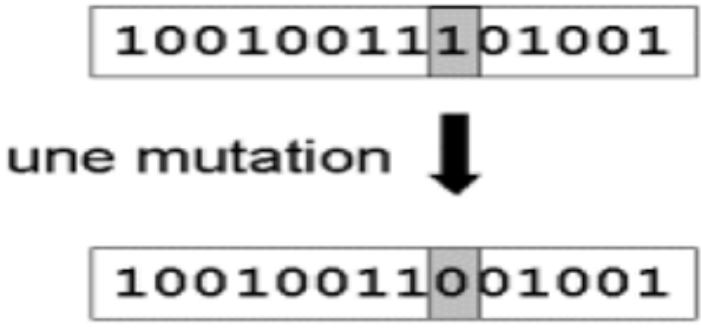


FIGURE 3.5 : Exemple de mutation.

F. Test d'arrêt : Si la condition d'arrêt n'est pas satisfaite alors retourner a la phase d'évaluation.

3.3.1 Avantages et Inconvénient

3.3.1.1 Avantages

- Ils sont basés sur des mécanismes très simples .
- Arrive à trouver de bonnes solutions sur des problèmes très complexes .
- Le traitement de solutions de manière simultanée .
- Ils sont appliqués dans les différents domaines tels que l'économie.

3.3.1.2 Inconvénients

- Le temps de calcul élève par rapport à d'autres métaheuristiques .
- Ils sont les plus souvent difficiles à mettre en œuvre .
- L'impossibilité d'être assure, même après un nombre important de Génènerations, que la solution trouvée soit la meilleure.
- On peut seulement être sur que l'on s'est approche de la solution optimale ;
- Un autre problème important est celui des optimaux locaux. En effet, lorsqu'une population évolue, a un certain instant il se peut que certains individus occupent une place importante au sein de cette population. A ce moment, il se peut que la population converge vers cet individu et s'écarte ainsi d'individus plus intéressants mais trop éloignés de l'individu vers lequel on converge.

Dans les auteurs modélisent la composition des services web avec un plan de composition qui est constituée de série, cycle, XOR-parallèle, et AND-parallèles comme modèles d'exécution.[53]

3.3.2 scénario de résolution

Le but de la composition non-fonctionnelle ou la composition à base de Qos après l'utilisation de 5 critères parmis les 13 critères de Qos est de minimiser

le coût et le temps de réponse et de maximiser la disponibilité, la fiabilité et la réputation (Optimisation multiobjectif).

3.3.2.1 Normalisation

Étant donné que chaque propriété de QoS peut être mesurée dans divers devraient être normalisés pour une évaluation appropriée.
Les propriétés QoS sont divisées en deux catégories[53] :

- Premièrement, des valeurs négatives telles que le temps de réponse et le coût d'exécution. La valeur plus élevée dans les propriétés négatives indique la qualité inférieure.

  Équation (1) sont utilisées pour normaliser les propriétés négatives, respectivement. q est la valeur de la propriété QoS.

$$q_{nrm} = \begin{cases} \frac{q_{\max} - q}{q_{\max}} - q_{\min} & q_{\max} - q_{\min} \neq 0 \\ 1 & q_{\max} - q_{\min} = 0 \end{cases} \quad (1)$$

- Deuxièmement, des valeurs positives telles que la disponibilité et la réputation. Le plus élevé dans les propriétés positives représente plus qualité et vice versa.

  Équation (2) est utilisée pour normaliser les propriétés positives, respectivement. q est la valeur de la propriété QoS.

$$q_{nrm} = \begin{cases} \frac{q - q_{\max}}{q_{\max} - q_{\min}} & q_{\max} - q_{\min} \neq 0 \\ 1 & q_{\max} - q_{\min} = 0 \end{cases} \quad (2)$$

Tache	service	cout	Treponse	disponibilité	fiabilité	réputation
tache0	service0	4.583469794960991	216.84035683685383	0.9773438426952576	0.6182161229646348	1.0
.	service1	13.99108580425399	134.67193649229193	0.7594342930450488	0.5193523743771483	4.0
.	service2	29.94781162183088	211.21857057052614	0.8402229387097692	0.8784564065055147	2.0
tache1	sevice0	11.739127224162955	292.0768245686957	0.8612010567500321	0.5142316131551348	4.0
.	sevice1	18.607882575640733	177.7534407642122	0.9521195530133317	0.6755749650614191	3.0
.	sevice2	3.254894223312732	285.9872027229679	0.714148872204752	0.5924912996351136	3.0
tache2	service0	1.8047668836326813	169.75119271667944	0.7460450578010887	0.9919082644785242	4.0
.	service1	16.854269652622257	285.3566719170691	0.9403174952472226	0.8833791539173117	2.0
.	service2	14.75072238867478	202.49085067234458	0.8016918223068966	0.6255440908605775	3.0
tache3	sevice0	12.353155728070037	295.17928835665316	0.8304630347687404	0.6556296140523944	1.0
.	sevice1	12.11381646225324	187.751786178617751124	0.9894305606248455	0.6904474033676031	1.0
.	sevice2	15.682976405943114	20.810835631773884	0.8588301194040674	0.7158895893386581	2.0

Tache	service	cout	Treponse	disponibilité	fiabilité	réputation
tache0	service0	1.0	0.0	1.0	0.2753067070885641	0.0
.	service1	0.6291007244143438	1.0	0.0	0.0	1.0
.	service2	0.0	0.06841784523486646	0.37074394304610925	1.0	0.3333333333333333
tache1	sevice0	0.44738882058985324	0.0	0.617942445874527	0.0	1.0
.	sevice1	0.0	1.0	1.0	1.0	0.0
.	sevice2	1.0	0.05326663402600416	0.0	0.4850505803637677	0.0
tache2	service0	1.0	1.0	0.0	1.0	1.0
.	service1	0.0	0.0	1.0	0.7037671301496056	0.0
.	service2	0.1397752002997711	0.7167983889508	0.2864367443849937	0.0	0.5
tache3	sevice0	0.9329424095325458	0.0	0.0	0.0	0.0
.	sevice1	1.1	0.3915446586961074	1.0	0.5777929570964406	0.0
.	sevice2	0.0	1.0	0.1784457830777617	1.0	1.0

TABLE 3.1 : les étapes de normalisation des propriétés de Qos.

3.3.2.2 Valeur d'agrégation de la propriété QoS

Généralement, les plans de composition sont constitués de séries, cycles d'exécution, XOR-parallèle et ET-parallèle. Selon la définition des propriétés QoS, l'agrégation La valeur de la composition du service Web est calculée en fonction de sa modèle de flux de travail. Les valeurs de description et d'agrégation de les modèles de workflow sont les suivants : Le modèle série qui utilise dans mon travail est un modèle d'exécution dans lequel les services sont exécutés l'un après l'autre et il n'y a pas de chevauchement entre périodes d'exécution des services Web. La figure suivant illustre ce modèle et le tableau II représente

la valeur d'agrégation de ce modèle.

- Selon le tableau suivant pour calculer la valeur d'agrégation de temps de réponse et coût d'exécution, chaque valeur de service Web doivent être ajoutés les uns aux autres[53].
- En outre, afin de calculer valeur d'agrégation de la disponibilité et du taux d'exécution réussi, les valeurs des services Web doivent être multipliées les unes par les autres, car les services Web sont indépendants les uns des autres.
- L'agrégation la valeur de la réputation est obtenue en prenant la moyenne de la réputation valeurs des services Web

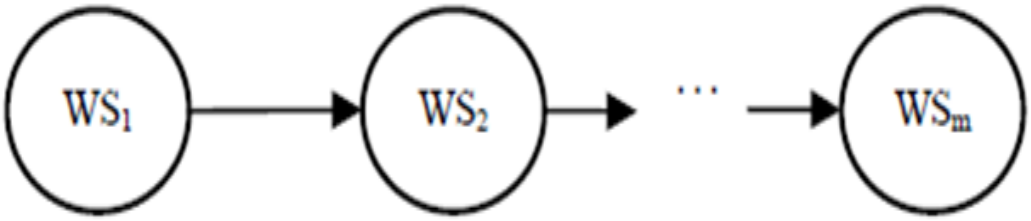


FIGURE 3.6 : Modèle de série

temps de réponse	$\sum_{i=0}^{m} SW.TR$
coût	$\sum_{i=0}^{m} SW.C$
disponibilité	$\prod_{i=0}^{m} SW.Di$
fiabilité	$\prod_{i=0}^{m} SW.Fi$
réputation	$\frac{\sum_{i=0}^{m} SW.Re}{m}$

TABLE 3.2 : L'agrégation les valeur QOS.

3.3.2.3 Constraints

Il y a deux contraintes :

- La première contrainte est que seul un service Web parmi les services Web candidats devrait être choisi pour une tâche. En d'autres termes, doit être satisfait.

Dans l'équation (3) bij est une variable de décision binaire (0 ou 1). Si $b_{i,j}$ = 1 alors le j.ème service Web candidat est sélectionné pour le i.ème processus.

$$\sum_{i=0}^{n} b_{i,j}=1 \quad 1 \leq i \leq m (3)$$

- Le deuxième contrainte est que la composition du service doit satisfaire contraintes de l'utilisateur. Pour les propriétés QoS négatives telles que l'exécution coût et temps, les valeurs d'agrégation doivent être inférieures à celles de l'utilisateur contraintes et pour les propriétés QoS positives, valeurs d'agrégation doit être supérieur aux contraintes de l'utilisateur. L'équation (4) décrit cette contrainte.Agg_d d est la valeur d'agrégation de la composition plan et Con_d est la contrainte de l'utilisateur.

$$\begin{cases} \text{Agg}_d \leq \text{Con}_d & \text{pour constraints négative} \\ \text{Con}_d \leq \text{Agg}_d & \text{pour constraints positive} \end{cases} \quad (4)$$

3.3.3 Implémentation

-La population

-Le Gène : représente un service.

- **L'individu :** représente une solution. L'individu est représenté sous forme d'un tableau d'entier de taille égale au nombre de tâche et chaque case contient le numéro du service candidat pour chaque tâche.

- **Le Chromosome :** est une composante de la solution (services composite).

  Des chromosomes sont générés. Générer chromosomes, 100 % des chromosomes sont déterminés au hasard.

-**Fonction objective :** On appelle aussi fonction d'adaptation ou fonction fitness qui associe une valeur d'adaptation à chaque individu pour le comparer à d'autres individus. Il permet, aussi, à l'algorithme génétique de déterminer l'individu sélectionne pour être reproduit ou pour déterminer s'il sera remplace. Pour chaque chromosome, la fonction fitness est calculée . Dans cet algorithme, M est un grand nombre (environ 1000) et à la fin, si la valeur d'ajustement était inférieure ou égal à 0, ce plan de composition sera un plan approprié.

```
Function FitnessFunc()
    M=Big number;
    Fit=0;
    For all web services in composition plan do
      If Aggd better that Cond
            Fit=Fit − |Aggd-Cond|;
      Else
            Fit=Fit + M * |Aggd-Cond|;
      End
    End
    Retunr Fit;
End
```

FIGURE 3.7 : Fonction fitness.

3.3.3.1 Sélection

L'opération de sélection permet de déterminer les individus les plus favorables à obtenir les meilleurs résultats.

Génération aléatoire de la population initiale :

Le principe de la génération aléatoire de la population est de choisir aléatoirement N opérations parmi les opérations des services Web qui formulent l'espace de recherche leur classement pour moins de fitness .

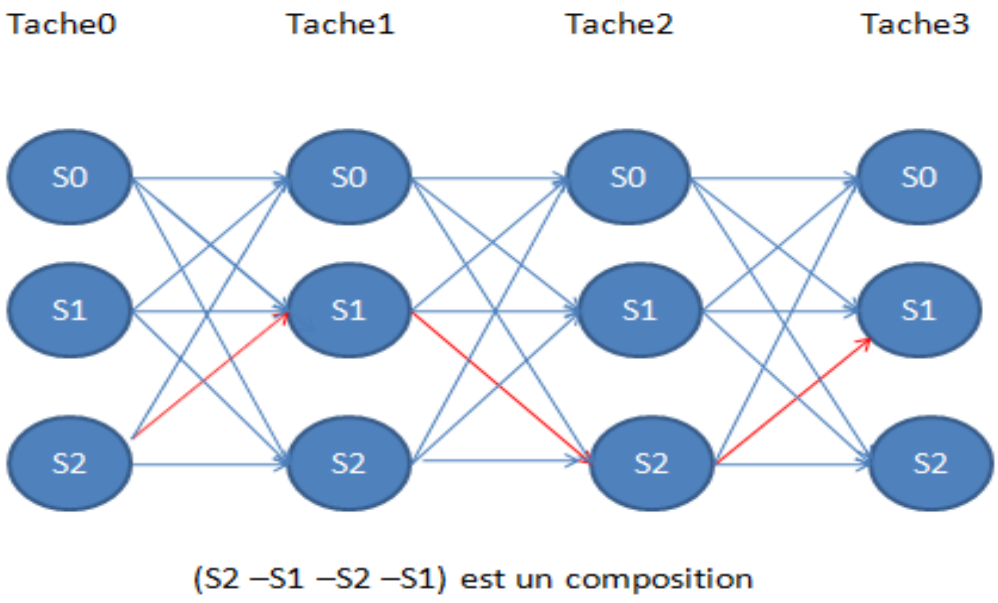


FIGURE 3.8 : Représentation des composition possibles..

3.3.3.2 Croisement

Le croisement est une fonction pour combiner deux parents ou plus chromosomes et obtenir un ou plusieurs chromosomes enfants. Nous appliquée deux types de croisement.

type 1 : comme illustré dans Figure, des gènes d'un enfant sont hérités de leurs parents alternativement.

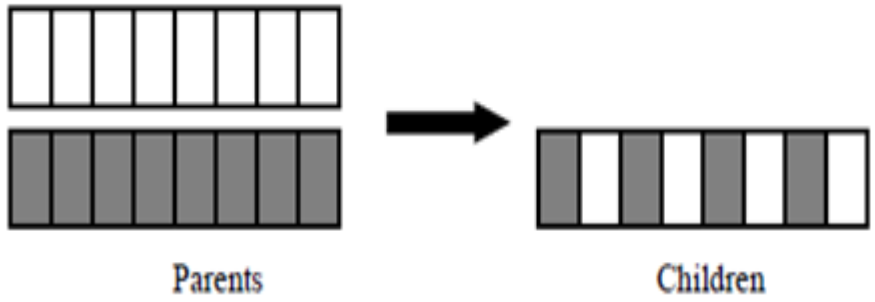


FIGURE 3.9 : croisement type 1 .

type 2 : comme représenté sur la figure , un certain pourcentage de gènes hérités d'un parent et de l'autre les gènes sont hérités de l'autre parent.

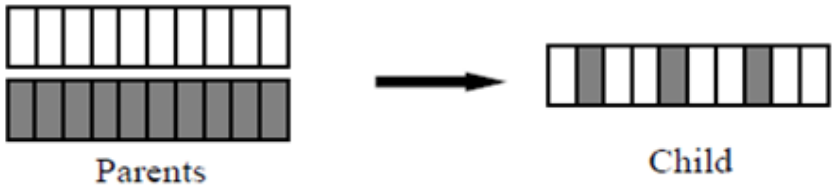


FIGURE 3.10 : croisement type 2.

1. Pour croisement opération, 20% des meilleurs chromosomes qui ont une bonne condition physique sont combinés entre eux par le type de croisement 1
2. pour le 80% restant, chaque chromosome est combiné avec un chromosome choisi au hasard appartenant à 20% des chromosomes avec fonction fitness élevée par croisement type 2.

- Pour sélectionner les chromosomes pour l'étape suivante (fonction de sélection), nombre constant des meilleurs chromosomes de l'étape précé-

dente sont sélectionnés et remplacés par les pires chromosomes dans la courante étape. Il en résulte la préservation des meilleurs chromosomes mais accélère la convergence de l'algorithme vers le local optimum.

- Pour échapper à l'optimum local, nous concevons une mutation et fonctions de chromosomes d'initialisation partielle

3.3.3.3 Mutation

En mutation, certains les gènes de certains chromosomes sélectionnés au hasard. Dans cette fonction, un nombre constant de meilleurs les chromosomes sont conservés et d'autres chromosomes sont générés encore (remplacé service par autre service de même tache)[53].

```
Function Composition_Plan_Optimizer
    T=number between 100 to 500

   Initialize chromosomes ();
   Sort chromosomes according to their Fitness function;
   Counter=0;
   While not find appropriate plan do
      Crossover ();
      Sort chromosomes according to their Fitness function;
      Selection ();
      Sort chromosomes according to their Fitness function;
      Mutation ();
      Sort chromosomes according to their Fitness function;
      Counter=counter+1;
      If (counter % T=0) Do
         Partial_initialization_Chromosome();
      End if
   End While
End Function
```

FIGURE 3.11 : corps de AG.

Le graphe suivant présente tous les étapes de l'application de l'algorithme génétique :

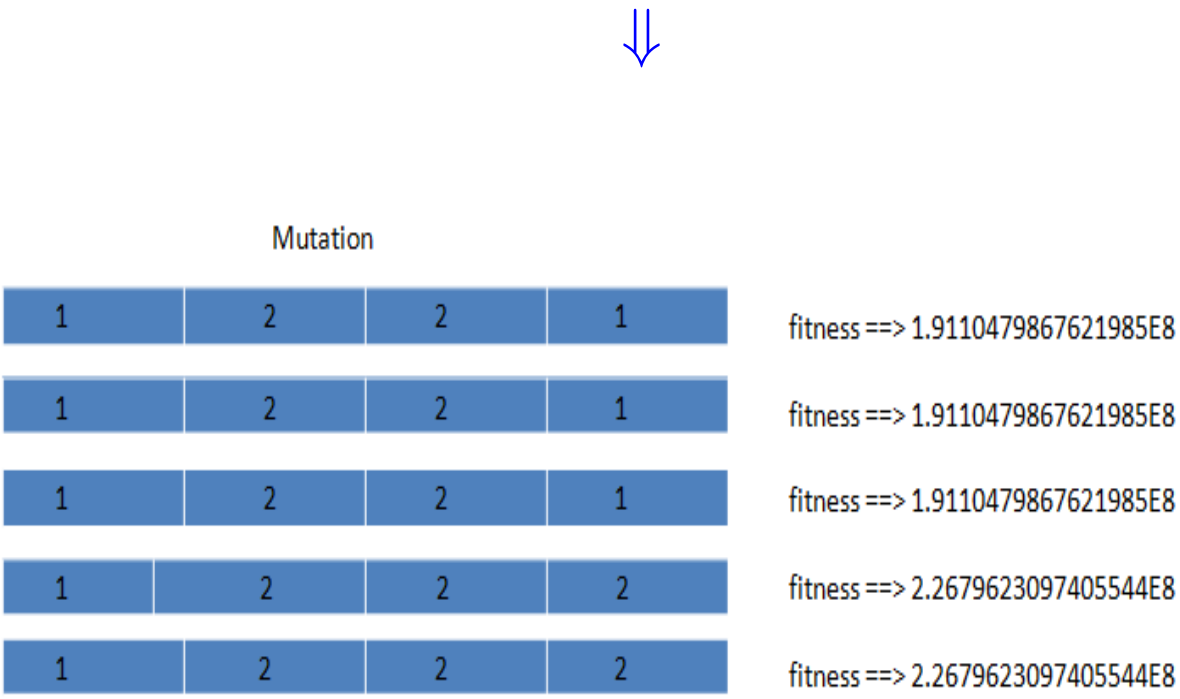
Nous effectuons une boucle tant que l'évaluation estime que la solution n'est pas optimale.

Génération i

2	1	2	1	fitness ==> 5.042019088603558E8
1	2	2	2	fitness ==> 2.2679623097405544E8
2	1	2	2	fitness ==> 5.3989334118061 91E8
1	0	1	0	fitness ==> 2.9937619026300 85E8
1	2	2	1	fitness ==> 1.9110479867621985E8

Croissement

1	2	2	1	fitness ==> 1.9110479867621985E8
1	2	2	2	fitness ==> 2.2679623097405544E8
1	0	1	0	fitness ==> 2.993761902630085E8
2	1	2	1	fitness ==> 5.042019088603558E8
2	1	2	2	fitness ==> 5.398933411806191E8
1	2	2	1	fitness ==> 1.9110479867621985E8
1	2	2	1	fitness ==> 1.9110479867621985E8
1	2	2	1	fitness ==> 1.9110479867621985E8
2	1	2	1	fitness ==> 5.042019088603558E8



3.4 L'algorithme génétique de tri non dominé 2

3.4.0.1 Définition

L'algorithme génétique de tri non dominé II (NSGA-II) est un algorithme génétique multi-objectif, proposé par Deb et al., En 2002. Il s'agit d'une extension et d'une amélioration de NSGA, qui a été proposée plus tôt par Srinivas et Deb, en 1995 . Il est basé sur les trois caractéristiques suivantes : il utilise le principe de l'élitisme, il favorise les solutions non dominées, et il utilise une variété explicite des solutions, grâce au critère de distance d'encombrement (Distance de crowding). Les principales caractéristiques de cet algorithme sont évoquées par la suite. Nous développons notamment la méthode de classification des individus, la définition de la distance de crowding, la description de la méthode de sélection et la structure de l'algorithme.

3.4.1 Déroulement de algorithme

Pour chaque chromosome, fitness est la d'agrégation dominante contraint Le processus de NSGA-II peut être divisé en cinq étapes, ce qui est illustré par la Figure suivant :

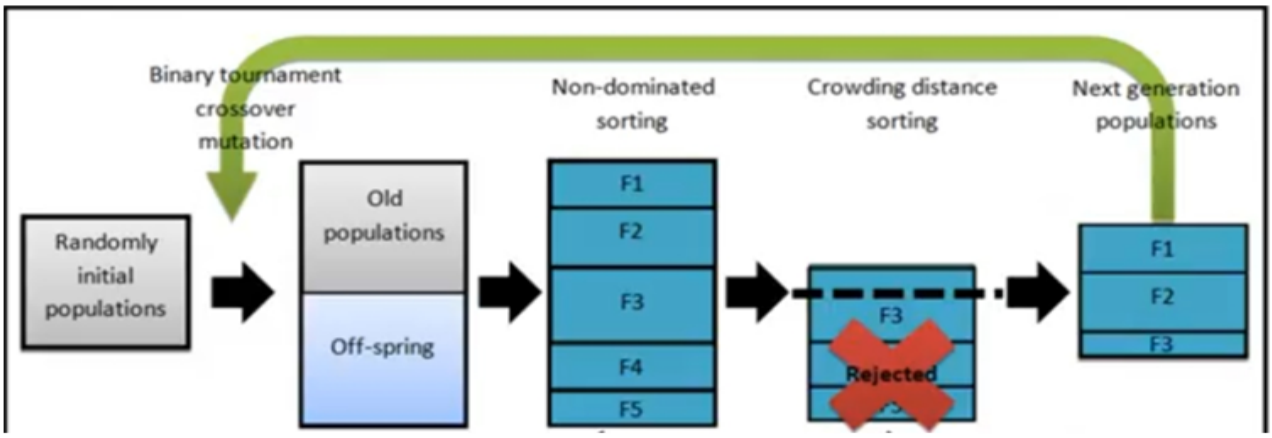


FIGURE 3.12 : Procédure de NSGA II [51].

3.4.1.1 Initialisation

Initialiser la population P0(sélection hasard des compositions).

3.4.1.2 Non-dominated sorting

la population P0 est triée par non-domination , Ensuit La procédure de tri rapide non dominé qui, lorsqu'elle est appliquée sur une population P0 renvoie une liste des fronts non dominés F c'est dire Tous les individus de la population non-dominés appartenant au front optimal (1er front) reçoivent un rang égal à 1. Ces individus sont retirés de la population et l'opération est répétée ainsi de suite pour les fronts successifs suivants en incrémentant le rang.

Si p domine q

Ajouter q à l'ensemble des solutions dominées par p

p appartient au premier front

sinon q appartient au F1

sinon si aucun solution dominates alors F1=$\{p, q.\}$ [52].

fast-non-dominated-sort(P)
for each $p \in P$
  $S_p = \emptyset$
  $n_p = 0$
  for each $q \in P$
    if $(p \prec q)$ then
      $S_p = S_p \cup \{q\}$
    else if $(q \prec p)$ then
      $n_p = n_p + 1$
  if $n_p = 0$ then
    $p_{\text{rank}} = 1$
    $\mathcal{F}_1 = \mathcal{F}_1 \cup \{p\}$
$i = 1$
while $\mathcal{F}_i \neq \emptyset$
  $Q = \emptyset$
  for each $p \in \mathcal{F}_i$
    for each $q \in S_p$
      $n_q = n_q - 1$
      if $n_q = 0$ then
        $q_{\text{rank}} = i + 1$
        $Q = Q \cup \{q\}$
  $i = i + 1$
  $\mathcal{F}_i = Q$

FIGURE 3.13 : La procédure de tri rapide non dominé [52].

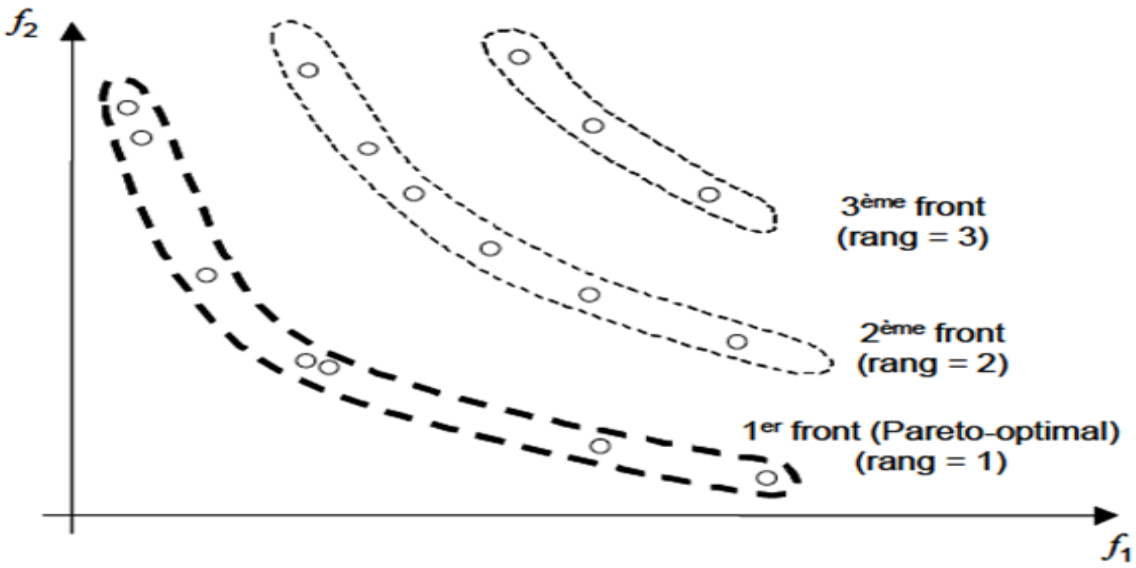


FIGURE 3.14 : Classification des individus selon le rang de Pareto.

3.4.2 Sélection

Il existe de nombreuses stratégies de sélection où les individus les plus aptes se voient attribuer plus d'exemplaires dans les prochaines générations que les autres. Ainsi, pour guider le processus de sélection, NSGA-II utilise un opérateur de comparaison, basé sur un calcul la Crowding Distance , Pour sélectionner N meilleurs parmi 2N individus, c'est-à-dire produire + 1. Cet opérateur est utilisé pour guider le processus de sélection vers les solutions Pareto-optimales. Chaque solution (i) de la population est identifiée par son rang (i_{rang}) et par sa distance de crowding ($i_{distance}$). L'opérateur défini ci-dessous permet didentifier un ordre de préférence entre deux solutions i et j : Entre deux solutions de rangs différents, on préfère la solution de plus petit rang (c'est-à dire appartenant au meilleur front). Pour deux solutions qui appartiennent au même front, on préfère la solution qui est localisée dans la région où la densité de solutions est moindre, soit l'individu possédant la plus grande valeur de distance de crowding. Cet opérateur de sélection intensifie donc la recherche des solutions Pareto-optimales mais préserve aussi

la diversité parmi des solutions équivalente.

3.4.2.1 La distance de crowding

La distance de crowding d'une solution i (ou d'un individu) se calcule en fonction du périmètre formé par les solutions du même front les plus proches de i sur chaque objectif. La figure II.9 montre une présentation à deux dimensions associée à la solution i. Le calcul de la distance de crowding nécessite avant tout le tri des solutions selon chaque objectif, dans un ordre ascendant. Ensuite, pour chaque objectif, les individus possédant les valeurs limites (la plus petite et la plus grande valeur de fonction objective) se voient associés à une distance infinie (∞). Pour les autres solutions intermédiaires, on calcule une distance de crowding égale à la différence normalisée des valeurs de fonctions objectives de deux solutions adjacentes. Ce calcul est réalisé pour chaque fonction objective. La distance de crowding d'une solution est calculée en sommant les distances correspondantes à chaque objectif[48].

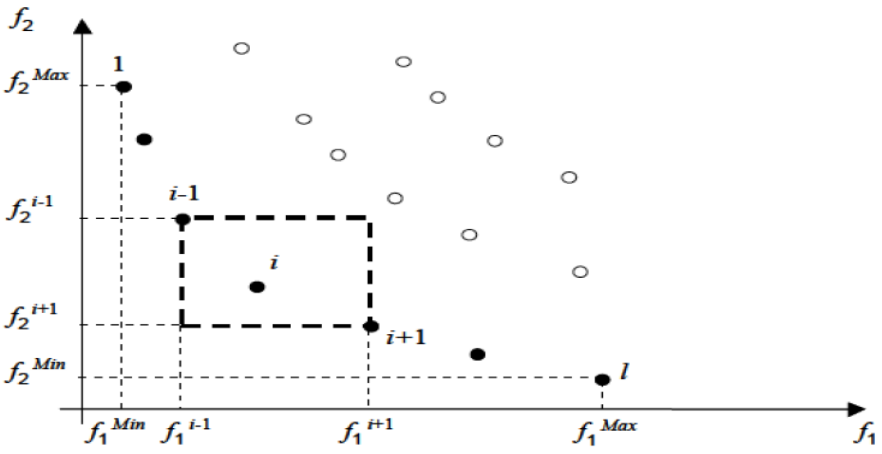


FIGURE 3.15 : Distance de crowding (Distance de surpeplement).

Est calcul comme suit :

1: **procedure** CROWDINGDISTANCE($\mathcal{F}$)
2: $N = |\mathcal{F}|$
3: **for** $i = 1 \ldots N$ **do**
4: $\mathcal{F}[i]_{\text{dist}} = 0$
5: **end for**
6: **for** $m = 1, \ldots, M$ **do**
7: SORT($\mathcal{F}, m$)
8: $\mathcal{F}[1]_{\text{dist}} = \mathcal{F}[N]_{\text{dist}} = \infty$
9: **for** $i = 2 \ldots N - 1$ **do**
10: $\mathcal{F}[i]_{\text{dist}} = \mathcal{F}[i]_{\text{dist}} + \frac{(\mathcal{F}[i+1].m - \mathcal{F}[i-1].m)}{f_m^{\max} - f_m^{\min}}$
11: **end for**
12: **end for**
13: **end procedure**

FIGURE 3.16 : Procedure distance de crowding .

3.4.3 Créez Qt + 1

Une fois que les individus appartenant à la population (Pt+1) sont identifiés, une nouvelle population enfant (Qt+1) et la recombinaison entre eux.

3.4.4 Crossover et mutation

Applique croisement et mutation (même chose de GA) Le processus continu d'une génération à la suivante, jusqu'à ce que un critère d'arrêt soit vérifié, Incrément t.

Corps de NSGA2 :

Algorithme NSGA2

entrée :Nb population,taches

sortie :resultat

for i=0 ;Nb population faire

Tant Que(resultat=vide) faire

non dominated sorting(chromosome)

crowding distance

chromosoms=next génération population

chromosoms= chromosoms $\cup partiel - initialization$

$crossover$

$mutation$

$FinTantQue$

$Finfor$

Le graphe suivant présente tous les étapes de l'application de NSGA :

Nous effectuons une boucle tant que l'évaluation estime que la solution n'est pas optimale.

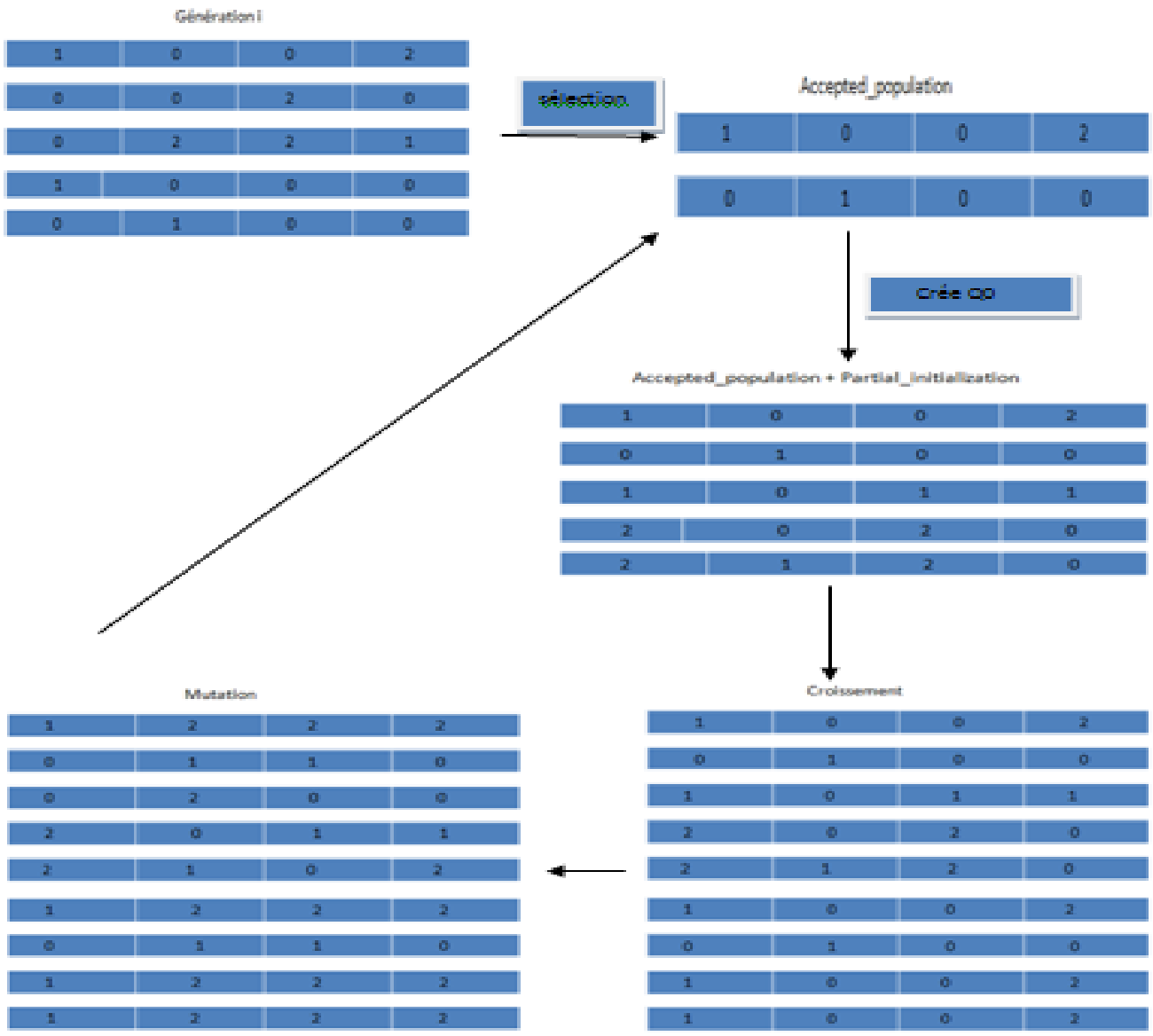


FIGURE 3.17 : L'application de l'NSGA au problème de composition .

3.5 Conclusion

Dans ce chapitre on a traité la composition de services web en adaptant les algorithmes évolutionnaires (AEs) comme étant les meilleurs algorithmes de résolution de problèmes d'optimisation multi-objectifs. Le chapitre qui ce suit présente notre travail implémenté sous forme d'une application, l'implémentation et l'environnement de travail.

4

Implémentation et expérimentation

4.1 Introduction

Dans le présent chapitre on s'intéresse à implémenter et à valider l'approches proposés pour le problème de composition de services au chapitre précédent, Nous allons commencer par présenter les différents outils techniques liés à l'implémentation, puis nous évaluons les performances de chaque approche en comparant ses résultats entre-eux.

4.2 Présentation des outils technologiques utilisés

4.2.1 Environnement de travail :

Le prototype a été développé sur un Intel Core i3 (HP) , avec une vitesse de 2.3 GHZ, doté d'une capacité mémoire de 4GB de RAM sous Windows 8.1 en utilisant des outils Open source graphiques et développés en JAVA. Nous détaillons, dans ce qui suit, chacun des outils et langages utilisés pour la manipulation des données ainsi que l'implémentation de l'interface utilisateur.

4.2.2 Le langage JAVA :

Pour le langage de programmation notre choix s'est porté sur le langage JAVA, et cela parce que JAVA est un langage orienté objet simple ce qui réduit les risques d'incohérence ; il est portable, il peut être utilisé sous Windows, sous Linux, sous Macintosh et sur d'autres plateformes sans aucune modification , enfin il 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 beaucoup d'autres fonctionnalités.

4.2.3 NetBeans :

NetBeans est un environnement de développement intégré pour développer principalement avec Java, mais aussi avec d'autres langages, en particulier PHP, C / C ++ et HTML5. Il s'agit également d'un cadre de plate-forme d'application pour les applications de bureau Java et autres. L'EDI NetBeans est écrit en Java et peut s'exécuter sur Windows, OS X, Linux, Solaris et d'autres plates-formes prenant en charge une JVM compatible. La plate-forme NetBeans permet de développer des applications à partir d'un ensemble de composants logiciels modulaires appelés modules. Les applications basées sur la plate-forme NetBeans peuvent être étendues par des développeurs tiers.

4.2.4 JavaFX :

Avec l'apparition de Java 8 en mars 2014, JavaFX devient la bibliothèque de création d'interface graphique officielle du langage Java, pour toutes les sortes d'application (applications mobiles, applications sur poste de travail, applications Web), le développement de son prédécesseur Swing étant abandonné (sauf pour les corrections de bogues). JavaFX est désormais une pure API Java (le langage Descript spécifique qui a été un temps associé à JavaFX est maintenant abandonné). JavaFX contient des outils très divers, notamment pour les médias audio et vidéo, le graphisme 2D et 3D, la programmation Web, la programmation multi-fils etc. Le SDK de JavaFX étant désormais intégré au JDK standard JavaSE, il n'ya pas besoin de réaliser d'installation spécifique pour JavaFX.

4.3 Description d'application

4.3.1 Les structures de données :

Dans notre exemple nous avons générer un nombre aléatoire de services Web dont chaque service est caractirisé par cinq critères de QOS et chaque critère est généré aléatoirement comme suit :

0<cout<30

0<temps de repense<300

0,7<disponibilité <1

0,5<fiabilité <1

1<réputation <5

Dans le contexte des deux algorithmes, on utilise les termes population, individu, chromosome et gène, tels qu'une population est constituée d'un ensemble de chromosomes et un chromosome est une suite de gènes. Dans cette section, nous décrivons les structures de données les plus importantes que nous utilisons pour représenter les paramètres génétiques de notre implémentation.

- Gène : est équivalent à un service candidat (noté service i).
- Chromosome : est équivalent à un service composite
- Population : est équivalente à une collection de services composites, appelée aussi Génération.

  Les algorithmes génétiques utilisent des opérations de mutation et de croisement. NSGA 2 utilisent des opérations de mutation et de croisement et de non dominated sorting et de Distance de surpeuplement.

4.3.2 Structuration de l'algorithme de composition :

L'implémentation en JAVA de l'approche génétique pour la composition nécessite la mise en œuvre de trois types de classes (un ensemble de classes pour la normalisation, génération des SW, taches, service web et un ensemble de classes pour les opérations de l'algorithme génétiques et un ensemble de classes pour les opérations de l'algorithme de NSGA2 et un ensemble de classes pour les opérations de view). Ces classes se divisent en quatre packages.

4.3.2.1 Le package opération (normalisation, génération des SW , taches ,service web) :

- normalisation : le rôle de cette classe est de transformer les valeurs de différents paramètres de QoS en valeurs comprises entre 0 et 1.
- génération des SW : cette classe générait les critères de QOS des services.
- taches : cette classe générait les taches.
- service web : cette classe générait les services web.

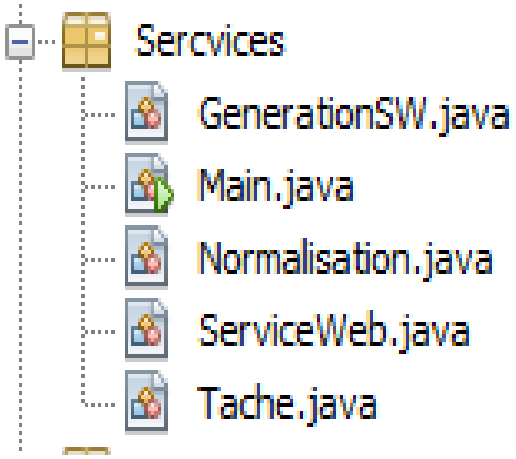


FIGURE 4.1 : Le package opération (normalisation, génération des SW , taches ,service web).

4.3.2.2 Le package des opérations de algorithme génétique :

Il contient les classes de mise en oeuvre des opérations génétiques (mutation, croisement, etc.) pour la sélection du meilleur service composite.

FIGURE 4.2 : Le package des opérations de algorithme génétique.

4.3.2.3 Le package des opérations de NSGA2 :

Il contient les classes de mise en oeuvre des opérations d'algorithme NSGA2 :

FIGURE 4.3 : Le package des opérations NSGA2.

4.3.2.4 Le package des opérations de view :

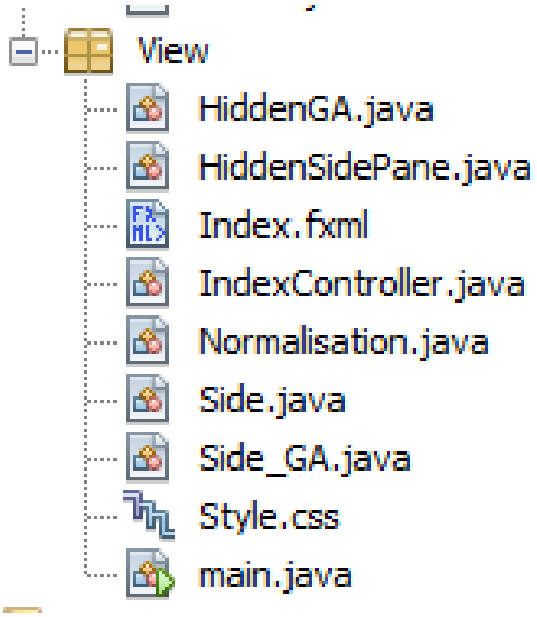


FIGURE 4.4 : Le package des opérations de view.

4.4 Présentation de l'application

L'application est présentée en sept Onglets :

— Onglet 1 : représente la phase du la base.

— Onglet 2 : représente la phase de la normalisation des servies.

— Onglet 3 : représente la phase du contraints.

— Onglet 4 : est fait le rôle de simulation par l'implémentation des algorithmes évolutionnaires utilisés.

— Onglet 5 : est fait le rôle de représentation d'algorithme génétique (afficher le graphe orienté).

— Onglet 6 : est fait le rôle de représentation d'algorithme NSGA2 (afficher le graphe orienté).

— Onglet 7 : est une comparaison entre deux algorithmes.

— Onglet 8 :graphe(histogramme)

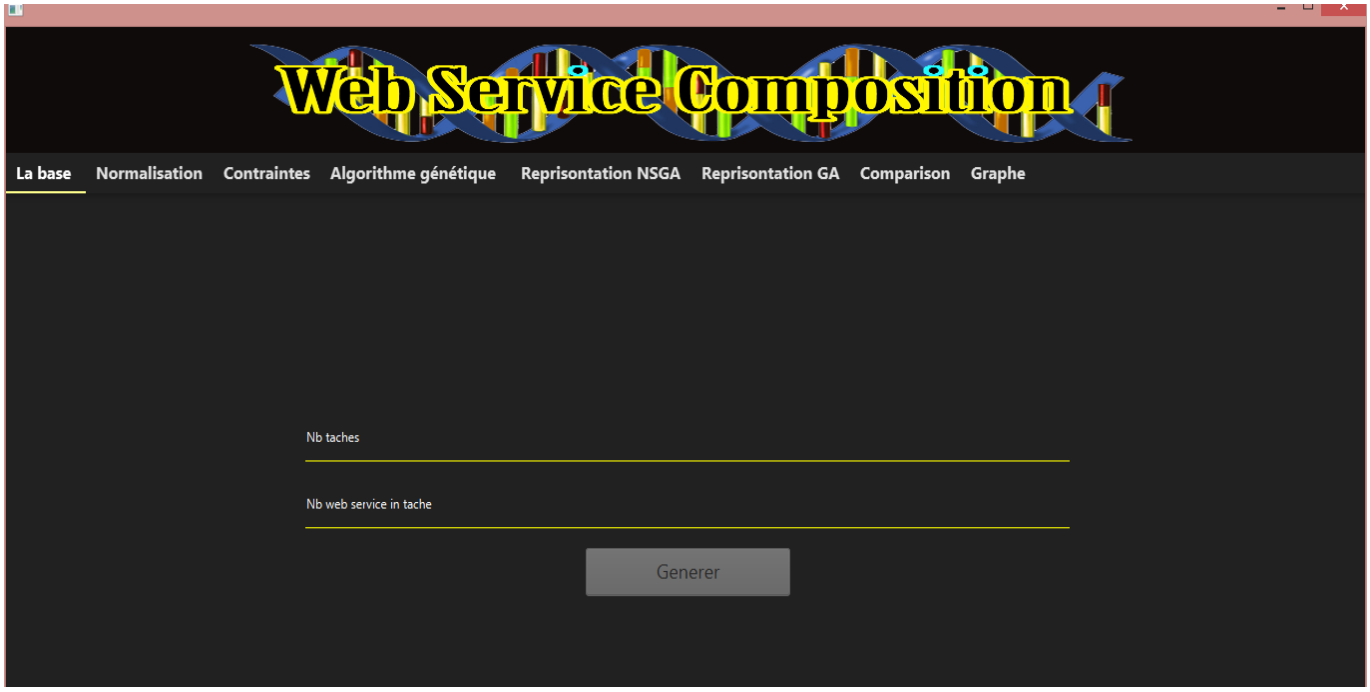
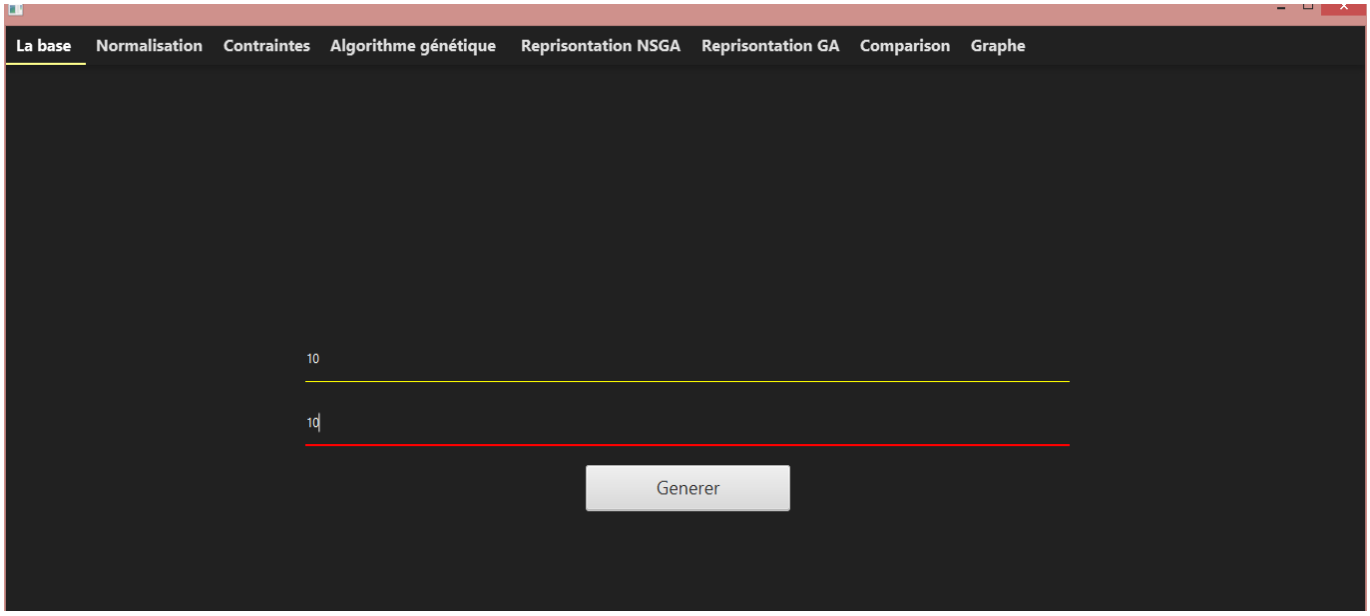
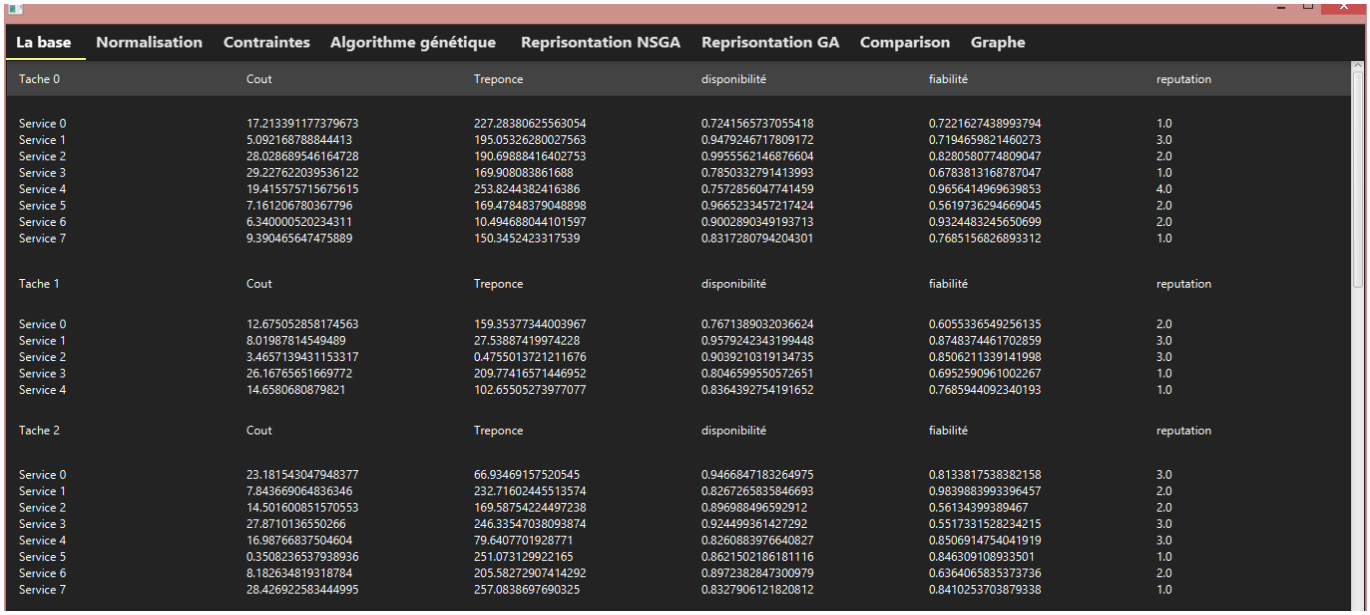


FIGURE 4.5 : Interface de l'application.

- Le premier onglet : pour le génération et le chargement de la base de données (les services web et les taches).

  on a supposé pour cette simulation qu'on a 10 services disponibles (minimum 1 maximaux 10) pour chaque tâche et le nombre de tâches a exécutés égale à 10 voici la figure :

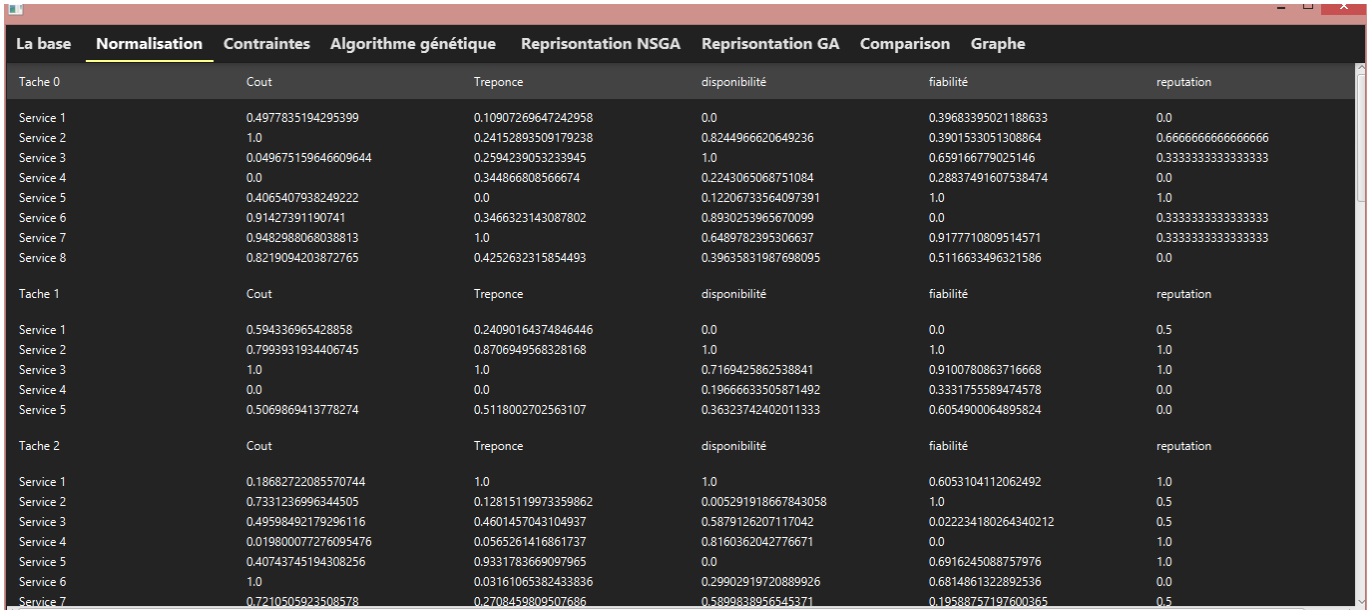




Tache 0	Cout	Treponce	disponibilité	fiabilité	reputation
Service 0	17.213391177379673	227.28380625563054	0.7241565737055418	0.7221627438993794	1.0
Service 1	5.092168788844413	195.05326280027563	0.9479246717809172	0.7194659821460273	3.0
Service 2	28.028689546164728	190.69888416402753	0.9955562146876604	0.8280580774809047	2.0
Service 3	29.227622039536122	169.908083861688	0.7850332791413993	0.6783813168787047	1.0
Service 4	19.415575715675615	253.8244382416386	0.7572856047741459	0.9656414969639853	4.0
Service 5	7.161206780367796	169.47848379048898	0.9665233457217424	0.5619736294669045	2.0
Service 6	6.340000520234311	10.494688044101597	0.9002890349193713	0.9324483245650699	2.0
Service 7	9.390465647475889	150.3452423317539	0.8317280794204301	0.7685156826893312	1.0
Tache 1	Cout	Treponce	disponibilité	fiabilité	reputation
Service 0	12.675052858174563	159.35377344003967	0.7671389032036624	0.6055336549256135	2.0
Service 1	8.01987814549489	27.53887419974228	0.9579242343199448	0.8748374461702859	3.0
Service 2	3.4657139431153317	0.4755013721211676	0.9039210319134735	0.8506211339141998	3.0
Service 3	26.16765651669772	209.77416571446952	0.8046599550572651	0.6952590961002267	1.0
Service 4	14.6580680879821	102.65505273977077	0.8364392754191652	0.7685944092340193	1.0
Tache 2	Cout	Treponce	disponibilité	fiabilité	reputation
Service 0	23.181543047948377	66.93469157520545	0.9466847183264975	0.8133817538382158	3.0
Service 1	7.843669064836346	232.71602445513574	0.8267265835846693	0.9839883993396457	2.0
Service 2	14.501600851570553	169.58754224497238	0.896988496592912	0.561343993894 67	2.0
Service 3	27.8710136550266	246.33547038093874	0.924499361427292	0.5517331528234215	3.0
Service 4	16.98766837504604	79.6407701928771	0.8260883976640827	0.8506914754041919	3.0
Service 5	0.3508236537938936	251.073129922165	0.8621502186181116	0.846309108933501	1.0
Service 6	8.182634819318784	205.58272907414292	0.8972382847300979	0.6364065835373736	2.0
Service 7	28.426922583444995	257.0838697690325	0.8327906121820812	0.8410253703879338	1.0

FIGURE 4.6 : affichage des services et des taches.

- Le deuxième onglet : Le resultat du filtrage de services.



Tache 0	Cout	Treponce	disponibilité	fiabilité	reputation
Service 1	0.4977835194295399	0.10907269647242958	0.0	0.39683395021188633	0.0
Service 2	1.0	0.24152893509179238	0.8244966620649236	0.3901533051308864	0.6666666666666666
Service 3	0.049675159646609644	0.2594239053233945	1.0	0.659166779025146	0.3333333333333333
Service 4	0.0	0.344866808566674	0.2243065068751084	0.28837491607538474	0.0
Service 5	0.4065407938249222	0.0	0.12206733564097391	1.0	1.0
Service 6	0.91427391190741	0.3466323143087802	0.8930253965670099	0.0	0.3333333333333333
Service 7	0.9482988068038813	1.0	0.6489782395306637	0.9177710809514571	0.3333333333333333
Service 8	0.8219094203872765	0.4252632315854493	0.3963583198769809 5	0.5116633496321586	0.0
Tache 1	Cout	Treponce	disponibilité	fiabilité	reputation
Service 1	0.594336965428858	0.24090164374846446	0.0	0.0	0.5
Service 2	0.7993931934406745	0.8706949568328168	1.0	1.0	1.0
Service 3	1.0	1.0	0.7169425862538841	0.9100780863716668	1.0
Service 4	0.0	0.0	0.19666633505871492	0.3331755589474578	0.0
Service 5	0.5069869413778274	0.5118002702563107	0.3632374240201133	0.6054900064895824	0.0
Tache 2	Cout	Treponce	disponibilité	fiabilité	reputation
Service 1	0.18682722085570744	1.0	1.0	0.6053104112062492	1.0
Service 2	0.7331236996344505	0.12815119973359862	0.0052919186678430 58	1.0	0.5
Service 3	0.4959849217929 6116	0.4601457043104937	0.5879126207117042	0.022234180264340212	0.5
Service 4	0.019800077276095476	0.0565261416861737	0.8160362042776671	0.0	1.0
Service 5	0.4074374519430 8256	0.9331783669097965	0.0	0.6916245088757976	1.0
Service 6	1.0	0.03161065382433836	0.2990291972088992 6	0.6814861322892536	0.0
Service 7	0.7210505923508578	0.2708459809507686	0.5899838956545371	0.19588757197600365	0.5

FIGURE 4.7 : Résultat de filttrage de services.

- La troisième onglet : Destinée pour la saisie et validation des contraintes.

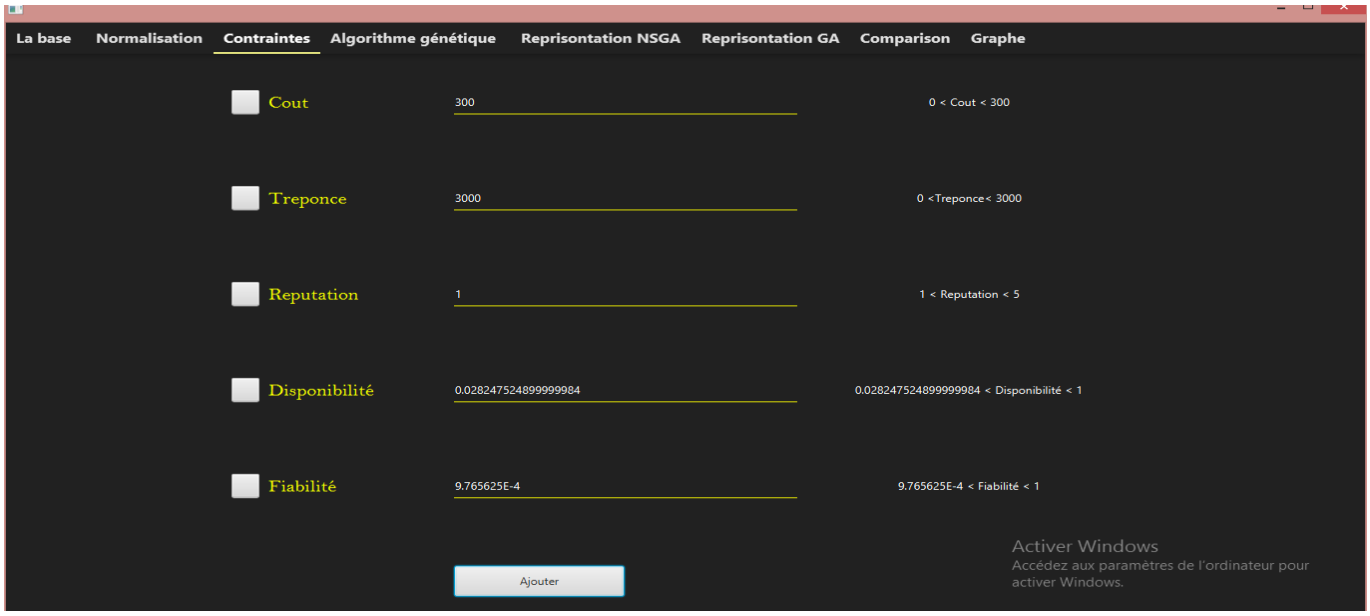


FIGURE 4.8 : validation des contraintes.

Le client à le choix de spécifier les contraintes qui peuvent grantir une bonne sélection et composition pour exécuter leur tâches, par exemple le client a spécifié que le coût doit être inférieur à 150 :

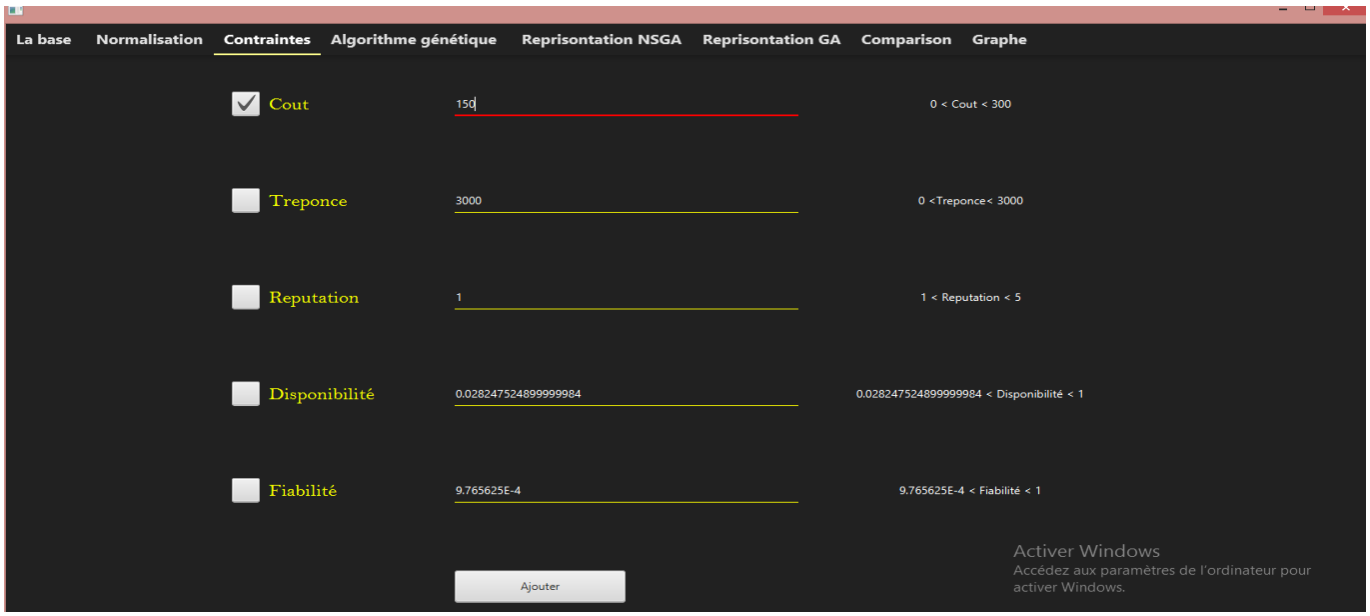
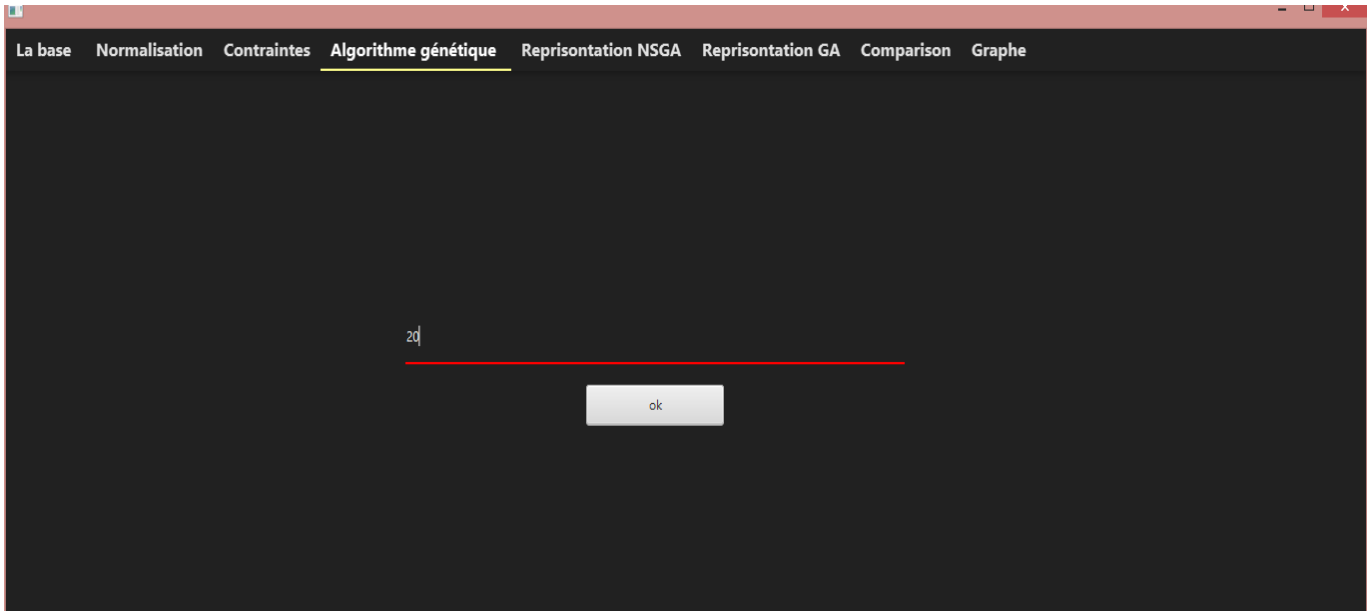


FIGURE 4.9 : spécification des contraintes par le client.

- La quatrième Onglet : Si le client choisir 20 population et appyué sur un button ok .



- chaque algorithme afficher un resultat de composition, peut être différente de l'autre.

FIGURE 4.10 : resultat de composition.

- la cinquième onglet : La figure suivante représente les étapes du dessin après l'appel d'algorithme GA :

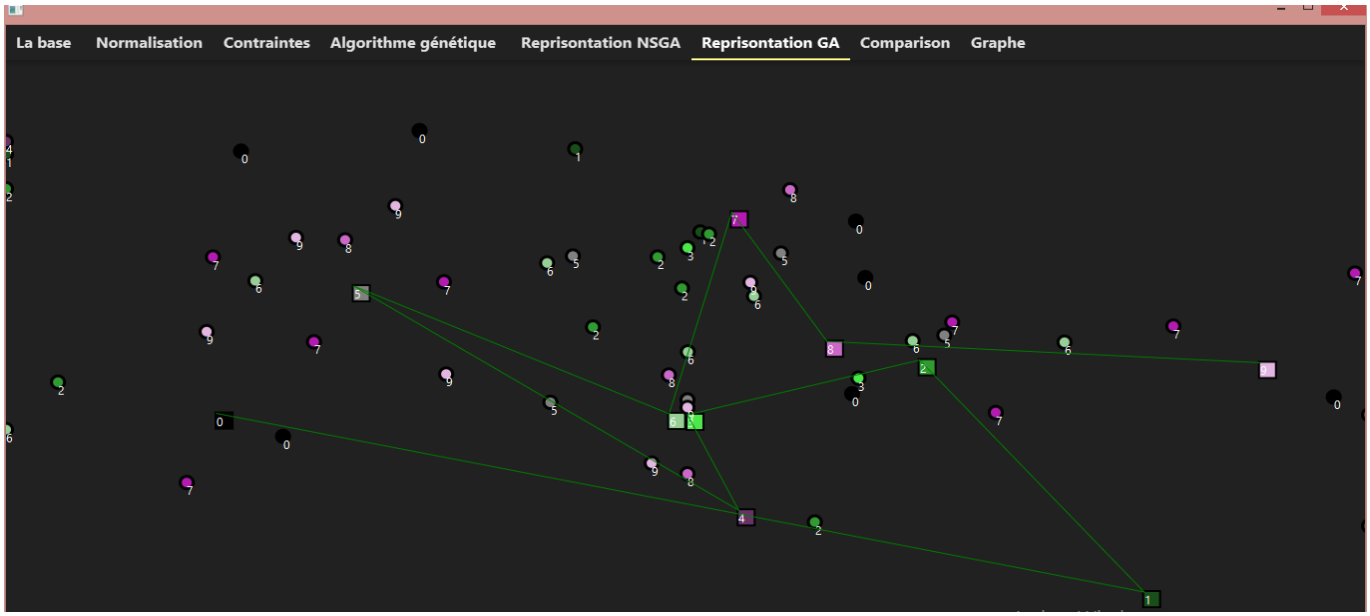


FIGURE 4.11 : Résultat du dessin par l'implémentation d'AG (services composites).

- la sixième onglet : La figure suivante représente les étapes du dessin après l'appel d'algorithme NSGA2 :

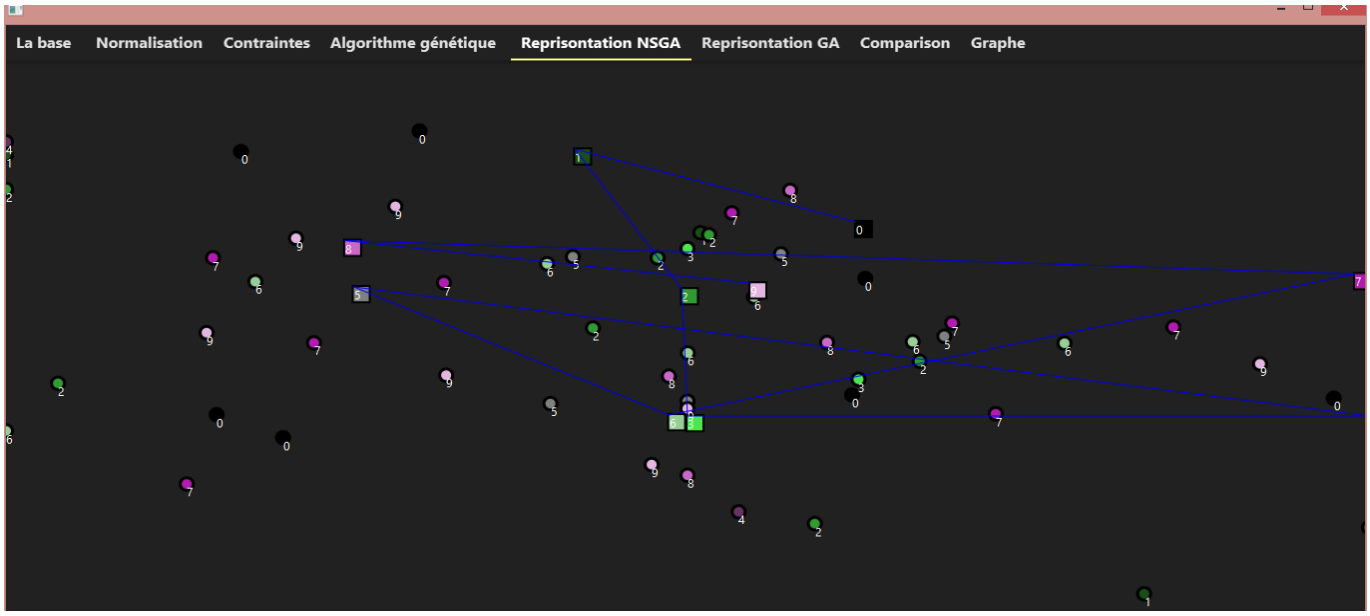
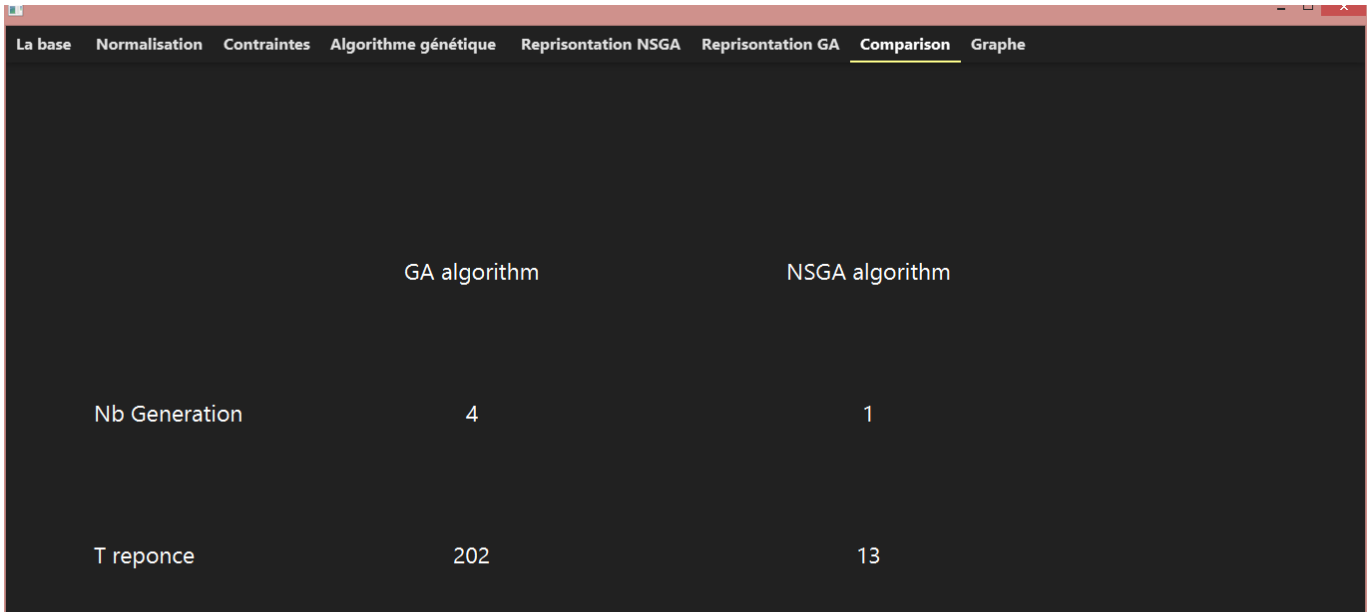
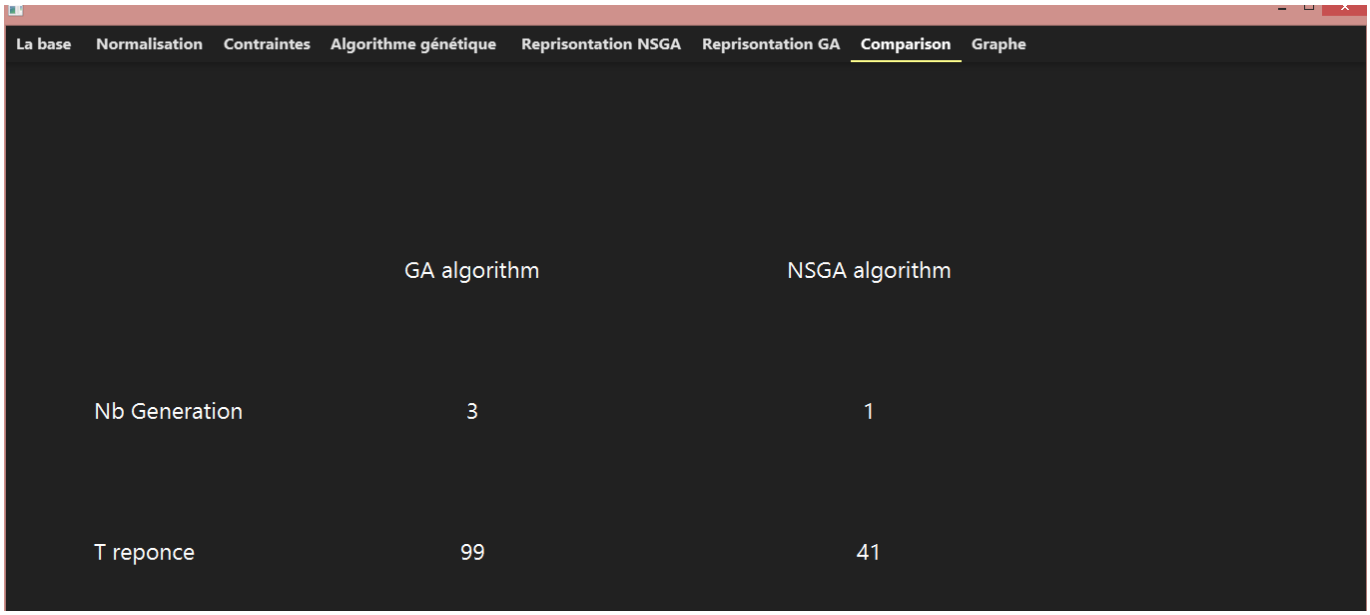


FIGURE 4.12 : Résultat du dessin par l'implémentation d'NSGA2 (services composites).

4.5 Expérimentation et évaluation

Les figures suivantes montre la différence de resultat du temps de réponse ou le temps d'exécution necessaire et nb génération pour trouver la meilleure composition pour chaque algorithme pour un nombre de tâches fixe égale à 10 et un nombre de services différent en chaque itération (Onglet 7 et 8) :





- Le premier itération pour un nombre de tâches égale à 10 et un nombre de service égale à 10 :

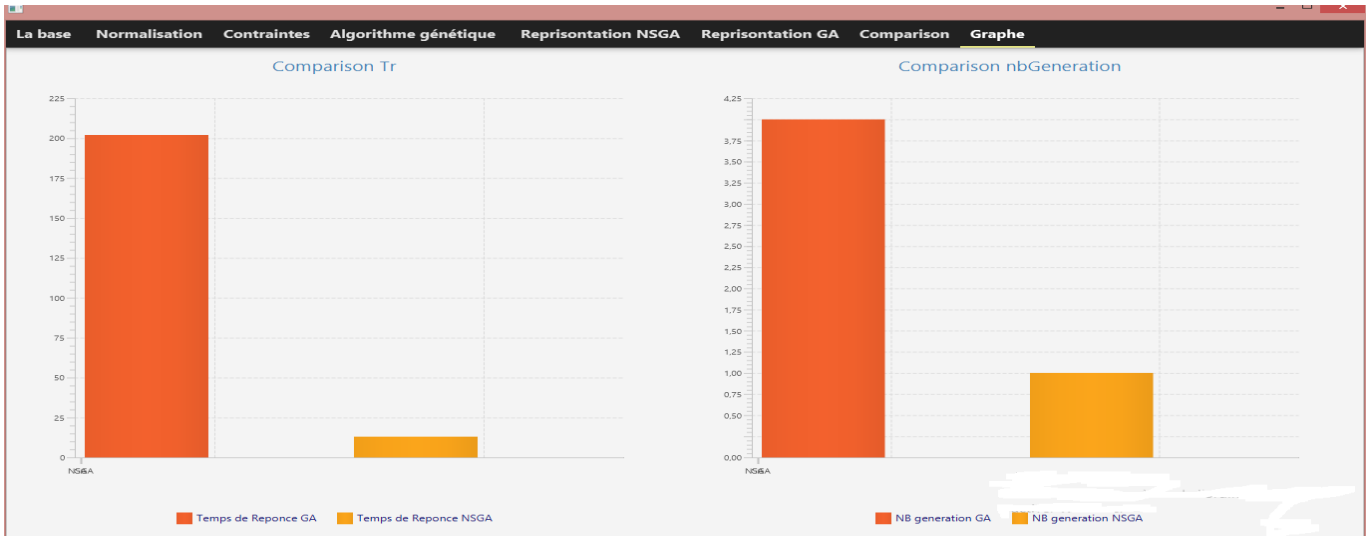


FIGURE 4.13 : Temps d'exécution necessaire et nb générations pour trouver la meilleur composition des deux algorithmes

- La deuxième itération pour un nombre de tâches égale à 10 et un nombre de service égale à 20 pour chaque tache :

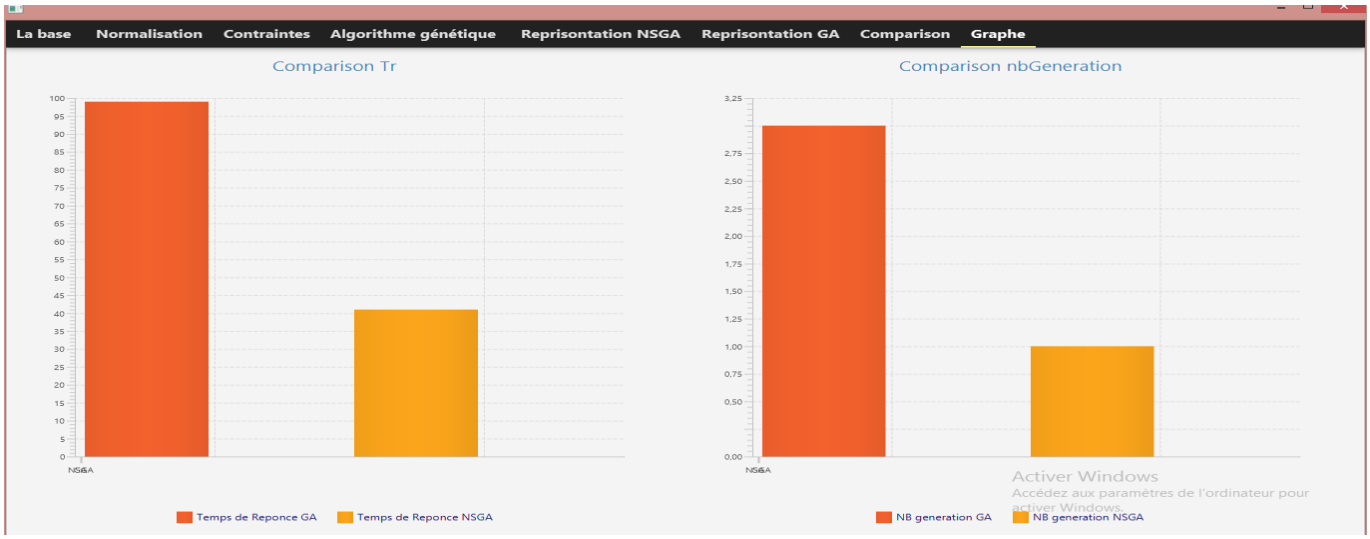


FIGURE 4.14 : Temps d'exécution necessaire et nb générations pour trouver la meilleur composition des deux algorithmes.

4.6 Conclusion

Dans ce chapitre nous avons proposé un exemple d'application pour illustrer notre travail. Nous avons présenté les résultats de la validation de nos

approches, afin d'évaluer l'efficacité de chaque méthode. D'après les résultats qu'on a obtenus, on a confirmé l'efficacité de NSGA2 et l'algorithme génétique dans le domaine de sélection et la composition de services.

Conclusion générale et perspectives

Ce projet de fin d'étude nous a permis d'instruire et améliorer nos connaissances et particulièrement le sujet de l'optimisation qui nous avons traité. Nous avons présenté dans ce mémoire les technologies liées aux services Web. Nous avons proposé aussi deux algorithmes de sélection qui se base sur Les algorithmes évolutionnaires pour l'optimisation des compositions. La première, est l'AG qui repose sur trois principes : le principe de variation, L'adaptation et le principe d'hérédité avec trois opérateurs d'évolution dans les algorithmes génétiques : La sélection, le croisement et la mutation. En suite le deuxième algorithme NSGA2 cette méthode qui permet de avant que la sélection soit entamée, les solutions sont classifiées à base de non-dominance. Tous les individus non-dominés sont classés dans une catégorie avec une valeur de fitness factice proportionnelle à la taille de la population afin de fournir une possibilité de reproduction égale pour tous les individus. Ces approches permettent de comprend le problème de la description de l'utilisateur, de l'acquisition des données nécessaires, de la conception et l'exécution du service composite optimale, et de la présentation des résultats à l'utilisateur par sélectionner les compositions de services les plus satisfaisantes, en se basant sur un groupe de cinq 05 critères de QOS : tempe de réponse, coût, fiabilité, disponibilité, réputation.

D'après notre étude et les résultats obtenus lors de l'étape d'implémentation La deuxième algorithme nous donne une solution de bonne qualité par apport à la première mais n'est pas la meilleure méthode publiée dans ce domaine.

Bibliographie

[1] https ://www.exoco-lmd.com/technologies-et-services-web/le-controle-de-qos-pour-les-services-web/action=dlattach ;attach=6627

[2] http ://deptinfo.unice.fr/twiki/pub/Linfo/Organisation20% Rapports/rapport-WebServices.pdf

[3] https ://docs.oracle.com/cd/E19644-01/817-5452/wsgoverview.html

[4] https ://web.maths.unsw.edu.au/ lafaye/CCM/web-services/soa-architecture-orientee-services.htm

[5] La sûreté de fonctionnement des services Web SOA HAMDI, Khaled http ://e-biblio.univ-mosta.dz/handle/123456789/6127 rapport-WebServices.pdf

[6] https ://pmb.univ-saida.dz/butecopac/doc_num.php ?explnum_id=994
Consulté le 13/7/2020 17 :56

[7] https ://www.ionos.fr/digitalguide/sites-internet/developpement-web/services-web
28/05/2020 23 :14

[8] http ://www.pfl-cepia.inra.fr/uploads/images/GestionDonneesImages/unites/Gd servicesWeb.pdf
6/6/2020 13 :54

[9] http ://www-igm.univ-mlv.fr/ dr/XPOSE2003/axis seng/soap.html
Consulté le 1/06/2020 00 :24

[10] http ://www.efort.com/r tutoriels/WebServices EFORT.p Consulté le 1/06/2020 00 :31

[11] https ://www.guru99.com/soap-simple-object-access-protocol.html
Consulté le 1/06/2020 00 :54

[12] https ://www.guru99.com/wsdl-web-services-description-language.html2
Consulté le 3/06/2020 00 :05

[13] https ://www.tutorialspoint.com/wsdl/wsdl introduction.htm Consulté le 4/06/2020 13 :51

[14] https ://crunchify.com/basic-wsdl-structure-understanding-wsdl-explained/ Consulté le 4/06/2020 14 :53

[15] https ://juddi.apache.org/docs/3.3/juddi-guide/html/ch01.html
Consulté le 5/06/2020 13 :53

[16] https ://www.institut-numerique.org/12-les-services-web-51f7a7f9c2daa
Consulté le 5/06/2020 15 :41.

[17] https ://www.memoireonline.com/12/08/1644/m_SOA–Definition-Utilisation-dans-le-monde-de-la-banque-et-methodologie-de-test2.html

[18] https ://www.tutorialspoint.com/webservices/what_are_web_services.htm

[19] Arenaza, N. (2006, Février). Composition semi-automatique des Web services. Projet de Master. Ecole Polytechnique Fédérale de Lausane.

[20] Benatallah, B. D.-C. (2002). Overview of Some Patterns for Architecting and Managing Composite Web Services. ACM SIGecom Exchanges.

[21] Y. Charif, e. N. (2007). Coordination in Introspective MultiAgent Systems. California, USA : Proceeding of the International Conference on Intelligent Agent Technology (IAT07).

[22] Mounir Lallali, Thèse de Doctorat : Modélisation et Test Fonctionnel de 'Orchestration deServices Web, Institut National des Télécommunications, 2009. Français

[23] Hajar OMRANA, Thèse de Doctorat : Vers une composition dynamique desServices Web : une approche decomposabilité offline, université Mohammed v AgdalRabatecole Mohammedia d'ingenieurs, 2014.

[24] Rao J and Su X. A survey of automated Web service composition methods, Proceedings of the First International Workshop on Semantic Web Services and Web Process Composition, California, USA : Springer, 2004.

[25] B. Orriens, J. Yang, and M. P. Papazoglou. Service Component : a mechanism for web service composition reuse and specialization. Journal of Integrated Design and Process Science, 8(2) : pages 13-28, 2004.

[26] P.Regnier ,Algorithmique de la Planification en IA. Cepadues-Editions, 2004

[27] P M. Ghallab, D. Nau, P, Traverso. Automated Planning : Theory and Practice.Elsevier , (Morgan Kaufmann Publishers) .(2004).

[28] A. Bekkouche. Composition des ServicesWeb Semantiques A base dAlgorithmes Genetiques. Memoire de magistère en informatique, Universite Abou-bekr Belkaid Tlemcen, Algerie, 2012.

[29] L, Zeng., B,Benatallah.,and al. QoS-Aware Middleware for Web Services Composition. IEEE transactions on software engineering. Volume 30, issue 5,(p.311-327), 2004.

[30] J. A.Parejo,P. Fernandez, and A. R. Cortes. Qos-aware services composition using tabu search and hybrid genetic algorithms. Actas de los Talleres de las Jornadas de Ingeniera del Software y Bases de Datos, 2(1), (p.55-66), 2008.

[31] M. AllamehAmiri, V. Derhami, and M. Ghasemzadeh. Qos-Based web service composition based on genetic algorithm. Journal of AI and Data MiningJournal of AI and Data Mining, 1(2), (p.63-73), Fevrier 2013.

[32] Z.Yang, C.Shang ,Al. A dynamic web services composition algorithm based on the combination of ant colony algorithm and genetic algorithm. Journal of Computational Information Systems, 6(8),(p. 2617-2622),2010

[33] D. B.Claro, P. Albers ,and J. K.Hao. Selecting web services for optimal composition. In ICWS International Workshop on Semantic and Dynamic Web Processes, Orlando-USA, Juillet 2005.

[34] F. Lecue .Optimizing qos-aware semantic web service composition. InThe Semantic Web- ISWC 2009 (p. 375-391). Springer Berlin Heidelberg, 2009.

[35] G. Canfora, M. Di Penta ,and Al. A lightweight approach for QoS-aware service composition. In Proceedings of 2nd international conference on service oriented computing (ICSOC'04), Novembre 2004.

[36] C. J. Michael and M. Gero. QoS-based selection of services : The implementation of a genetic algorithm. In Service-Oriented Architectures und ServiceOriented Computing (SOA/SOC) Bern, Switzerland,(p.359-370), Mars 2007.

[37] Y.Vanrompay, P.Rigole, and Y. Berbers. Genetic algorithm-based optimization of service composition and deployment. In Proceedings of the 3rd international workshop on Services integration in pervasive environments (p. 13-18). ACM, Juillet 2008.

[38] Driss, M., Jamoussi, Y., Moha, N., Jézéquel, J. M., Ghézala, H. H. B. (2011). Une approche centrée exigences pour la composition de services web. Revue des Sciences et Technologies de l'Information-Série ISI : Ingénierie des Systèmes d'Information, 16(2), 97-125.

[39] A. Vogel, B. Kerherve, and Al. Distributed multimedia and qos : A survey. IEEE Multi- Media, 2(2), (p.10-19), 1995

[40] Berro, A. (2001). Optimisation multiobjectifs et stratégies d'évolution en environnement dynamique (Doctoral dissertation, ANRT [diff.]). Leymann F. Web Service Flow Language 1.0.IBM Report [en ligne],2001/ <http ://www-306.ibm.com/software/solutions/webservices/pdf/WSFL.pdf>.

[41] Thatte, S., XLANG : Web Services for Business Process Design. Microsoft Specification, 2001.

[42] Andrews, T., Curbera, F., Dholakia, H., Goland, Y., Klein, J., Leymann, F., Liu, K., Roller, D., Smith, D., Thatte, S., Trickovic, I., Weerawarana, S. Business Process Execution Language for Web Services, Version 1.1. IBM Specification [en ligne], 2003. Disponible sur : <http ://www-128.ibm.com/developerworks/library/specification/ws-bpel>.

[43] Peltz, C. Web Services Orchestration and Choreography. IEEE Computer, 2003, vol.36, nř10, pp.46-52.

[44] http ://www-igm.univ-mlv.fr/ dr/XPOSE2004/woollams/definition.html

[45] http ://dspace.univkm.dz/xmlui/bitstream/handle/123456789/2019/rapport.pdf quence=1&isAllowed=y

[46] Zitzler, E., M. Laumanns, and S. Bleuler, "A Tutorial on Evolutionary Multiobjective Optimization. Metaheuristics for Multiobjective Optimisation", Workshop on Multiple Objective Metaheuristics (MOMH 2002), Springer-Verlag, Berlin, Germany, pp.3-38, 2003.

[47] Lepadatu, D., "Optimisation Des Procédés De Mise En Forme Par Approche Couplée Plans D'expériences, Éléments Finis Et Surface De Réponse", thèse de doctorat, Institut des Sciences et Techniques de lIngénieur d'Angers, 2006.

[48] Abdenour Abdelli, Optimisation multicritère d'une chaîne éolienne passive, l'institut national polytechnique de TOULOUSE, 15 octobre 2007.

[49] J. Holland. Adaptation in Natural and Arti_cial Systems. The University of Michigan Press, 1975.

[50] k.Zidi. Systeme Interactif d'Aide au Déplacement Multimodal (SIADM).

Ecole Centrale de Lille Université des Sciences et Technologies de Lille,13 Décembre 2006

[51] https ://www.youtube.com/watch ?v=RFJ3tsrFgyA&feature=share

[52] https ://www.iitk.ac.in/kangal/Deb_NSGA-II.pdf

[53] https ://ieeexplore.ieee.org/document/5734077