UNIVERSITE SAAD DAHLEB DE BLIDA

Faculté des sciences

Département d'informatique

MEMOIRE DE MASTER

En Informatique

Option : Ingénierie du Logiciel

THÈME :

Sélection à base de contexte et de QoS dans des environnements de multi-Cloud

Réalisé par

Lamara Sabrina

Laghouati Mohamed Sofiane

Encadré par

Mme Mezzi (Présidente)

Mme Daoud (Examinatrice)

Mme. MANCER Yasmine (Promotrice)

Promotion : 2021/2022

Soutenu le : 26/09/2022

Remerciments

Nous tenons à remercier le bon Dieu de nous avoir donné la force et le courage pour accomplir ce modeste travail.

Nous tenons à remercier notre promotrice Mme. MANCER,pour son aide,sa disponibilité ,sa patience et sa bienveillance.

Nous remercions sincèrement nos chers parents, nos familles et tous nos amis.

Nous remercions aussi tous les professeurs, intervenantes et toutes les personnes qui par leurs paroles, leurs écrits, leurs conseils et leurs critiques ont guidé nos réflexions et ont accepté de répondre à nos questions durant notre recherche.

Nos remerciements vont aussi aux membres de jury qui ont accepté de juger notre travail. Merci à tous.

Dédiace Sabrina

Je dédie ce modeste travail :
Aux deux personnes les plus chères à mon cœur , mon père et ma mère pour leurs amour,affection et soutien inconditionnel ainsi que leurs encouragement tout au long de mes études. Aucun hommage ne pourrait être à la hauteur de l'amour qu'ils ne cessent de m'offrir.
À mon frère le gamer Amazigh et ma petite sœur adorée Nounou.
À mes tantes Djahida, Ouahiba et Samia, ma grand-mère et mon grand-père et mon oncle Mustapha pour leurs amour et encouragement.
À mes cousines qui ont toujours été là pour moi, Ahlem et Nesrine.
À mon cher binôme Sofiane et mes chères amies Yasmine, Lylia, Nour, Roumaissa,Ikram merci d'être dans ma vie que dieu vous protège.
Sans oublier tous ceux qui m'ont enseigné tout au long de mon parcours universitaire.
À tous ceux qui me sont chers, à vous tous Merci.

Dédicaces Sofiane

Je dédie ce modeste travail :

Aux deux personnes les plus chères à mon cœur , mon père paix à son âme et ma mère pour leurs amour inconditionnel et leur soutien tout au long de mon parcours univeristaire. Je prie Dieu pour qu'il me protège ma mère. À mon frère, mon bras droit Amino et ma petite sœur adorée Moon.

À ma chère binôme Sabrina et mes amis d'être dans ma vie que dieu vous protège.

Sans oublier ceux tous qui m'ont enseigné tout au long de mon parcours universitaire.

À tous ceux qui me sont chers, à vous tous Merci.

Résumé

Le cloud computing, cette technologie devenue incontournable au cours de la dernière décennie et qui est au cœur de toutes les discussions de la tech mondiale. Son utilisation connaît une croissance énorme, de jour en jour : Toutes les grandes firmes majeures l'utilisent, Facebook, Google, Amazon, Apple, et ce pour l'utilité énorme qu'elle apporte, utilité qui est telle qu'aujourd'hui on construit et on adapte même des infrastructures dédiées spécialement au cloud pour faciliter sa maintenance et assurer son bon fonctionnement.
Cependant, cette croissance augmente la complexité des requêtes utilisateurs , ce qui diminue leur degré de satisfaction et et sature l'utilisation des ressources du cloud, d'où l'intérêt de trouver une composition de plusieurs services adéquats qui pourraient répondre à leurs demandes qui varient selon l'endroit où ils demandent le service, avec des valeurs de Qualité de Service (QoS) différentes l'une à l'autre.
Dans ce mémoire, nous avons tenté de trouver une solution à la composition des services du cloud en prenant en compte les préférences en matière de QoS mais aussi le contexte géographique de l'utilisateur en utilisant une sélection globale et locale.
Pour mettre en oeuvre la solution que nous avons proposé, nous l'avons été modélisée conformément à l'étude conceptuelle, Nous avons développé une application et l'avons comparé à d'autres solutions antérieures pour démontrer l'efficacité de notre modélisation, par rapport à d'autres. Nous avons trouvé qu'en utilisant l'algorithme génétique, combiné avec l'utilisation des vues de SQL et la décomposition en groupes nous a permi d'améliorer de manière efficace les résultats.

Mots clés :
Cloud Computing, sélection globale, sélection locale, multi-Cloud, contexte, qualité de service (QoS), composition.

Abstract

Cloud computing, this technology that has become essential over the last decade and is at the heart of all discussions in global tech. Its use is growing enormously, day by day : All the big major firms use it, Facebook, Google, Amazon, Apple, and this for the enormous utility it brings, utility which is such that today we even build and adapt infrastructures dedicated specifically to the cloud to facilitate its maintenance and ensure its proper functioning.
However, this growth increases the complexity of user requests, which decreases their degree of satisfaction and saturates the use of cloud resources, hence the interest in finding a composition of several adequate services that could meet their requests. vary depending on where they request the service, with different Quality of Service (QoS) values to each other.
In this thesis, we have tried to find a solution to the composition of cloud services by taking into account the preferences in terms of QoS but also the geographical context of the user by using a global and local selection.
To implement the solution we proposed, we modeled it in accordance with the conceptual study, We developed an application and compared it to other previous solutions to demonstrate the effectiveness of our modeling, for compared to others. We have found that using the Genetic Algorithm, combined with the use of SQL views and group decomposition has allowed us to effectively improve results.

Keywords :
Cloud Computing, global selection, local selection, multi-Cloud, context, quality of service (QoS), composition.

ملخص

الحوسبة السحابية، هذه التكنولوجيا التي أصبحت حتمية خلال الأحدث العقد والذي يقع في قلب جميع المناقشات في مجال التكنولوجيا العالمية. استخدامه ينمو بشكل هائل يومًا بعد يوم: تستخدمه جميع الشركات الكبرى، وهذا من أجل المنفعة الهائلة التي يوفرها، وهي مثل اليوم نقوم ببناء البنى التحتية المخصصة خصيصًا للسحابة لتسهيل صيانتها وضمان عملها بشكل صحيح.

ومع ذلك، فإن هذا النمو يؤدي إلى تعقيد الطلبات المستمرة لمستخدميها، بسبب كثرة هذه الأخيرة مما يطرح مشكلة في درجة وجوده الرضا عن الموارد السحابية واستخدامها، ومن ثم الاهتمام بإيجاد تركيبة لـ العديد من الخدمات المناسبة التي يمكن أن .المختلفة من واحد إلى الأخرى (QoS) تلبي مطالبهم والتي تختلف حسب الموقع حيث يطلبون الخدمة، مع قيم جودة الخدمة في هذه الأطروحة، حاولنا إيجاد حل لتكوين الخدمات السحابية مع مراعاة التفضيلات من حيث جودة الخدمة وكذلك السياق الجغرافي لـمستخدمًا باستخدام الاختيار العالمي والمحلي.

لتنفيذ الحل الذي اقترحناه، تم نموذجة هذا الحل وفقًا للدراسة المفاهمة، باستخدام لغة جافا. لقد قمنا بتطوير ملف التطبيق ومقارنته بالحلول السابقة الأخرى لإثبات فعالية نموذجنا، مقارنة بالآخرين.

الكلمات الدالة

الحوسبة السحابية، الاختيار، السحابة المتحددة، السياق، جودة الخدمة، التكوين.

Table des matières

Table des matières **9**

Table des figures **13**

Liste des tableaux **15**

1 Chapitre I : Généralités sur le Cloud Computing **20**

1.1 Introduction 20
1.2 Historique 20
1.3 Définitions du Cloud Computing 22
1.4 Les types de services dans le Cloud 23
1.4.1 Software as a service 25
1.4.1.1 Les caractéristiques du SaaS 25
1.4.2 Platform as a Service 25
1.4.3 Les caractéristiques du PaaS 26
1.4.4 Infrastructure as a Service 26
1.4.4.1 Les caractéristiques du Iaas selon 26
1.4.5 X as a Service 26
1.5 Les modèles de déploiement du Cloud 27
1.5.1 Le Cloud Privé 27
1.5.2 Le Cloud public 28
1.5.3 Le Cloud communautaire 28
1.5.4 Le cloud hybride 28
1.6 Les caractéristiques du Cloud 28
1.7 Les acteurs du Cloud 29
1.7.1 Le consommateur du Cloud (Cloud consumer) 29
1.7.2 Le fournisseur du Cloud (Cloud Provider) 30
1.7.3 L'auditeur du Cloud (Cloud Auditor) 30
1.7.4 Courtier en Cloud (Cloud broker) 30

1.7.5 Opérateur Cloud (Cloud Carrier) : 31
1.8 Les éléments constitutifs du Cloud 32
1.9 L'architecture du cloud computing 34
1.9.1 La couche matérielle 34
1.9.2 La couche infrastructure 34
1.9.3 La couche plate-forme 34
1.9.4 La couche applicative 34
1.10 Virtualisation et son rôle dans le Cloud 35
1.10.1 Virtualisation complète : 36
1.10.2 Paravirtualisation 36
1.11 Accord de niveau de service 37
1.12 Evolution du Cloud 37
1.12.1 Inter-Cloud 38
1.12.2 Pourquoi utiliser l'Inter-Cloud 38
1.12.3 Les types d'Inter-Cloud 38
1.12.3.1 Fédération de Cloud 38
1.12.3.2 Multi- Cloud 39
1.12.4 Topologies des différentes architectures de l'Inter-Cloud 39
1.13 Avantages et inconvénients du cloud 41
1.14 Conclusion 42

2 Chapitre II : La composition de service dans le cloud 43
2.1 Introduction 43
2.2 Définition de la composition de services dans le cloud 43
2.3 Présentation du problème de composition de services 44
2.4 Les défis de la composition de services 47
2.5 Cycle de vie de la composition de services 48
2.6 Processus de composition de services 49
2.7 Les étapes de composition de services 49
2.8 Les types de composition de services 50
2.9 Les méthodes de sélection 51
2.10 Les exigences fonctionnelles et non-fonctionnelles 52
2.11 Critères de Qualité de Service 52
2.11.1 Fonctions d'agrégations des valeurs de QoS 53
2.12 Optimisation multiobjective 54
2.13 Définitions d'un problème 55
2.14 Les algorithmes génétiques 55
2.14.1 Introduction 55
2.14.2 Principes 56

2.14.3 Paradigmes . . . 56
2.15 Opérateurs d'évolution . . . 57
2.15.1 Algorithme . . . 60
2.16 Conclusion . . . 60

3 Chapitre III : Travaux connexes 61
3.1 Introduction . . . 61
3.2 Etudes des travaux connexes de la composition de services . . . 61
3.3 Comparaison des différents travaux . . . 68
3.4 Conclusion . . . 70

4 Chapitre IV : Conception de la solution 71
4.1 Introduction . . . 71
4.2 Démarches de travail . . . 71
4.3 Spécification des besoins . . . 72
4.4 Processus de composition général . . . 73
4.4.1 Introduction des services par les fournisseurs . . . 74
4.4.2 Récupération et définition des taches nécessaires pour la réalisation de la requête du client . . . 74
4.4.3 Définir les services pour réaliser chaque tache . . . 74
4.4.4 Normalisation des services en fonction de la requête du client . . . 74
4.4.5 Élaboration de la sélection . . . 75
4.4.6 Élaboration de la composition . . . 75
4.5 Définition des critères de QoS . . . 76
4.6 Correspondance entre la requête et les services disponibles . . . 77
4.7 Les scenarios de composition . . . 78
4.7.1 La sélection locale Scénario 1 . . . 78
4.7.2 Exemple de scénario 1 . . . 81
4.7.3 La sélection globale Scénario 2 . . . 83
4.8 Mesure de performance du service composé . . . 83
4.9 La fonction fitness . . . 84
4.9.1 Exemple de scénario 2 . . . 87
4.10 Conclusion . . . 89

5 Chapitre V : Expérimentation 91
5.1 Introduction . . . 91
5.2 Outils et envrionnement de développement . . . 91
5.2.1 Outils . . . 91
5.3 Environnement de développement . . . 92
5.3.1 Netbeans . . . 92

5.3.2 JDK et JRE 92
5.3.3 Protégé 92
5.3.4 SQL 92
5.3.5 XAMPP 93
5.3.6 JDBC 93
5.4 Présentation des interfaces de l'application 93
5.5 Matching entre l'ontologie de description unifié et la base de données relationnelle 101
5.5.1 OBDA 102
5.6 Conclusion 107

Bibliographie **110**

*

Table des figures

1.1 Historique du Cloud Computing 22
1.2 Les types de services dans le Cloud Computing 24
1.3 Le modèle conceptuel de référence 29
1.4 Les interactions entre les différents acteurs du cloud 31
1.5 L'architecture du Cloud Computing 35
1.6 Fédération Inter-Cloud peer to peer 39
1.7 Fédération centralisée Inter-Cloud 40
1.8 Répartition des responsabilités 40
1.9 Répartition des responsabilités d'une librairie 41

2.1 La relation entre la composition de service et la sélection des clouds 45
2.2 la composition de services dans le cloud computing 46
2.3 Cycle de vie de la composition des services Web 48
2.4 Les modes de composition de services 49
2.5 Tableau montrant la différence entres les besoins fonctionnels et non fonctionnels . 52
2.6 Fonction d'agrégation pour chaque attribut de QoS 54
2.7 Image montrant les principes de l'algorithme genetique 56
2.8 Les paradigmes de l'algorithme génétique 57
2.9 Représentation binaire des individus 58
2.10 Croisiement de deux individus 59
2.11 Double Croisiement de deux individus 59
2.12 Opérateur de mutation 59
2.13 Principes de base de l'algorithme 60

3.1 Pseudo code de l'algorithme COM2 62
3.2 Ontologie de la desription du Cloud unifié 63
3.3 Pseudocode de l'algorithme de sélection 64
3.4 Pseudocode de l'algorithme de composition 65
3.5 Comparaison entre les différentes approches 69

4.1 Le cycle de l'Extreme Programming 72
4.2 Diagramme de cas d'utilisation du système 73
4.3 L'architecture globale du système de composition. 76
4.4 Types de critères de QoS utilisés. 77
4.5 Les scenarios de compositions 78
4.6 Pseudo code sélection locale 79
4.7 Encodage du problème en individu 85
4.8 Pseudocode de l'algorithme de sélection globale 86
4.9 Operateur de croisement 87
4.10 Architecture globale de la sélection globale 89

5.1 Interface d'inscription de l'application 94
5.2 Interface de login de l'admin 94
5.3 Signup client 95
5.4 Sign up fournisseur 96
5.5 Espace fournisseur 97
5.6 Données XML du service 97
5.7 Espace client 98
5.8 Espace client 98
5.9 Espace commande client 99
5.10 Contrat du client SLA 100
5.11 Espace de consultation de la commande 101
5.12 L'ontologie utilisée 101
5.13 Etablissement de la connexion entre l'ontologie et la base de données 102
5.14 Spécification des classes et des propriétés 103
5.15 Représentation des règles de matching 103
5.16 Mapping effectué 104
5.17 Instances importées 104
5.18 Courbe du temps d'execution de notre solution comparée aux deux autres (Ghezouani et Merizig en fonction du nombre de service) 105
5.19 Histogramme du temps d'execution de notre solution comparée aux deux autres (Ghezouani et Merizig en fonction du nombre de service) 106

Liste des tableaux

1.1 Avantages et inconvénients du SaaS, IaaS, et PaaS 27

3.1 Avantages et inconvénients des différentes approches 70

4.1 Exemple de la première fonctionnalité du scénario 1 81
4.2 Exemple de la deuxième fonctionnalité du scénario 1 81
4.3 Exemple de la première fonctionnalité du scénario 1 avec distance 81
4.4 Exemple de la deuxième fonctionnalité du scénario 1 avec distance 82
4.5 QoS de S1 82
4.6 QoS de S2 83
4.7 Exemple de fonctionnalité pour le scénario 2 87
4.8 Exemple de fonctionnalité pour le scénario 2 après aggrégation 88

5.1 Récapitulatif de l'approche proposée 106

Acronymes :

AwS Amazon Web Services

IBM International Business Machines

IaaS Infrastructure-as-a-service

NIS National Institute Of Standards And Technology

OBDA Ontology Based Data Access

PaaS Platform-as-a-service

QoS Quality Of Service

SLA Service Level Agreement

SQL Structured Query Language

SGBD Database Management System

SaaS Software-as-a-service

XML Extensible Markup Language

XaaS X as a Service

Introduction générale

Le Cloud Computing offre différentes options (infrastructure, plateforme, logiciel) en tant que services, il permet d'augmenter l'évolutivité des applications et de réduire le coût de l'infrastructure dans l'entreprise. Malgré ces potentiels bénéfiques, les processus de découverte, de sélection et de composition des services Cloud posent d'énormes problèmes aux consommateurs. Cela est dû au manque de standard pour la description des services Cloud.
L'importance des services Cloud est apparue lors de la crise du Covid 19 qui a frappé le monde, où l'utilisation des services Cloud a fortement augmenté en raison des mesures de quarantaine prises dans la plupart des pays du monde. Compte tenu de cela, les demandes des clients sont devenues plus complexes, invoquant plusieurs services à la fois. Ces derniers nécessitent une composition de services pour répondre aux exigences du client. Avec la disponibilité d'un grand nombre de services Cloud, beaucoup d'entre eux offrent le même service, mais avec des valeurs de qualité de service différentes.

La sélection consiste à choisir, parmi les services découverts, ceux qui répondent au mieux aux exigences de l'utilisateur sur la base des besoins fonctionnels et/ou non fonctionnels. Cette découverte est fondée sur les informations contenues dans les descriptions des services.
Plusieurs services peuvent répondre aux mêmes besoins fonctionnels, dans ce cas le meilleur service sera retourné en fonction des caractéristiques non fonctionnelles du service.

Problématique

Les requêtes des utilisateurs peuvent être non satisfaites par un seul service, où il est nécessaire d'avoir une combinaison des services pour les satisfaire. Pour cela, la sélection des services est très importante pour répondre aux demandes complexes. Le processus de sélection est un processus compliqué, surtout s'il considère les critères de QoS, ce problème est considéré comme un problème NP-Complet. La problématique de ce travail consiste à définir une solution pour la sélection de services Cloud dans des environnements multi-Cloud, en se basant sur les critères de qualité de services (QoS) et le contexte.

Objectifs

Le but de ce travail est de réaliser les objectifs suivants :

1. Etude de la sélection dans le Cloud Computing.
2. Etude comparative des solutions existantes pour la sélection dans le multi-Cloud.
3. Définition des exigences de sélection dans les environnements de multi-Cloud.
4. Définition une solution pour la sélection de services dans le multi-Cloud. La solution proposée doit garantir les objectifs suivants :
   a. Définition des scénarios de sélection.
   b. Définition des critères de qualité de services à utiliser.
   c. Elaboration des correspondances entre la requête de l'utilisateur et les services disponibles.
   d. Elaboration de la sélection globale.
5. Validation de la solution proposée par une expérimentation à travers une application Java.

Organisation générale du mémoire

Ce mémoire est subdivisé en cinq principaux chapitres, il commence par une introduction générale, Où nous avons expliqué le concept général de ce mémoire, définissant la problématique et définissant les différents objectifs pour résoudre le problème mentionné.

Le premier chapitre

Est un chapitre de généralité, il présente une vue globale sur le domaine du Cloud computing. Il comporte l'historique, les définitions des différents aspect liés au Cloud, les types du Cloud , les différents modes de déploiement ,les caractéristiques du Cloud ,ses acteurs principaux ainsi que ses avantages et inconvénients.

Le deuxième chapitre

Est consacré à l'étude de la sélection et de la composition des services dans le Cloud .Il décrit les différents concepts, les modes d'exécutions, les défis, le cycle de vie, les étapes, les méthodes et les exigences de la composition. Il comporte aussi une introduction aux concepts de l'optimisation multi-objective et algorithme génétique.

Le troisième chapitre

Est consacré à l'étude des travaux connexes, dans le but d'extraire les différents critères d'évaluation des solutions, et se termine par une étude comparative de ces travaux en mettant le point sur les aspects traités ou pas et les avantages et inconvénients de chaque travail.

Le quatrième chapitre

Est dédié à la définition de l'architecture et la modélisation de notre système, avec la spécification des différents besoins du système. En spécifiant et en détaillant les processus nécessaires de la composition, sous forme de diagramme et un pseudos codes, depuis la récupération de la requête du client à la sélection locale ou globale jusqu'à l'exécution de la composition.

Le cinquième chapitre

Est consacré à la réalisation de notre solution proposée, avec en premier lieu la présentation des outils et l'environnement de développement, par la suite la présentation des interfaces de l'application de notre solution, une expérimentations et simulation de notre solution et pour finir une discussion des résultats obtenus. Enfin, notre travail s'achève par une conclusion générale résumant les grands points qui ont été abordés ainsi que les perspectives que nous souhaitons accomplir dans le futur.

Chapitre 1

Chapitre I : Généralités sur le Cloud Computing

1.1 Introduction

Le cloud computing permet actuellement d'acheter des services informatiques à partir d'un portail web. On est en mesure de louer à partir d'une vitrine virtuelle la base nécessaire pour construire un centre des données virtuel, tels que le processeur, la mémoire et le stockage, et ajouter au-dessus le middleware nécessaire tels que les serveurs d'applications Web, bases de données et bus serveur d'entreprise, etc [1] Dans ce chapitre nous allons présenter les notions fondamentales du Cloud Computing, ses enjeux, ses évolutions et son utilité ainsi que la technologie qui le constitue et les différents acteurs du domaine. Nous devons dans un premier lieu étudier le Cloud Computing de manière générale (Définitions, Avantages, inconvénients...), dans un second lieu nous allons étudier les trois services principaux, sur lesquels le Cloud Computing repose : applicatif, plateforme, infrastructure, qui ont donné naissance aux fameux SaaS,PaaS,IaaS. Et la dernière partie de ce chapitre présente les différents avantages et inconvénients du Cloud Computing.

1.2 Historique

Quand est-ce que la révolution du cloud a elle commencé ? Selon [2] l'histoire du cloud s'est déroulé comme suit : Internet a atteint son apogée le 10 mars 2000, puis éclate progressivement au cours des semaines suivantes, avec la vente massive d'actions par des grands noms de la technologie de pointe, tels que Dell et Cisco.
Pour continuer à survivre, les entreprises doivent repenser ou ajuster leur modèle commercial et leurs offres destinées aux clients. Parmi les plus récentes, nombreuses sont celles qui décident de proposer des services basés sur Internet, plutôt que de l'utiliser comme moyen de passer

commande ou de communiquer avec les clients.
La fin des années 1990 et le début des années 2000 ont été une excellente période pour démarrer ou investir dans une entreprise en ligne. Avec le développement des architectures multi-locataires, la multiplication du haut débit et la mise en place de standards communs d'interopérabilité entre les logiciels, c'est le cadre idéal pour que le cloud computing prenne son envol. Salesforce.com a été lancé en 1999. Il a été le premier à délivrer des applications d'entreprise à partir d'un simple site web standard, accessible via un navigateur web : c'est ce que l'on appelle aujourd'hui le cloud computing.
Amazon.com a lancé Amazon Web Services en 2002. Ce nouveau service permet aux utilisateurs de stocker des données et de tirer parti des compétences d'un grand nombre de personnes pour de très petites tâches (par exemple, sur Amazon Mechanical Turk). Fondé en 2004, Facebook révolutionne la façon dont les utilisateurs communiquent et stockent leurs propres données (photos et vidéos), faisant involontairement du cloud un service personnel.
En 2006, Amazon développe son service cloud. Tout d'abord, Elastic Compute Cloud (EC2), qui permet aux utilisateurs d'accéder aux ordinateurs et d'y exécuter leurs propres applications, le tout dans le cloud. Le deuxième service lancé est Simple Storage Service (S3). Il a présenté aux clients et à l'industrie le modèle de paiement à l'utilisation, qui est désormais une pratique courante.
Salesforce.com lance ensuite Force.com en 2007. Cette plate-forme en tant que service (PaaS) permet aux développeurs de concevoir, de stocker et d'exécuter toutes les applications et tous les sites Web nécessaires à leurs activités sur le Cloud.
Lancé en 2009, Google Apps permet à ses utilisateurs de créer et de stocker des documents entièrement dans le cloud. Plus récemment, les entreprises de cloud computing ont cherché à améliorer encore l'intégration de leurs produits. En 2010, salesforce.com a lancé Cloud Database avec Database.com pour les développeurs, marquant l'évolution des services de cloud computing pouvant être utilisés sur n'importe quel appareil, exécutés sur n'importe quelle plate-forme et écrits dans n'importe quel langage de programmation.

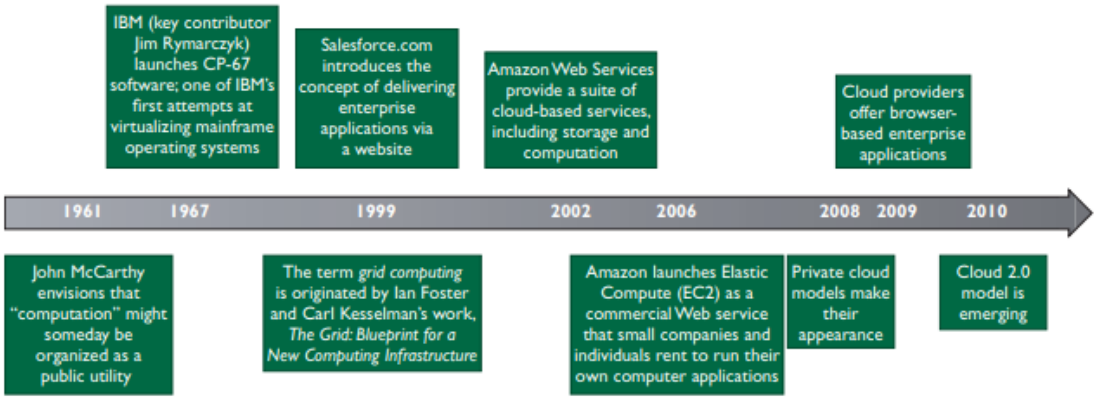

FIGURE 1.1 – Historique du Cloud Computing

[3]

1.3 Définitions du Cloud Computing

Le terme informatique en nuage " cloud computing " est un terme très vaste qui nous mènes à trouver une multitude de définitions. Et parmi ces définitions, nous avons :

Selon IBM (International Business Machines Corporation) : "Cloud computing is on-demand access, via the internet, to computing resources—applications, servers (physical servers and virtual servers), data storage, development tools, networking capabilities, and more—hosted at a remote data center managed by a cloud services provider (or CSP).. " Autrement dit Le cloud computing est un accès à la demande, via Internet, à des ressources informatiques (applications, serveurs (serveurs physiques et serveurs virtuels), stockage de données, outils de développement, capacités de mise en réseau, etc.) hébergées dans un centre de données distant géré par un service cloud. fournisseur (ou CSP). [1]

Selon AWS (Amazon Web Services) : " Le Cloud Computing est la mise à disposition de ressources informatiques à la demande via Internet, avec une tarification en fonction de votre utilisation. Au lieu d'acheter, de posséder et de gérer des serveurs et des centres de données physiques, vous pouvez accéder à votre guise aux services technologiques, tels que la puissance de calcul, le stockage et les bases de données, d'un fournisseur Cloud tel qu'Amazon Web Services (AWS 3) " [2]

Selon Microsoft : "Le cloud computing est la fourniture de services informatiques (notamment des serveurs, du stockage, des bases de données, la gestion réseau, des logiciels, des outils d'analyse, l'intelligence artificielle) via Internet, dans le but d'offrir une innovation plus rapide, des ressources flexibles et des économies d'échelle. En règle générale, vous payez uniquement les services cloud que vous utilisez (réduisant ainsi vos coûts d'exploitation), gérez votre infrastructure plus efficacement et adaptez l'échelle des services en fonction des besoins de votre

1. https ://www.ibm.com/cloud/learn/cloud-computing
2. https ://aws.amazon.com/fr/what-is-cloud-computing/

entreprise " [3]

Selon NIST(National Institute of Standards and Technology) : " Le cloud computing est un modèle permettant un accès réseau omniprésent, pratique et à la demande à un pool partagé de ressources informatiques configurables (par exemple, réseaux, serveurs, stockage, applications et services) qui peuvent être rapidement provisionnés et publiés avec un minimum d'effort de gestion ou d'interaction avec le fournisseur de services. Ce modèle cloud favorise la disponibilité et se compose de cinq caractéristiques essentielles, trois modèles de service et quatre modèles de déploiement."[4]

Nous constatons donc que le cloud computing est un modèle qui permet un accès facile à des ressources informatiques. Il est basé sur le concept de la virtualisation qui est une technologie informatique qui simule les fonctionnalités matérielles pour créer des services informatiques basés sur logiciel comme des applications, des serveurs, des espaces de stockage et des réseaux. Après avoir définit le concept du cloud computing , nous allons définir dans les prochaines sections les modèles de services et de déploiements, les acteurs du cloud , ses caractéristiques , son architecture et pour finir ses avantages et inconvénients.

1.4 Les types de services dans le Cloud

Dans le cloud computing les fonctionnalités sont offertes aux consommateurs comme des services. Ces services s'organisent selon la nature du service livré en trois grands groupes(niveaux) : le niveau infrastructure (IaaS), le niveau plateforme (PaaS) et le niveau application (SaaS), la figure suivante (figure 2) montre ces trois niveaux

Les principaux concepts à savoir service, plateforme et infrastructure, sont définit par [6] comme suit :

• **Service** : Un service est un mécanisme qui est capable de fournir une ou plusieurs fonctionnalités, qu'il est possible d'utiliser dans le respect des restrictions et des règles définies par le fournisseur et à travers une interface.

• **Plateforme** : Une plate-forme est un système informatique fondamental qui comprend l'équipement matériel, les systèmes d'exploitation, et dans certains cas, des outils de développement d'applications et des interfaces utilisateurs sur lesquels les applications peuvent être déployées et exécutées.

• **Infrastructure** : L'infrastructure fait référence aux composants physiques qui sont nécessaires à un système pour exécuter ses fonctionnalités. Dans les systèmes d'information, ces composants peuvent contiennent des processeurs, du stockage, d'équipement de réseau et dans certains les systèmes de gestion de base de données et les systèmes d'exploitation.

3. https ://azure.microsoft.com/fr-fr/resources/cloud-computing-dictionary/what-is-cloud-computing/

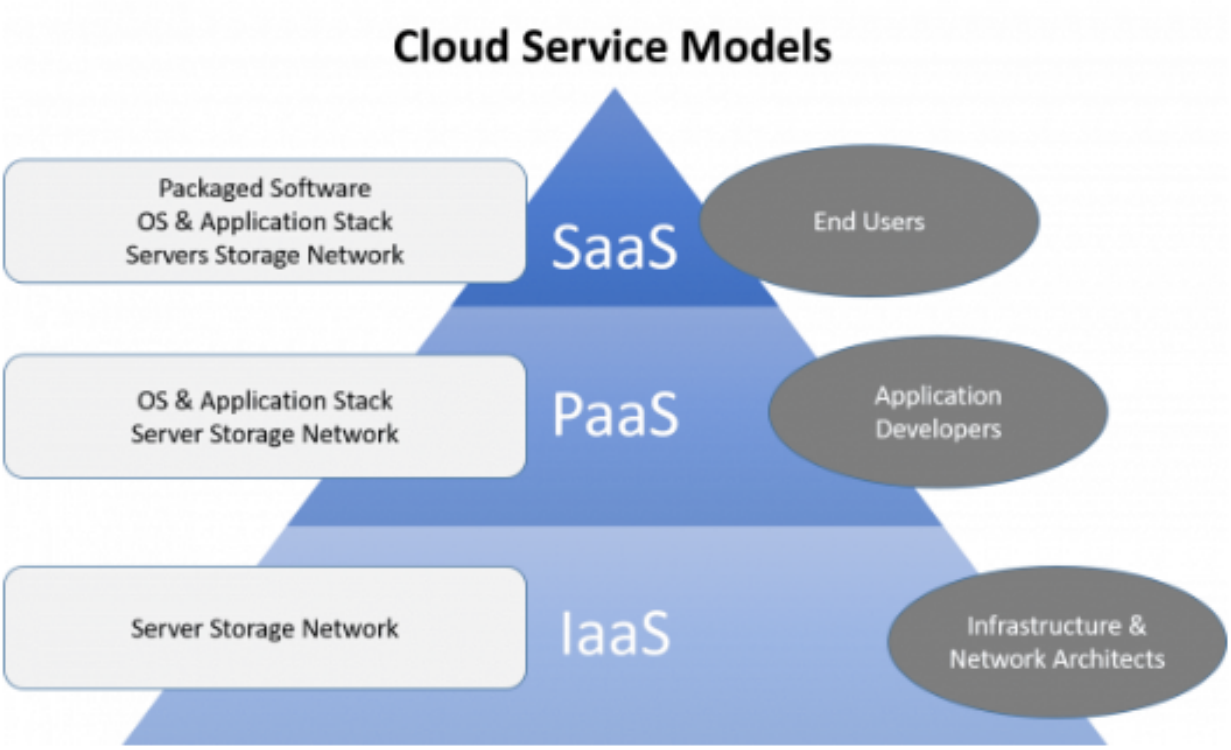

FIGURE 1.2 – Les types de services dans le Cloud Computing [5]

Ainsi que les différents types de services ont été definit par [7] comme suit :

1.4.1 Software as a service

SaaS est un logiciel ou une application qui est exécuté sur l'infrastructure d'un fournisseur reconnue comme un service à condition que le consommateur dispose d'une autorisation d'accès limitée ,la fourniture se fait via un client léger (par exemple, un navigateur Web) ou une interface de programme pour envoyer des données et recevoir des résultats. Le consommateur ne connaît pas l'infrastructure du fournisseur d'applications et dispose d'un pouvoir limité pour configurer certains paramètres [4]

1.4.1.1 Les caractéristiques du SaaS

Selon [8], les caractéristiques du SaaS sont définies comme ceci :
• Le logiciel est mis à disposition via Internet.
• Le logiciel est maintenu par le fournisseur de services.
• La licence du logiciel est basée sur l'abonnement ou l'utilisation et facturée sur une base récurrente.
• Aucune maintenance n'est requise du côté de l'utilisateur final et, par conséquent, les applications SaaS sont très rentables.
• Le logiciel est disponible sur demande et peut être augmenté ou réduit en fonction de la demande.
• Le logiciel est mis à niveau et mis à jour automatiquement et prend également en charge la multilocation.

1.4.2 Platform as a Service

PaaS, également connu sous le nom de cloudware, offre une plate-forme informatique hébergée qui permet aux clients de déployer des applications sans avoir à acheter et à gérer le matériel requis et les couches logicielles sous-jacentes. En règle générale, PaaS fournit aux clients tout ce dont ils ont besoin pour créer et fournir des applications basées sur le cloud et services. Les offres, fournies sous forme de solution intégrée via le Web, incluent des fonctions d'application, conception, développement et test, Intégration de services Web , intégration de base de données ,Sécurité, évolutivité ,et espace de stockage.

4. https ://azure.microsoft.com/fr-fr/resources/cloud-computing-dictionary/what-is-cloud-computing/

1.4.3 Les caractéristiques du PaaS

Selon [8], les caractéristiques du PaaS sont comme suit :
• La sécurité intégrée, l'évolutivité et les interfaces de service Web sont fournies par PaaS.
• Des outils intégrés pour définir les règles métier et définir les processus de workflow et d'approbation sont fournis par PaaS.
• L'intégration d'applications avec d'autres applications sur la même plate-forme est facile.
• Le PaaS fournit des interfaces de services Web qui nous permettent de connecter les applications en dehors de la plate-forme.

1.4.4 Infrastructure as a Service

Il permet aux clients d'externaliser entièrement la fourniture de serveurs, de logiciels, d'espace de centre de données, et/ou des équipements réseau. Les composants en couches incluent généralement la facturation des services publics ou les accords de niveau de service, un environnement pour l'exécution de machines virtuelles spécifiées par les clients, le matériel informatique, l'ordinateur la mise en réseau (y compris les pare-feu et l'équilibrage de charge) et la connectivité Internet. IaaS incarne l'essence de l'informatique en nuage.

1.4.4.1 Les caractéristiques du Iaas selon

Selon [8], les caractéristiques du Cloud sont :
• IaaS fournit des machines virtuelles avec des systèmes d'exploitation préinstallés.
• Les ressources sont disponibles à la demande.
• IaaS permet de stocker des copies de données à différents endroits.
• Les ressources informatiques dans le cloud peuvent être facilement augmentées et réduites.

1.4.5 X as a Service

Anything-as-a-service, ou XaaS peut être traduit par " Tout et n'importe quoi en tant que Service ". Ce terme est né suite à l'émergence de nombreux services cloud computing tels que les SaaS,Paas et IaaS.
Dans l'acronyme XaaS, la lettre X fait référence au mot " everything " (tout) ou" anything " (n'importe quoi) [9].

Le tableau ci-dessous illustre les avantages et inconvénients des différents types de service :

	Avantages	Inconvénients
SaaS	-Pas d'installation. -Pas de licence. -Migration.	-Logiciel limité. -Sécurité. -Dépendance des prestataire.
PaaS	-Pas d'infrastructure nécessaire -Pas d'installation. -Environnement hétérogène.	-Limitation des langages. -Pas de personnalisation dans la configuration des machines virtuelles
IaaS	-Administration. -Personnalisation. -Flexibilité d'utilisation.	-Sécurité. -Besoin d'un administrateur système.

TABLE 1.1 – Avantages et inconvénients du SaaS, IaaS, et PaaS

1.5 Les modèles de déploiement du Cloud

Les services de cloud computing existants diffèrent dans leur utilisation pour les entreprises ou d'autres types d'utilisateurs. À cette fin, le cloud propose quatre modèles de déploiement, qui sont décrits comme suit :

1.5.1 Le Cloud Privé

L'infrastructure du cloud privé est provisionnée pour une utilisation exclusive par une seule organisation avec plusieurs consommateurs. Le cloud privé peut être appartenir, être géré et être exploité par l'organisation, par un tiers ou par une combinaison de ces derniers, et il peut exister à l'intérieur ou à l'extérieur des locaux de l'organisation [10].

1.5.2 Le Cloud public

L'infrastructure de cloud public est accessible par l'Internet. Elle est ouverte au public ou à de grands groupes industriels. Il peut appartenir, être géré et être exploité par une entreprise de commerce, par une organisation académique, ou par une organisation gouvernementale. Il existe sur les lieux du fournisseur du cloud public [10].

1.5.3 Le Cloud communautaire

L'infrastructure du cloud communautaire est provisionnée pour une utilisation exclusive par une communauté spécifique de consommateurs qui partagent les mêmes domaines d'intérêts. Il peut appartenir, être géré et être exploité par une organisation ou plus dans la communauté, par un tiers ou par une combinaison d'eux.[10]

1.5.4 Le cloud hybride

Est une composition de deux ou plusieurs types de clouds (privé, public et communautaire) [10].

1.6 Les caractéristiques du Cloud

Définit par l'institut national des standards et de la technologie NIST,le modèle de cloud computing contient essentiellement cinq caractéristiques, à savoir :

• **Accès à la demande par le consommateur** Les ressources et les services offerts par les fournisseurs sont toujours disponibles à la demande des utilisateurs.

• **Large accès au réseau** Le cloud computing utilise au possible les technologies les plus standardisées (essentiellement l'Internet), afin de rendre l'accès possible en utilisant n'importe quel appareil technologique.

• **Réservoir de ressources (Resource pooling)** La virtualisation est une pile principale pour construire une solution cloud computing, alors les ressources d'une plateforme cloud peuvent être physiques ou virtuelles, elles sont partagées et dynamiquement allouées afin de satisfaire les demandes des utilisateurs.

• **Redimensionnement rapide (élasticité)** En fonction de la demande, les ressources et les capacités peuvent être vendues rapidement et même dans certains cas automatiquement, provisionnées et libérées élastiquement. Pour le consommateur, les capacités disponibles pour l'approvisionnement semblent souvent illimitées et peuvent être appropriées à n'importe quelle quantité à n'importe quel moment.

• **Paiement à l'usage** Les services et ressources sont payés à l'usage selon la durée et la quantité d'utilisation. Cette faculté permet de diminuer le coût d'utilisation pour les consommateurs.

1.7 Les acteurs du Cloud

Le NIST [4] a organisé les acteurs majeurs du cloud en 5 catégories (Fig 4,Fig 5). Car pour une bonne compréhension du cloud computing, il est primordial de prendre connaissance des acteurs du domaine ainsi que leurs rôles.

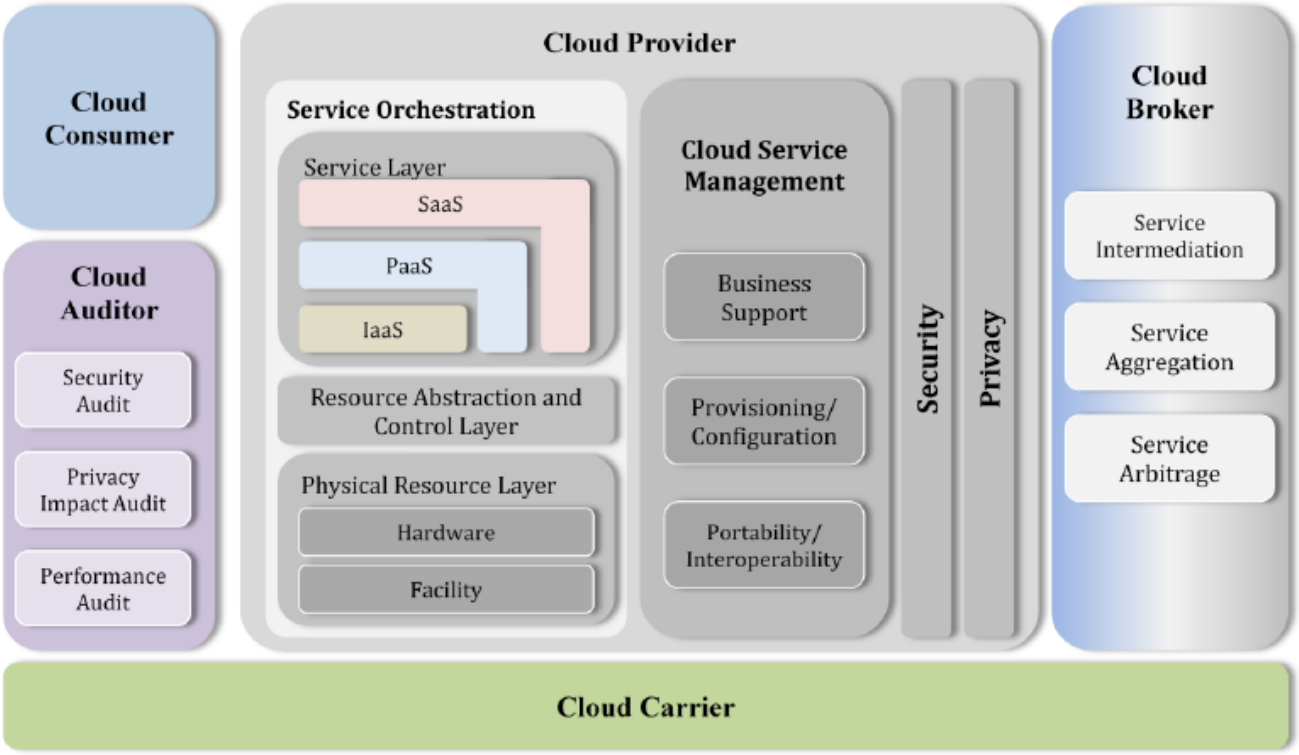

FIGURE 1.3 – Le modèle conceptuel de référence [4]

1.7.1 Le consommateur du Cloud (Cloud consumer)

Le consommateur de cloud est la partie prenante ultime pour laquelle le service de cloud computing est créé. Un consommateur de cloud représente une personne ou une organisation qui gère une relation et utilise le service d'un fournisseur de cloud. Un consommateur de cloud parcourt le catalogue de services d'un fournisseur de cloud, demande le service approprié, établit des contrats de service avec le fournisseur de cloud et utilise le service. Le consommateur de cloud peut être facturé pour le service fourni et doit organiser les paiements en conséquence. Selon les services demandés, les activités et les scénarios d'utilisation peuvent être différents parmi les consommateurs de cloud.

1.7.2 Le fournisseur du Cloud (Cloud Provider)

Un fournisseur de cloud peut être une personne, une organisation ou une entité chargée de fournir un service disponible pour les consommateurs de cloud. Un fournisseur de cloud construit le logiciel/la plate-forme/ services d'infrastructure, gère l'infrastructure technique nécessaire à la fourniture des services, fournit les services à des niveaux de service convenus et protège la sécurité et la confidentialité des services. Comme illustré dans la figure 4,les fournisseurs de cloud entreprennent différentes tâches pour la fourniture des différents modèles de services.

1.7.3 L'auditeur du Cloud (Cloud Auditor)

Un auditeur cloud est une partie qui peut effectuer une évaluation indépendante des services cloud, opérations, performances et sécurité du système d'information d'une implémentation cloud. Un auditeur du cloud peut évaluer les services fournis par un fournisseur de cloud en termes de contrôles de sécurité, impact sur la vie privée, performances, etc.

1.7.4 Courtier en Cloud (Cloud broker)

À mesure que le cloud computing évolue, l'intégration des services cloud peut être trop complexe pour le cloud consommateurs à gérer. Un consommateur de cloud peut demander des services cloud à un courtier cloud, au lieu de contacter directement un fournisseur de cloud. Un courtier cloud est une entité qui gère l'utilisation, la performance et la livraison des services cloud et négocie les relations entre le fournisseurs cloud et les consommateurs de cloud. En général, un courtier cloud peut fournir des services dans trois catégories :

- Intermédiation de services (Service Intermediation) un courtier cloud améliore un service donné en améliorant certaines capacités spécifiques et fournir des services à valeur ajoutée aux consommateurs de cloud. L'amélioration peut être la gestion de l'accès aux services cloud, la gestion des identités, rapports de performance, sécurité renforcée, etc.

- Agrégation de services (Service Aggregation) Aun courtier cloud combine et intègre plusieurs services en un seul ou plusieurs nouveaux services. Le courtier fournit l'intégration des données et assure la sécurité des données mouvement entre le consommateur de cloud et plusieurs fournisseurs de cloud.

- Arbitrage des services (Service Arbitrage) : l'arbitrage de services est similaire à l'agrégation de services, sauf que les services agrégés ne sont pas fixes. L'arbitrage de service signifie qu'un courtier a la flexibilité de choisir les services de plusieurs agences. Le courtier cloud, par exemple, peut utiliser un service de notation de crédit pour évaluer et sélectionner une agence avec la meilleure note.

1.7.5 Opérateur Cloud (Cloud Carrier) :

Un opérateur cloud agit comme un intermédiaire qui fournit la connectivité et le transport des services cloud entre les consommateurs de cloud et les fournisseurs de cloud. Les opérateurs cloud donnent accès au consommateurs via le réseau, les télécommunications et d'autres dispositifs d'accès. Par exemple, les consommateurs peuvent obtenir des services cloud via des périphériques d'accès au réseau, tels que des ordinateurs, ordinateurs portables, téléphones portables, appareils Internet mobiles (MID), etc. La distribution de services cloud est normalement fourni par des opérateurs de réseaux et de télécommunications ou un agent de transport, lorsqu'un agent de transport fait référence à une organisation commerciale qui assure le transport physique de stockage des supports tels que des disques durs de grande capacité. Notez qu'un fournisseur de cloud configurera le niveau de service accords (SLA) avec un opérateur cloud pour fournir des services conformes au niveau des SLA offerts aux consommateurs de cloud, et peut exiger que le fournisseur de cloud fournisse des services dédiés et connexions cryptées entre les consommateurs de cloud et les fournisseurs de cloud.
La figure ci-dessous illustre les différentes intéractions entre les différents acteurs :

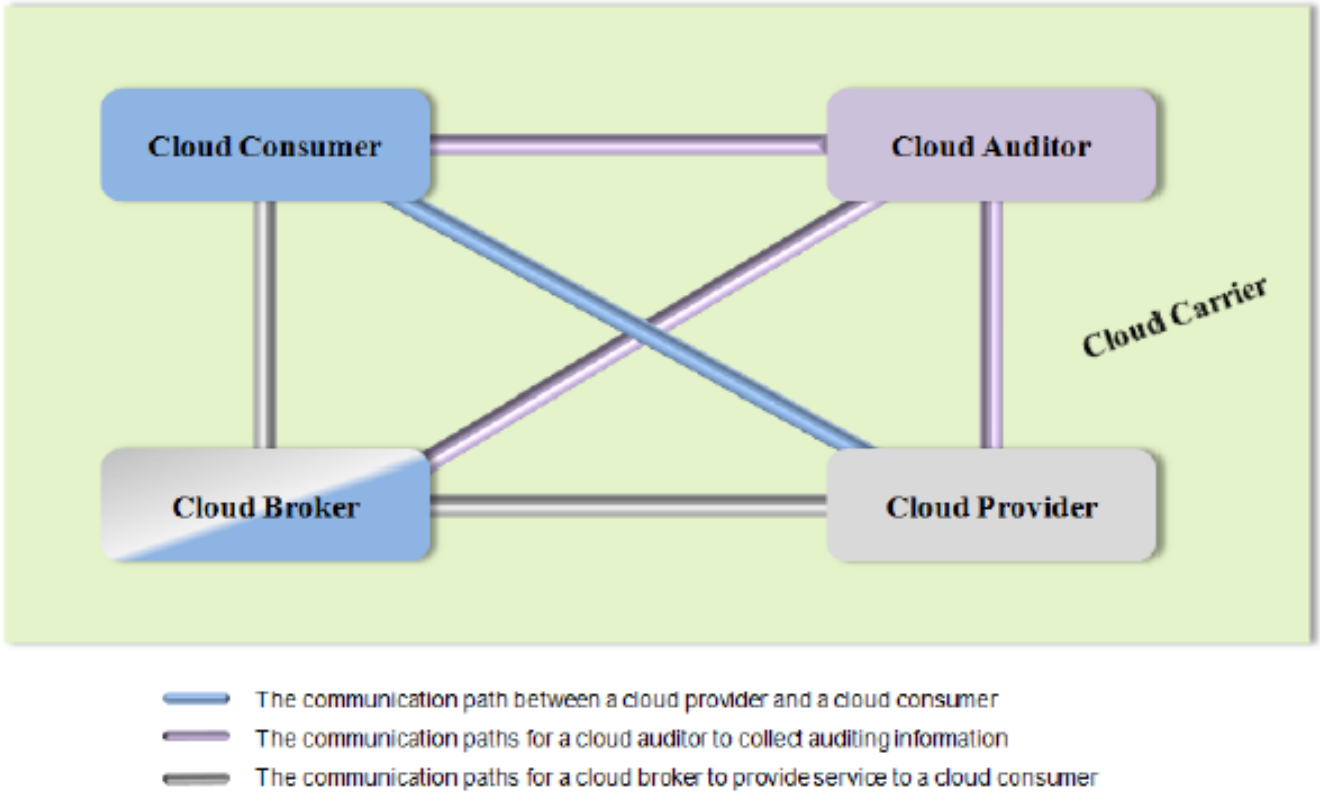

FIGURE 1.4 – Les interactions entre les différents acteurs du cloud [4]

1.8 Les éléments constitutifs du Cloud

Dans cette section nous jetons un regard de haut niveau sur les éléments technologiques qui constituent les bases de l'infrastructure informatique du Cloud. Selon [11] nous pouvons les ranger dans les grandes catégories suivantes :

• **Infrastructure :** L'infrastructure informatique du Cloud est un assemblage de serveurs, d'espaces de stockage et de composants réseau organisés de manière à permettre une croissance incrémentale supérieure à celle que l'on obtient avec les infrastructures classiques. Ces composants doivent être sélectionnés pour leur capacité à répondre aux exigences d'extensibilité, d'efficacité, de robustesse et de sécurité. Les serveurs d'entreprise classiques ne disposent pas des capacités réseau, de la fiabilité ni des autres qualités nécessaires pour satisfaire efficacement et de manière sécurisée les accords de niveau de service (SLA, service level agreement). Par ailleurs, les serveurs d'un Cloud affichent des coûts de fonctionnement moins élevés et ils peuvent être plus fiables s'ils ne sont pas tous équipés de disques internes.

• **Réseaux IP :** Dans une infrastructure de Cloud, le réseau non seulement connecte les utilisateurs au Cloud, mais sert également à l'interconnexion interne du Cloud. Le modèle mis en œuvre dans un réseau d'entreprise ne répond pas aux besoins d'efficacité et de sécurité associés à l'acquisition et au fonctionnement du Cloud. À l'échelle du Cloud, le réseau doit s'orienter vers un système Carrier-Grade, avec des stratégies réseau optimisées. Les multiples commutateurs disséminés tout au long des chemins de données deviennent des points uniques de défaillance (SPOF, single points of failure) et ajoutent des coûts de différentes manières. Bien que l'optimisation puisse conduire à un seul réseau unifié, la sécurité nécessite un partitionnement ou une virtualisation du réseau pour une séparation réelle entre les différentes classes de trafic. Le réseau sera probablement plus plat, mais vous devez vous attendre à plusieurs réseaux en parallèle pour une meilleure sécurité. Certains isolent la gestion de la plateforme des données publiques et du trafic de service, tandis que d'autres peuvent être nécessaires pour autoriser les évolutions. Ces réseaux supplémentaires induisent de nouveaux coûts, mais vous obtenez alors une séparation physique et une meilleure sécurité.

• **Virtualisation :** Avec des racines profondément ancrées dans l'informatique, la virtualisation sert à partitionner un seul serveur physique en plusieurs machines virtuelles ou une seule ressource physique, comme un espace de stockage ou un réseau, en plusieurs ressources virtuelles. Elle permet une consolidation de serveurs avec une grande souplesse d'utilisation. Dans le contexte de l'informatique en nuage, la virtualisation est importante pour la mise en service et le retrait rapide de serveurs. Un logiciel de virtualisation du Cloud présente également une perspective dynamique et une vue unifiée de l'utilisation et de l'efficacité des ressources, cela afin d'assurer le fonctionnement des services du Cloud. La virtualisation est la principale technologie permettant d'arriver à une utilisation rentable des serveurs tout en prenant en charge

la séparation entre de multiples locataires d'un matériel physique. Il existe d'autres solutions pour arriver à ces objectifs, mais ses avantages en font l'approche de choix.

• **Logiciel :** Il autorise la mise en œuvre de tous les aspects de la gestion, de la mise à disposition, du développement des services, de la comptabilité et de la sécurité de l'infrastructure du Cloud. Il est indispensable que l'infrastructure du Cloud soit capable d'appliquer dynamiquement des politiques de séparation, d'isolation, de surveillance et de constitution d'un service. Le choix d'une configuration standard pour l'infrastructure permet au logiciel d'automatiser les tâches sous-jacentes à l'élasticité et au changement de forme de manière à présenter des services constitués de serveurs, de machines virtuelles, d'espaces de stockage, de services et d'autres composants informatiques. Grâce au logiciel, nous pouvons automatiser la mise en service et le retrait.

• **Interfaces de service :** L'interface de service placée entre le fournisseur et le client est un élément de différenciation du Cloud. Elle représente un contrat qui fait respecter la proposition de valeur décrite par des SLA et des conditions tarifaires. Si le Cloud semble nouveau, c'est principalement en raison de cette interface. Elle représente la valeur d'un fournisseur et sert de base à la concurrence. Par l'ajout d'interfaces de libre-service, nous obtenons d'autres optimisations. Les clients du Cloud sont en mesure d'engager des ressources de manière automatisée sans que le service informatique soit un obstacle. L'espace de stockage et les ressources sont présentés au travers d'une interface graphique que l'utilisateur peut manipuler de manière à obtenir et à instancier une infrastructure informatique virtuelle. Un navigateur web et une carte bancaire, voilà tout ce qu'il vous faut pour construire votre propre Datacenter virtuel.

• **Data center :** Le terme "Datacenter" signifie différemment pour différentes personnes. Parmi les noms utilisés, citons centre de données, hall de données, ferme, entrepôt de données, salle informatique, salle de serveur, R and D laboratoire logiciel, laboratoire haute performance, hébergement, colocation, etc. L'Environnent Protection Agency des États-Unis définit un Datacenter comme :

- Principalement les équipements électroniques utilisés pour le traitement des données (serveurs), stockage de données (équipement de stockage), et communications (équipement de réseau). Collectivement cet équipement traite, stocke et transmet des données numériques (information)[12] .
- Équipements spécialisés très puissant de conversion et de restauration maintenir une alimentation fiable et de haute qualité, ainsi qu'équipement de contrôle de l'environnement pour maintenir la température et humidité appropriées pour les équipements électroniques [9].

1.9 L'architecture du cloud computing

L'architecture du cloud est composée de couche, quatre pour être bien précis le matériel (couche centre de données), couche infrastructure, la couche plate-forme et la couche d'application, comme illustré dans la Fig. 1 [13] ,Nous décrivons chacune d'elle en détail dans ce qui suit :

1.9.1 La couche matérielle

Cette couche est responsable de la gestion des ressources physiques du cloud, y compris les serveurs physiques, les routeurs, les commutateurs, les systèmes d'alimentation et de refroidissement. En pratique, la couche matérielle est typiquement implémentée dans les centres de données. Un centre de données contient généralement des milliers de serveurs organisés en racks et interconnectés via des commutateurs, des routeurs ou d'autres tissus. Les Problèmes typiques à la couche matérielle comprennent la configuration matérielle, la tolérance aux pannes, la gestion du trafic, la gestion de l'énergie et refroidissement des ressources

1.9.2 La couche infrastructure

Aussi connu sous le nom de couche de la virtualisation, la couche infrastructure crée un pool de stockage et les ressources informatiques en partitionnant les ressources physiques en utilisant des technologies de virtualisation. La couche d'infrastructure est un composant essentiel du cloud computing, car de nombreuses fonctionnalités, telles que l'affectation dynamique des ressources, sont uniquement mis à disposition grâce aux technologies de virtualisation.

1.9.3 La couche plate-forme

construite au-dessus de la couche infrastructure, la couche plate-forme se compose de systèmes d'exploitation et cadres applicatifs. Le but de la couche plate-forme est de minimiser la charge de déploiement des applications directement dans des conteneurs de VM. Par exemple, Google App Engine exploite au niveau de la plate-forme pour fournir un support API pour la mise en œuvre stockage, base de données et logique métier du Web typique applications.

1.9.4 La couche applicative

Au plus haut niveau de la hiérarchie la couche d'application se compose des applications cloud réelles. Différente des applications traditionnelles, les applications cloud peuvent tirer parti de la fonction de mise à l'échelle automatique pour atteindre de meilleures performances, disponibilité et coûts d'exploitation réduits.
Par rapport aux environnements d'hébergement de services traditionnels comme les fermes de serveurs dédiés, l'architecture du cloud informatique est plus modulaire.

Chaque couche est faiblement couplée avec les couches au-dessus et en dessous, permettant à chaque couche d'évoluer séparément. Ceci est similaire à la conception du modèle OSI pour les protocoles réseau. La modularité architecturale permet au cloud computing de prendre en charge un large éventail d'applications exigences tout en réduisant la gestion et la maintenance aérien.

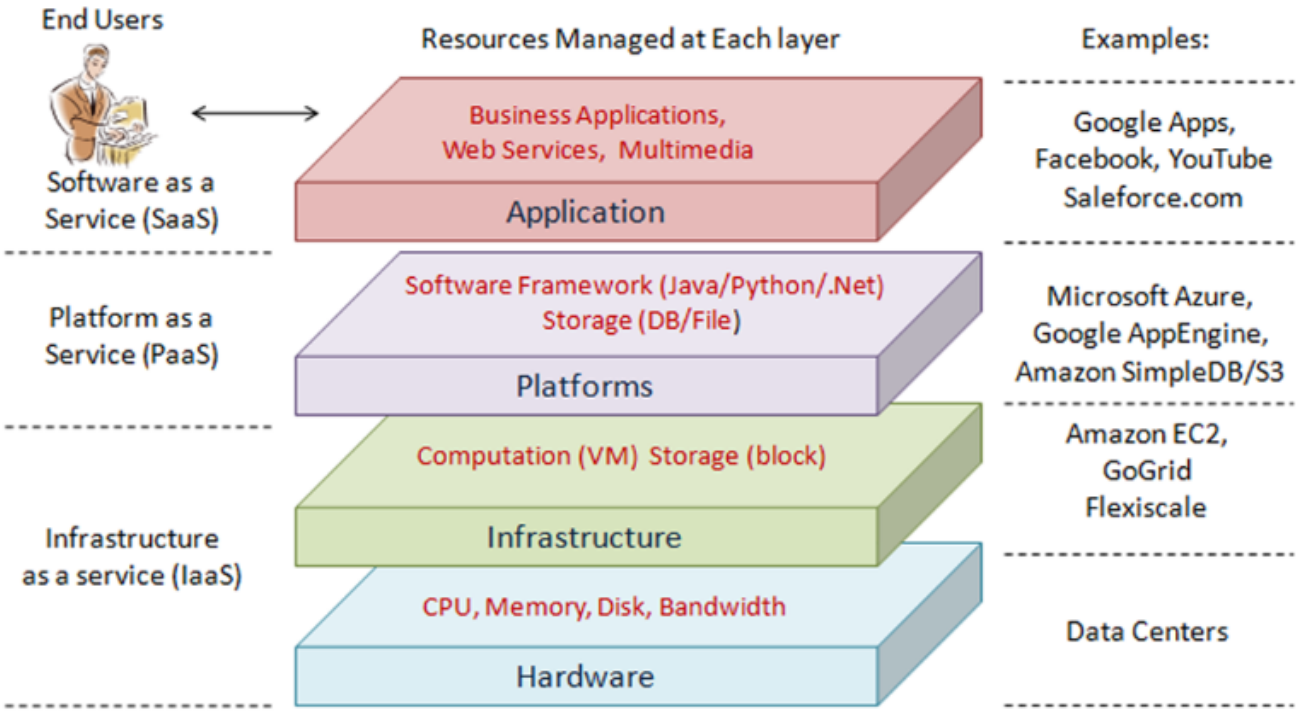

FIGURE 1.5 – L'architecture du Cloud Computing

[13]

1.10 Virtualisation et son rôle dans le Cloud

La virtualisation est utilisée pour générer un système physique simulé sur un système physique réel. Elle permet d'utiliser une ressource informatique virtuelle à partir d'une machine physique réelle. Dans la plupart des cas, il existe de multiples systèmes physiques simulés. C'est dans ce sens que la virtualisation est utilisée pour créer une densité de systèmes. Nous pouvons avoir plusieurs systèmes virtuels, appelés machines virtuelles, fonctionnant sur un seul système physique. Ces systèmes virtuels partagent l'utilisation des ressources physiques tels qu'un processeur, une interface réseau ou un disque dur, ces derniers sont alloués à une machine virtuelle pour que celle-ci fonctionne comme une machine physique.

Lorsqu'un système virtuel n'utilise pas les ressources d'un système physique, celles-ci peuvent être utilisées par un autre système virtuel. Dans un environnement non virtualisé, les ressources du système peuvent être inactives pendant une période de temps. Il existe deux type de virtualisation [14] :

• **Virtualisation complète.**

• **Paravirtualisation.**

1.10.1 Virtualisation complète :

La virtualisation complète est conçue pour fournir une abstraction totale du système physique sous-jacent et crée un système virtuel complet dans lequel les systèmes d'exploitation invités peuvent s'exécuter. Aucune modification n'est requise dans le système d'exploitation ou l'application invité. Le système d'exploitation ou l'application invité n'est pas au courant de l'environnement virtualisé et peut donc s'exécuter sur la machine virtuelle de la même manière que sur un système physique. Cette approche peut être avantageuse car elle permet de découpler complètement le logiciel du matériel. En conséquence, la virtualisation complète peut simplifier la migration des applications et des charges de travail entre différents systèmes physiques. La virtualisation complète permet également d'isoler complètement différentes applications, ce qui contribue à rendre cette approche hautement sécurisée.

1.10.2 Paravirtualisation

La Paravirtualisation présente à chaque machine virtuelle une abstraction du matériel similaire mais non identique au matériel physique sous-jacent. Les techniques de Paravirtualisation nécessitent des modifications des systèmes d'exploitation invités qui s'exécutent sur les ordinateurs virtuels. En conséquence, les systèmes d'exploitation invités savent qu'ils s'exécutent sur une machine virtuelle, permettant ainsi des performances quasi natives.

Le but de la virtualisation est de permettre la transparence d'utilisation, l'efficacité d'exploitation des ressources, d'assurer le fonctionnement des différents services et la séparation entre les multiples utilisateurs impliqués dans un matériel physique [7].

Pour faire simple, le cloud s'appuie sur la technologie de virtualisation pour atteindre l'objectif d'allocation dynamique des ressources informatiques en fonction de l'utilité.

Certains des principaux avantages de la virtualisation, qui sont liés au Cloud, sont les suivants [15] :

• La facturation est basée sur l'utilisation (la tarification des services) et plutôt que sur la capacité matérielle fixe.
• Le déploiement rapide de serveurs supplémentaires.
• Les clients sont séparés des emplacements de serveurs physiques.
• L'utilisation est basée sur des accords de Niveau de Service /Service-Level Agreements (SLA).
• La tolérance aux pannes.
• La mobilité des applications entre les serveurs et les centres de données.
• Des réseaux flexibles, agiles réduit considérablement les coûts.

1.11 Accord de niveau de service

" Accord de niveau de service " ou " Service-Level Agreement "ou " contrat de niveau de service " [13] est la partie d'un contrat qui définit exactement les services qu'un fournisseur de services fournira et le niveau ou la norme requise pour ces services.
Le SLA fait généralement partie d'un accord d'externalisation ou de services gérés, ou peut être utilisé dans des accords de gestion des installations et d'autres accords de fourniture de services [16].
En d'autres termes, il s'agit d'une clause contractuelle qui définit les objectifs précis et la qualité de service que le prestataire contracté exige et attend.

1.12 Evolution du Cloud

Étant donné que différents types de contenus multimédias numériques peuvent être produits et diffusés sur différents réseaux, de sorte qu'un mécanisme standard est nécessaire pour permettre l'interopérabilité entre les clouds et le transcodage des contenus multimédias. Le but du media cloud est de résoudre ce problème et de permettre aux utilisateurs de constituer un cloud et gérer les contenus médias en toute transparence, même s'il est situé en dehors du domaine de l'utilisateur. Pour la découverte du le service et la création de plus de services, la communication entre deux ou plusieurs nuages devient nécessaire. C'est ce qu'on appelle l'informatique inter-cloud [17].

La communication entre deux ou plusieurs cloud est connue sous le nom de d'informatique inter-cloud. Lorsqu'il existe de nombreux clouds avec du contenu multimédia, les clouds doivent être en mesure de communiquer les uns avec les autres, créant ainsi un scénario d'informatique inter-cloud. Ceci est également important pour répondre aux demandes croissantes, car l'utilisateur peut avoir divers types d'exigences, qui peuvent ne pas être offertes par un seul cloud. Pour répondre à ces exigences, un cloud doit demander un autre cloud ou plusieurs nuages. En outre, le nuage doit être capable de découvrir les services disponibles ailleurs.

Cette informatique inter-cloud va créer un "cloud de clouds " (CoC), capable de communiquer directement les données qui ne sont pas stockées dans ses centres de données directement[17].

1.12.1 Inter-Cloud

L'Inter-Cloud est un "nuage de nuages" mondial interconnecté et une extension du "réseau de réseaux" sur lequel il est basé. L'informatique inter-cloud consiste à interconnecter les infrastructures de plusieurs fournisseurs de clouds. L'accent est mis sur l'interopérabilité directe entre les fournisseurs de services de clouds publics. Pour fournir avec succès des services en cloud en tant qu'utilitaires, des clouds interconnectés sont nécessaires, et l'interopérabilité et la portabilité sont des facteurs importants de l'informatique en cloud[8].

1.12.2 Pourquoi utiliser l'Inter-Cloud

Les limites du cloud sont qu'ils ont des ressources physiques limitées. Si un cloud a épuisé tous les calculs et des ressources de stockage, il ne peut pas fournir de service aux clients. L'Inter-Cloud traite de telles situations où chaque cloud utiliserait les ressources de calcul, de stockage ou tout autre type de ressources des infrastructures d'autres clouds. L'environnement Inter-Cloud offre des avantages tels que divers emplacements géographiques, une meilleure résilience des applications et évite le verrouillage du fournisseur sur le cloud client. Les avantages pour le fournisseur de cloud sont étendus à la demande et meilleurs accords de niveau de service (SLA) avec le cloud client [8].

1.12.3 Les types d'Inter-Cloud

Nous pouvons distinguer deux types d'inter-cloud, la fédération et le multi-cloud. Et chaque type est lui-même subdivisé en deux sous-familles celles qui sont citées par[8] :

1.12.3.1 Fédération de Cloud

La fédération de cloud est un inter-cloud où un ensemble de fournisseurs de clouds interconnectent volontairement leurs infrastructures de cloud afin de partager des ressources entre eux. Les fournisseurs de clouds de la fédération collaborent volontairement pour échanger des ressources. Ce type d'inter-cloud convient à la collaboration de clouds gouvernementaux (des cloud détenus et utilisés par une institution à but non lucratif ou un gouvernement) ou de portefeuilles de clouds privés (le cloud fait partie d'un portefeuille de clouds qui appartiennent à la même organisation). Les types de clouds de fédération sont les clouds Peer to Peer et les clouds centralisés.

1.12.3.2 Multi- Cloud

Dans un Multi-Cloud, un client ou un service utilise plusieurs clouds indépendants. Un environnement multi-cloud n'a pas d'interconnexion volontaire et de partage des infrastructures des fournisseurs de services cloud. La gestion de l'approvisionnement et de la planification des ressources relève de la responsabilité du client ou de ses représentants. Cette approche est utilisée pour utiliser les ressources des clouds gouvernementaux et des portefeuilles de cloud privés. Les types de multicloud sont les services et les bibliothèques.

1.12.4 Topologies des différentes architectures de l'Inter-Cloud

Il existe quatre topologies pour les différents types de l'inter-cloud, cités par [17] dans ce que suit :

• **Fédération Inter-Cloud peer to peer :** Les clouds collaborent directement les uns avec les autres mais peuvent utiliser des entités distribuées pour les annuaires ou le courtage. Les clouds communiquent entre eux et négocient directement sans médiateurs. La fédération Peer to Peer Inter-Cloud est illustrée dans la figure 8. Les projets Inter-Cloud qui utilisent le Peer to Peer fédération sont RESERVOIR (Projet de Virtualisation des Ressources et des Services sans Barrières), Open Cirrus,OPTIMIS, Arjuna Agility et Global Inter-Cloud par Bernstein et al.

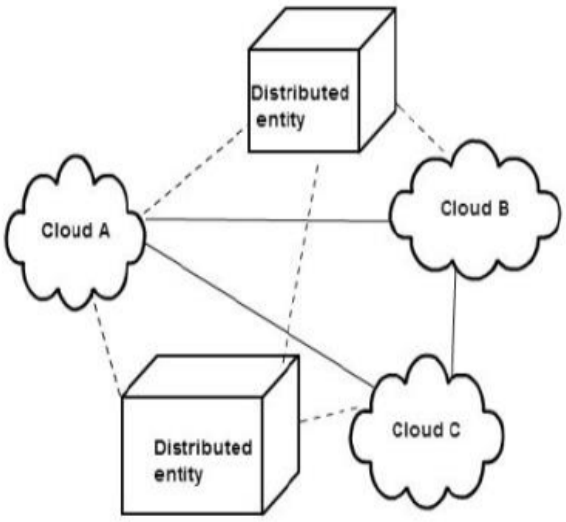

FIGURE 1.6 – Fédération Inter-Cloud peer to peer .
[8]

• **Fédération centralisée Inter-Cloud :** Les clouds utilisent une entité centrale pour effectuer ou faciliter le partage des ressources. L'entité centrale agit comme un entrepôt où les ressources cloud disponibles sont enregistrées. La fédération inter-cloud centralisée est illustrée dans la figure 9. Les projets Inter-Cloud qui utilisent la fédération Centralisée Inter-Cloud sont Inter-Cloud, Contrail, Dynmic Cloud Collaboration (DCC) et Federated Cloud Management.

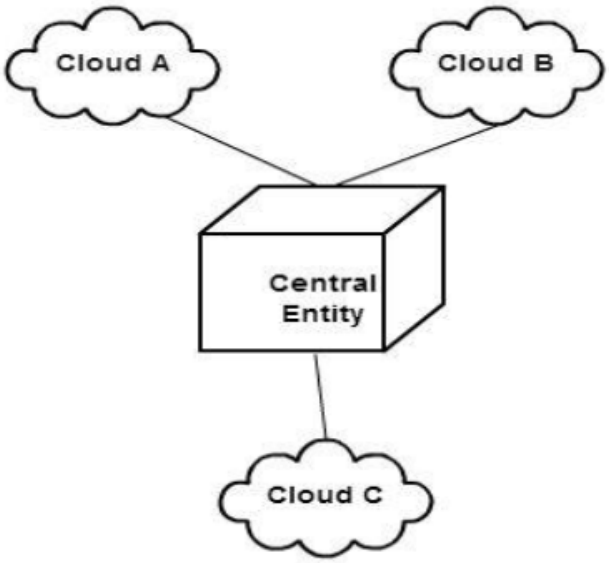

FIGURE 1.7 – Fédération centralisée Inter-Cloud .
[8]

• **Service Multi-Cloud :** Les clients accèdent à plusieurs clouds via un service. Un service est hébergé par le cloud client en externe ou en interne. Les services contiennent des composants de courtier. Le Service Multi-cloud est décrit dans la figure 10. Les projets Inter-Cloud qui utilisent les services Multi-cloud sont OPTIMIS, Contrail, mOSAIC, STRATOS et Commercial Cloud Management Systems.

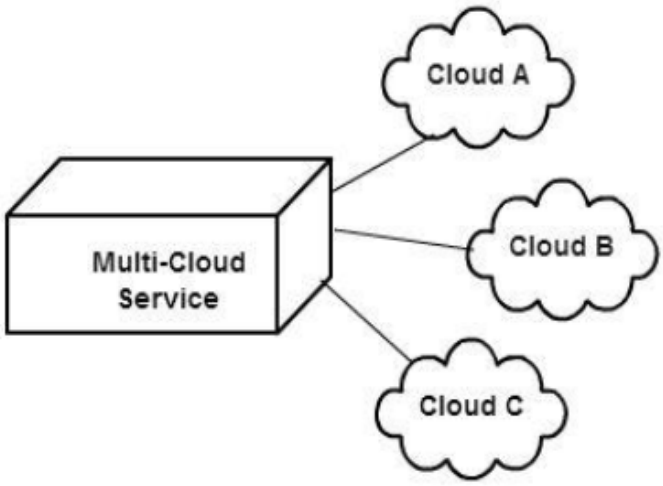

FIGURE 1.8 – Répartition des responsabilités
[8] .

• **Bibliothèque Multi-Cloud :** Les clients développent leurs propres courtiers en utilisant une API cloud unifiée comme bibliothèque. Les Inter-Clouds qui utilisent des bibliothèques facilitent l'utilisation des clouds de manière uniforme. La bibliothèque multi-cloud est illustrée dans la figure 11 .Des exemples de plusieurs bibliothèques multi-cloud sont la bibliothèque Java JClouds, la bibliothèque Python Apache LibClouds, Ruby bibliothèque Apache DeltaCloud, bibliothèque PHP SimpleCloud, Apache Nuvem.

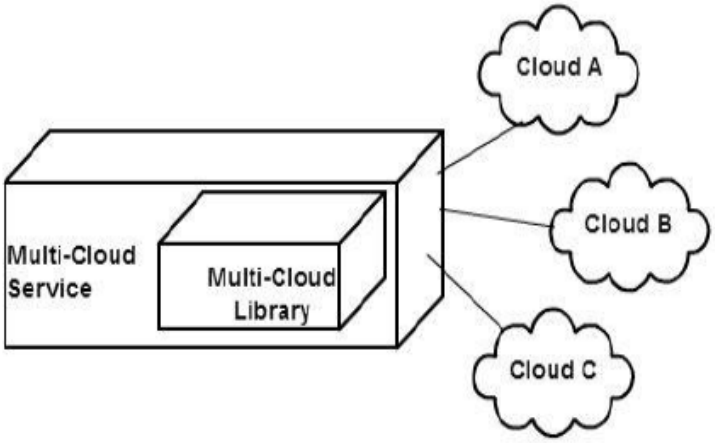

FIGURE 1.9 – Répartition des responsabilités d'une librairie

[8] .

1.13 Avantages et inconvénients du cloud

Le Cloud computing est comme toutes autres technologies ,Il offre plusieurs avantages comme cité sur [18] :

- Gestion facile :

La maintenance de l'infrastructure, qu'elle soit matérielle ou logicielle, est simplifiée, donc moins de casse-tête pour l'équipe informatique. De plus, les applications qui nécessitent beaucoup de stockage sont plus faciles à utiliser dans l'environnement cloud que lorsqu'elles sont utilisées par l'organisation seule. Au niveau de l'utilisateur également, vous avez surtout besoin d'un simple navigateur Web avec une connectivité Internet.

- Réduction des coûts :

Des systèmes coûteux ne sont pas nécessaire pour une utilisation occasionnelle de ressources informatiques intensives. De plus, la main-d'œuvre requise pour de tels systèmes n'est pas requise. Même des applications simples comme le courrier électronique peuvent être configurées et principalement via des applications telles que Google Apps. De plus, comme la plupart du temps, ces fournisseurs sont assez fiables en termes de disponibilité, il est clairement gagnant.

- Services ininterrompus :
Des pannes plus faibles sont fournies par le cloud computing services, fournissant ainsi des services ininterrompus à l'utilisateur.
- La gestion des catastrophes :
En cas de sinistre, une sauvegarde hors site est toujours utile. La sauvegarde des données cruciales à l'aide de services de stockage du cloud est le besoin de l'heure pour la plupart des organisations. De plus, les services de stockage du cloud conservent non seulement vos données hors site, mais ils garantissent également des systèmes mis en place pour la reprise après sinistre.

- Un démarrage rapide :
Le cloud computing permet de tester le business plan rapidement, à coûts réduits et avec facilité.
- L'agilité pour l'entreprise :
Résolution des problèmes de gestion informatique simplement sans avoir à vous engager à long terme.
- Un développement plus rapide des produits :
Réduisons le temps de recherche pour les développeurs sur le paramétrage des applications.
- Pas de dépenses de capital :
Plus besoin des locaux pour élargir vos infrastructures informatiques.

Quant aux Inconvénients nous citons [19] :
- La bande passante peut faire exploser votre budget :
La bande passante qui serait nécessaire pour mettre cela dans le Cloud est gigantesque, et les coûts seraient tellement importants qu'il est plus avantageux d'acheter le stockage nousmêmes plutôt que de payer quelqu'un d'autre pour s'en charger.
- Les performances des applications peuvent être amoindries :
Un Cloud public n'améliorera définitivement pas les performances des applications.
- La fiabilité du Cloud :
Un grand risque lorsqu'on met une application qui donne des avantages compétitifs ou qui contient des informations clients dans le Cloud,

1.14 Conclusion

Dans ce chapitre nous avons introduits les notions de bases et généralités sur le Cloud Computing afin de bien comprendre ce domaine et pour avoir une idée globale sur le Cloud.
Le but de ce chapitre est de mettre l'accent sur l'importance de l'utilisation du cloud, ses différentes caractéristiques et acteurs du domaine.
Dans le prochain chapitre, nous allons explorer l'optimisation multi-objectives et son importance dans la sélection.

Chapitre 2

Chapitre II : La composition de service dans le cloud

2.1 Introduction

Pour établir un contact entre un client et un fournisseur dans le cloud, nous avons besoin d'une connexion internet où les clients utilisent un terminal pour effectuer ou exécuter des fonctionnalités offertes par le fournisseur.
Cependant pour satisfaire la requête d'un utilisateur qui est un ensemble de fonctionnalités cela nécessite l'invocation de services cloud pour l'exécution de ces fonctionnalités, donc la satisfaction de la requête du client. Mais étant donné que les requêtes client sont de plus en plus complexe donc composé de plusieurs fonctionnalités et que les services cloud sont atomiques (assure l'exécution d'une seule fonctionnalité).
Nous sommes dans l'urgence de faire une combinaison de services, car un seul ne suffit pas pour satisfaire la requête du client. Dans ce cas il s'agit de la composition de services ou service composite.

2.2 Définition de la composition de services dans le cloud

La composition de services dans le cloud est le processus qui permet la combinaison de plusieurs services atomiques complexes pour produire un nouveau service complexe afin de satisfaire la requête du client (utilisateur).
En général, la composition des services est un processus qui permet de former et exécuter une séquence des services atomiques pour réaliser une tâche complexe qui ne peut pas être réalisée grâce à un seul service atomique.

La composition de services Web est notamment un processus complexe qui consiste à trouver des composants appropriés, les sélectionner et les combiner, fournir des informations d'ordre comportemental renseignant sur leur orchestration et chorégraphie en vue de les rendre exécutables. Ces deux derniers concepts d'orchestration et de chorégraphie, servent à décrire le comportement interne et externe d'un service Web composite [20]

2.3 Présentation du problème de composition de services

Le problème de la composition de services est un problème récurrents compte tenu du nombre complexes de services cloud disponible. Ce problème a été présenté par [21] comme suit : Les utilisateurs des plates-formes clouds utilisent des services pour satisfaire leurs demandes. Mais ces demandes deviennent très complexes, et ils ont besoin plus d'un seul service pour accomplir une demande. Le processus de collecte d'un ensemble des services pour satisfaire une demande de l'utilisateur est appelé la composition de services.

En générale, l'architecture de cloud est composée de deux parties : la première, appelée Front End, qui est visible pour les clients, et généralement représentées par un ordinateur ou tout type de terminal. La deuxième partie, appelée Back End, qui rassemble les technologies complexes du cloud qui sont transparents aux clients. La connexion entre deux parties est raccordée via Internet, où les clients utilisent le frontal pour exécuter les fonctionnalités offertes par le fournisseur de cloud en tant que service. Au niveau du Front End, les utilisateurs voulant satisfaire leurs demandes, où chaque requête est un ensemble de fonctionnalités, ils invoquent les services de cloud qui assurent l'exécution de ces fonctionnalités sur le cloud même. Au niveau de Back End, les services du cloud sont atomiques et chaque service assure l'exécution d'une seule fonctionnalité, alors que la requête de l'utilisateur est composée de nombreuses fonctionnalités. C'est pourquoi un service ne peut pas satisfaire la demande de l'utilisateur, mais il faut plusieurs services pour donner à l'utilisateur le résultat attendu.

Le processus de construction d'un service à partir de l'ensemble des services sélectionnés pour donner à l'utilisateur le meilleur résultat est appelée le Cloud Computing Service Composition (CCSC) [22]. La figure 1 montre le mécanisme de fonctionnement d'un environnement cloud durant l'exécution d'une requête envoyée par un utilisateur.

Egalement, la figure 1 montre qu'il y a une transparence totale pour l'utilisateur durant l'exécution de sa requête. L'opération de composition de services est achevée sans l'intervention de l'utilisateur. Aussi, on voit clairement que le processus de composition de services conduit à la sélection d'un ensemble de cloud bases, où les services impliquées dans le processus de composition sont situés.
L'exemple donné dans la figure 16 montre la requête de l'utilisateur qui est représenté par la liste des taches t1, t2, t3, t4 exécute par les services S1S2, S3, S4, les compositions possibles pour cette exemple sont les suivantes (on ne cite pas ici toutes les compositions, mais juste des instances) :

La composition de services	**le combinaison de cloud bases sélection**
$C_1/S_1 - C_2/S_2$ - $C_1/S_3 - C_3/S_4$	$C_1 - C_2 - C_3$
$C_1/S_1 - C_2/S_2 - C_4/S_3 - C_3/S_4$	$C_1 - C_2 - C_3 - C_4$
$C_1/S_1 - C_3/S_2$ - $C_1/S_3 - C_3/S_4$	$C_1 - C_3$
$C_1/S_1 - C_2/S_2$ - $C_1/S_3 - C_2/S_4$	$C_1 - C_2$

FIGURE 2.1 – La relation entre la composition de service et la sélection des clouds [21]

Il est évidemment que, s'il est possible de diminuer le nombre des cloud bases impliqués dans le processus de composition, cela permet de réduire le coût de la communication entre ces services. Cette réduction apparait clairement dans les points suivants :

1. Temps de réponse : si le nombre de cloud bases impliqués dans le processus de composition est élevée, le temps de réponse sera plus grand, en particulier si les services sont interdépendants entre eux (la sortie d'un service est l'entrée d'un autre).

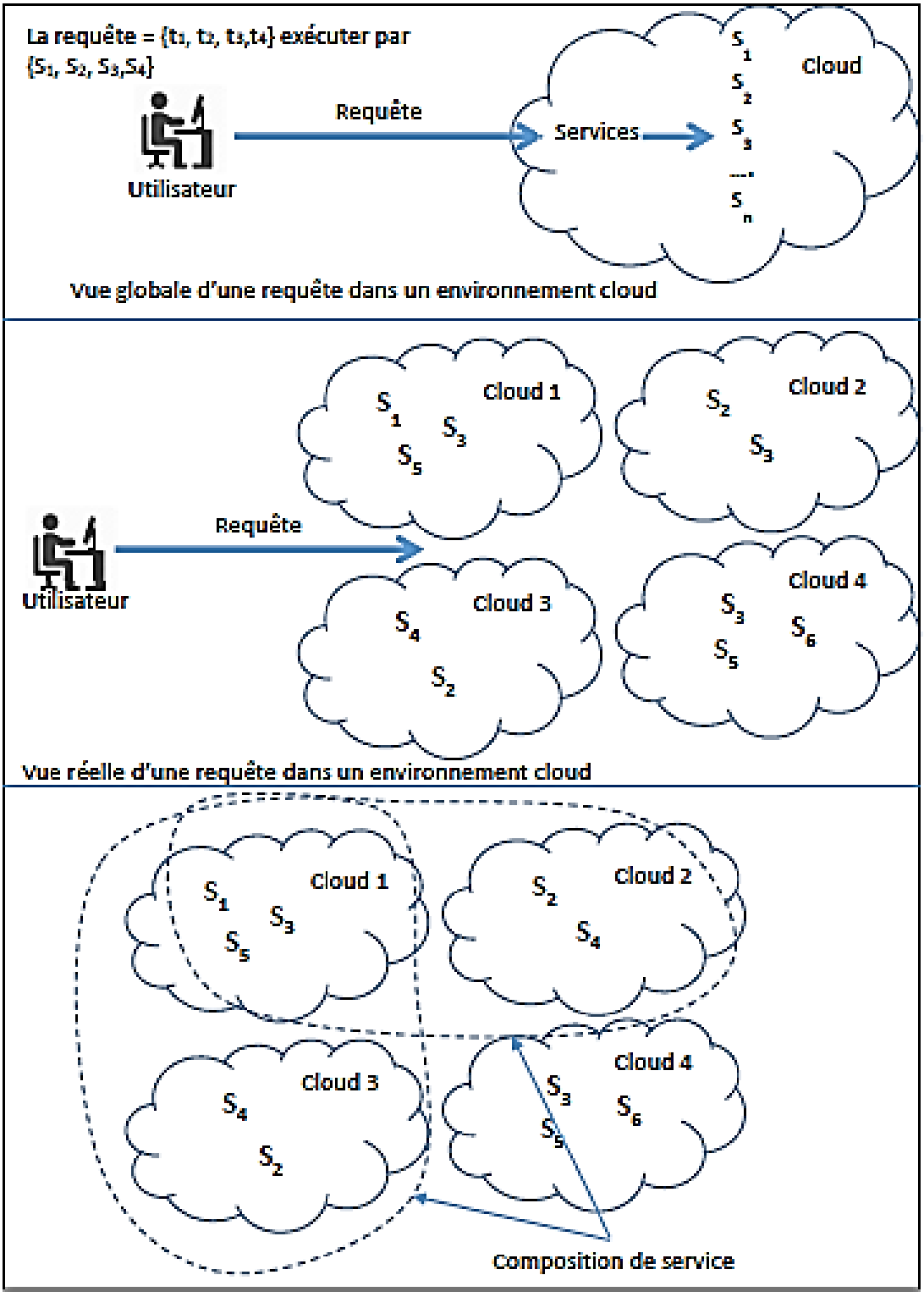

FIGURE 2.2 – la composition de services dans le cloud computing

2. Prix d'accès : si l'accès à l'ensemble des cloud bases n'est pas gratuit, il est clair que l'utilisateur préfère utiliser le minimum de nombre de ces bases.

3. La bande passante du réseau : à partir du point (1), il est clair que si le processus de composition arrive à trouver un nombre minimal de cloud bases, ce qui permettra de réduire la quantité de données échangées sur le réseau.
Les services sont conçus pour effectuer un ensemble d'opérations atomiques, où une opération ne satisfait pas le demande du client mais juste une partie de celui-ci.
Pour satisfaire complètement la demande du client, nous avons besoin de composer plusieurs services pour construire un service composé capable d'effectuer des tâches complexes. Par exemple, si un client veut voyager pour passer ses vacances dans un pays étranger, son agence lui proposera de réserver un vol, de réserver un hôtel, de louer une voiture et de participer à certaines activités organisées, Ces tâches sont proposées comme des services sur des plateformes cloud, et pour chaque tâche, il existe plusieurs services pour effectuer une tâche donnée, Parce que ces multiples services sont différents en plusieurs termes, l'agence a besoin de stratégies ou de mécanismes pour sélectionner un service pour chaque tâche et pour composer les services sélectionnés dans un seul service afin de maximiser la satisfaction du client.

2.4 Les défis de la composition de services

La composition des services est une opération laborieuse qui comprend plusieurs difficultés et défis la rende plus complexe et plus difficile à mettre en oeuvre, définit par [21] comme suit :

• L'interopérabilité entre les services ou bien entre les cloud est un défi dans la composition, parce que chaque service peut être décrit de manière diffèrent par rapport à autre, et aussi chaque cloud utilise des protocoles et des langages différents, ça rendre la communication entre les services et les cloud difficile et le processus de composition deviendra plus complexe.

• L'augmentation de nombre des services candidats pour réaliser chaque tâche pose un problème de passage à l'échelle.

• Le changement rapide des valeurs de qualité de service pour chaque service peut poser un problème dans la composition parce que les critères de qualité de service peuvent se changent chaque instant et ça rendre la mesure des attributs de qualité de service plus complexe et influencer la qualité de service du service composé.

• La sélection optimale des services par un courtier dépend de la disponibilité d'informations complètes et actualisées sur les services. Faire face à plusieurs changements dans les caractéristiques du service pourrait entraîner la perte de certaines données.

• Décrire et mesurer les attributs de qualité de service des services de réseau. L'absence de consensus sur la définition et la description des paramètres de qualité de service des services de réseau parmi les services de cloud computing mondiaux est toujours un défi important pour les développeurs de cloud computing. L'absence de moyens convenus pour mesurer la qualité de service des réseaux est un autre problème qui n'est pas complètement résolu et qui devrait être pris en considération.

• La sécurité est aussi un défi important, conception et l'application des règles, des politiques et des instructions de sécurité font partie des responsabilités fondamentales des fournisseurs de services cloud.

2.5 Cycle de vie de la composition de services

Comme illustré dans la figure 2 schématisant le cycle de vie de la composition des services Web, ces activités consistent à [20] :
1. Découvrir les services Web susceptibles de coopérer pour répondre à la requête traitée.
2. Vérifier leur compossibilité et générer les plans (ou solutions) de composition possibles.
3. Sélectionner le ou les meilleurs plans qui répondent aux exigences et préférences du client en termes de propriétés non-fonctionnelles.
4. Exécuter le service composite et le publier.
5. Publier les services composites.

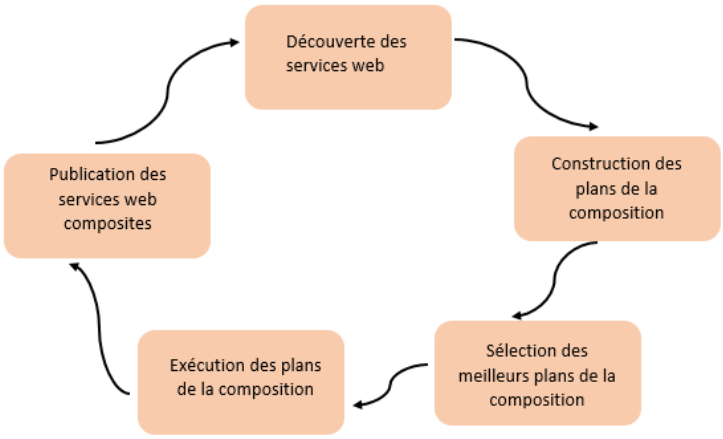

FIGURE 2.3 – Cycle de vie de la composition des services Web [20]

2.6 Processus de composition de services

Afin de définir la séquence de services nécessaire, nous devons d'abord définir les taches pour avoir une séquence de services concret pour les services composés.
Les services peuvent être combinés selon quatre modes de base : mode séquentiel (sequential mode), mode boucle (loop mode), mode conditionnel (conditional mode) et mode parallèle (parallel mode), comme indiqué dans la figure 3 [23] :

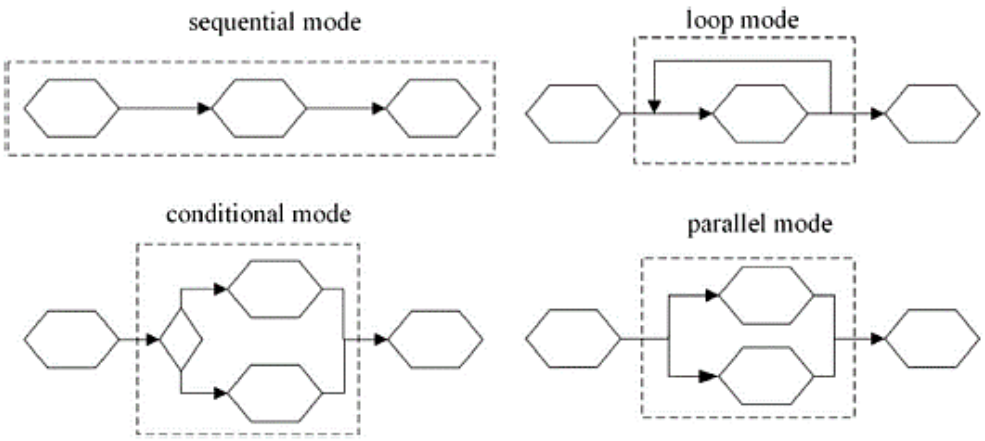

FIGURE 2.4 – Les modes de composition de services
[23]

2.7 Les étapes de composition de services

Nous savons déjà que la composition de services et un processus assez complexe, et tous processus complexe est composé de plusieurs étapes. Selon [24], nous distinguons l'existence de trois étapes : la conception et déploiement de la composition de services, l'exécution et le suivi de la composition de services, le retour d'informations sur l'évaluation de la composition de services.
La conception et déploiement de la composition de services :
Cette étape consiste principalement à construire un ensemble de services candidats. Les demandes de tâches sont fournies par les utilisateurs, et les compositions de services abstraits sont ensuite construites. La décomposition des tâches et la recherche de correspondance des services sont mises en œuvre sur la base du processus des tâches. La décomposition des tâches vise principalement à décrire les tâches et les demandes. Une tâche est décomposée en plusieurs sous-tâches selon les exigences fonctionnelles fournies par les utilisateurs (la demande de service unique est une forme spéciale de demande de services multiples).

Ce processus implique l'analyse des exigences fonctionnelles, l'analyse de la structure du processus et la construction de la composition abstraite du service. Les catégories de fonctions de service pour l'accomplissement des tâches sont identifiées sur la base des caractéristiques et des fonctions des exigences de la tâche, et le contenu spécifique du service dans la bibliothèque de services n'est pas impliqué. La correspondance de recherche de services fait correspondre la composition de services abstraits à la bibliothèque de services, qui forme un ensemble de services candidats par la recherche, la correspondance et la sélection de services. Ce processus implique le raisonnement sémantique, l'extension sémantique, la correspondance de similarité, le tri de l'ensemble de services candidats, la sélection de poids, l'évaluation des services et d'autres opérations. La sélection de poids, l'évaluation des services et d'autres opérations.
Enfin, de nombreux ensembles de services candidats avec des fonctions similaires et des qualités de service différentes sont obtenus, et chaque service candidat a des sous-tâches correspondantes.

L'exécution et le suivi de la composition de services :
L'ensemble des services candidats générés lors de la conception et du déploiement sont évalués en fonction de la qualité de service. Un algorithme intelligent est appliqué pour l'optimisation de la composition. Le chemin optimal de la composition du service est sélectionné pour répondre à la demande de tâche, et l'exécution de la tâche est surveillée. La composition optimale du service composition est initialement formée en analysant les données et les contraintes dans l'ensemble des services candidats, et chaque demande de tâche est ensuite répondue.
Des opérations, telles que le regroupement des services et la configuration des tâches, sont également impliquées dans les différents modes de tâche. Le lien de surveillance se concentre principalement sur le schéma de configuration du groupement de la transformation dynamique, le lien de service et l'appel de processus, et le processus de coordination de l'exécution de la composition du service.

Le retour d'informations sur l'évaluation de la composition de services :
Les résultats de l'exécution des tâches sont renvoyés aux utilisateurs pour évaluer les résultats de l'exécution. Les valeurs de qualité de service sont ajustées et mises à jour en fonction de la satisfaction de l'utilisateur, et les résultats de l'évaluation de la composition des services sont renvoyés à la bibliothèque de services en ligne.
En cas de résultats insatisfaisants, le système recompose les services en fonction des exigences de l'utilisateur et met à jour la capacité de connaissance et les statistiques d'évaluation en temps réel, ce qui constitue une référence pour le prochain appel de service

2.8 Les types de composition de services

La composition ou l'agrégation de services est un processus qui consiste à construire de nouveaux services appelés "service composite", en assemblant des services existants, offerts par

plusieurs fournisseurs, dans un processus de flux de travail. Ce processus spécifie quels services doivent être invoqués, dans quel ordre et sous quelles conditions préalables. Dans le processus de composition, un "service" peut être un "service atomique" ou un "service composite". La composition peut être statique ou dynamique [25].

Composition statique : Une composition statique utilise des services atomiques de manière immuable en fonction du contexte du client, ce type de composition crée des applications qui sont inflexibles et qui ne peuvent pas satisfait les exigences de clients. Cependant, il existe deux approches principales pour la composition statique (orchestration et chorégraphie).

Composition dynamique : La composition dynamique est plutôt recherchée dans les applications avec des clients de services en ligne, dont les transactions doivent être enregistrées en temps réel. La composition des services ne peut pas être prédéfinie à l'avance et se fera également au moment de l'exécution.

2.9 Les méthodes de sélection

Plusieurs services individuels sont généralement composés en un nouveau service complexe pour fournir une fonction d'agrégation. Jusqu'à présent, il existe trois méthodes typiques de composition des services cloud selon le type de sélection, à savoir la méthode de sélection locale, la méthode de sélection globale et la méthode de sélection intelligente[26] :
Sélection locale : La méthode de sélection locale permet de sélectionner, pour chaque objectif/tâche, le meilleur service parmi tous les services existants pour cette tâche. Bien que la méthode soit efficace, elle est limitée à la satisfaction d'une contrainte locale, par exemple la recherche du service parmi un nombre N de services qui satisfont aux critères de la contrainte locale[27].
Sélection globale : la méthode de sélection globale vise principalement à satisfaire la contrainte globale. Les contraintes globales sont définies comme la satisfaction de la valeur de qualité de service applicables à la composition de services individuels[27].
Sélection intelligente : Méthode d'optimisation intelligente. Il s'agit également d'une méthode d'optimisation globale. La différence réside dans le fait que le problème est non linéaire et qu'il peut être résolu à l'aide d'algorithmes d'optimisation intelligents, tels que les algorithmes génétiques (GA), l'algorithme d'optimisation par essaims de particules (PSO), l'algorithme d'optimisation par colonies de fourmis (ACO), etc [26].

2.10 Les exigences fonctionnelles et non-fonctionnelles

Les exigences fonctionnelles définissent le système de base et déterminent donc ce que le système fera et ne fera pas. Il s'agit ici de la manière dont le système réagit aux apports, ce qui se traduit généralement en pratique par une situation hypothétique, comme les calculs, la saisie de données et les processus opérationnels.
Alors que les exigences fonctionnelles déterminent ce qu'un système fait, leurs homologues non fonctionnelles définissent la manière dont il le fait. Elles sont non fonctionnelles car elles n'ont pas d'incidence sur ce que le système est en mesure de réaliser ou non fonctionnelles.
La figure ci-dessous illustre exemples des besoins fonctionnels et non fonctionnels :

Besoin fonctionnel	Besoin non fonctionnel correspondant
Le client reçoit un mail automatique après paiement de sa commande.	Les informations de l'email correspondent à sa commande. L'email arrive quelques rapidement après achat.
Le client peut modifier la quantité des articles de son panier.	Le système est suffisamment intuitif pour faciliter l'opération au client.
Le client doit entrer un mot de passe pour accéder à la zone réservée aux membres.	Les mots de passe sont stockés de manière sécurisée.
L'utilisateur est informé sur les données collectées.	Le moyen d'information est compréhensible par tous.

FIGURE 2.5 – Tableau montrant la différence entres les besoins fonctionnels et non fonctionnels [28]

2.11 Critères de Qualité de Service

La composition de service prend en considération les exigences non fonctionnelles des services pour optimiser le processus de composition à partir des critères de qualité de service.La qualité de service a été définie selon une recommandation du CCITT (Comité Consultatif International Téléphonique et Télégraphique) comme "l'effet général de la performance d'un service qui détermine le degré de satisfaction d'un utilisateur du service" [].
Le vecteur de QoS pour chaque service est représenté par **Q(Sij)= Q1, Q2,.... ,Qk ou Q(Sij)** c'est **k** attributs pour **j-ième** service concrets défini pour i-ième tâche (service abstrait) [27].
Parmi les critères de QoS nous avons [29] : Disponibilité Cela représente la valeur qui mesure si le service est accessible aux utilisateurs ; il est défini par le pourcentage de temps que le client

consomme pour accéder au service. En outre, cela représente la mesure dans laquelle le service est opérationnel

- Temps de réponse : Le plus important dans la délivrance de services est de fournir un service au consommateur dans un délai raisonnable. Le temps de réponse est mesuré en fonction de certains sous-facteurs tels que le temps de réponse moyen et le temps de réponse maximal promis par le fournisseur de services.

- La reputation : est le niveau de confiance à partir duquel le service est accepté par les utilisateurs ; il mesure la confiance gagnée par le service en fonction d'expériences antérieures

- Le coût : Il se réfère aux frais d'accès et d'utilisation d'un service que le demandeur de service doit payer. En outre, cela dépend du nombre de tâches qu'un utilisateur de service doit exécuter.

Un critère de QoS peut être un critère à maximiser ou un critère à minimiser :

- Un critère à maximiser : c'est les critères positifs qu'il faut maximiser comme la disponibilité ou la fiabilité.

- Un critère à minimiser : c'est un critère négatif qu'il faut minimiser le maximum comme le cout et le temps de réponse.

2.11.1 Fonctions d'agrégations des valeurs de QoS

Pour calculer les valeurs de QoS pour le service composé, il faut agréger les valeurs de chaque attribut de QoS dans les différents modèles de composition (séquentiel, parallèle, conditionnel, boucle).

Le vecteur de QoS agrégé pour chaque composition peut être écrit comme **Q(SC)= Q1, Q2, Q3, . . ,QK où Qk** représente le K-ième attribut agrégé pour toute composition arbitraire [27], et SC c'est le service composé. Le tableau suivant montre la fonction d'agrégation pour chaque attribut selon les différents patterns.

Critère de QoS	Séquentiel (n services séquentiels)	Parallèle (m services)	Conditionnel (appel Sij avec probabilité pr)	Boucle (service Sij boucle 1 fois)
Cout (Qco)	$\sum_{i=1}^{n} Qco(Sij)$	$\sum_{i=1}^{m} Qco(Sij)$	$\sum_{i=1}^{m} (Qco(Sij) * pr)$	l*Qco(Sij)
Temps de réponse (Qtr)	$\sum_{i=1}^{n} Qtr(Sij)$	Max(Qtr(Sij)) $1 \leq i \leq m$	$\sum_{i=1}^{m} (Qtr(Sij) * pr)$	l*Qtr(Sij)
Disponibilité (Qds)	$\prod_{i=1}^{n} Qds(Sij)$	$\prod_{i=1}^{m} Qds(Sij)$	$\sum_{i=1}^{m} (Qds(Sij) * pr)$	$Qds(Sij)^l$
Fiabilité (Qfb)	$\prod_{i=1}^{n} Qfb(Sij)$	$\prod_{i=1}^{m} Qfb(Sij)$	$\sum_{i=1}^{m} (Qfb(Sij) * pr)$	$Qds(Sij)^l$
Réputation (Qre)	$\frac{1}{n} * \sum_{i=1}^{n} Qre(Sij)$	$\frac{1}{m} * \sum_{i=1}^{m} Qre(Sij)$	$\sum_{i=1}^{m} (Qre(Sij) * pr)$	Qre(Sij)

FIGURE 2.6 – Fonction d'agrégation pour chaque attribut de QoS [30]

Sij : représente le j-ième service concret à partir de l'ensemble de services candidats Si.

2.12 Optimisation multiobjective

L'optimisation multiobjectif est un axe de recherche très important à cause de la nature multiobjectif de la plupart des problèmes réels. Les premiers travaux menés sur les problèmes multiobjectifs furent réalisés au 19éme siècle sur des études en économie par Edgeworth et généralisés par Pareto.

L'optimisation multiobjectif est un domaine fondamental de l'aide a la décision multicritère, auquel de nombreux milieux scientifiques et industriels se doivent faire face, la résolution d'un problème d'optimisation multiobjectif consiste a déterminer la solution correspondant au mieux aux préférences du décideur parmi les solutions de bonne compromis.

Dans la plupart des problèmes du monde réel, il ne s'agit pas d'optimiser seulement un seul critère mais plutôt d'optimiser simultanément plusieurs critères et qui sont généralement conflictuels. Dans les problèmes de conception, par exemple, il faut le plus souvent trouver un compromis entre des besoins technologiques et des objectifs de coût. L'optimisation multiobjective consiste donc à optimiser simultanément plusieurs fonctions. La notion de solution optimale unique dans l'optimisation uni-objective disparait pour les problèmes d'optimisation multiobjective au profit de la notion d'ensemble de solutions Pareto optimales.

2.13 Définitions d'un problème

Un problème d'optimisation est défini par :
– **un espace de recherche** (de décision) : ensemble de solutions ou de configurations constitué des différentes valeurs prises par les variables de décision .
– **une ou plusieurs fonction(s)** dite objectif(s) , à optimiser (minimiser ou maximiser)
– **un ensemble de contraintes** à respecter [31]
Dans la plupart des problèmes, l'espace d'état (décision) est fini ou dénombrable.
Les variables du problème peuvent être de nature diverse (réelle, entier, booléenne, etc.) et exprimer des données qualitatives ou quantitatives. La fonction objectif représente le but à atteindre pour le décideur. L'ensemble de contrainte définit des conditions sur l'espace d'état que les variables doivent satisfaire.
Ces contraintes sont souvent des contraintes d'inégalité ou d'égalité et permettent en général de limiter l'espace de recherche (solutions réalisables). La résolution optimale du problème consiste à trouver le point ou un ensemble de points de l'espace de recherche qui satisfait au mieux la fonction objectif. Le résultat est appelé valeur optimale ou optimum . Néanmoins en raison de la taille des problèmes réels, la résolution optimale s'est souvent montrée impossible dans un temps raisonnable.
Cette impossibilité technique impose la résolution approchée du problème, qui consiste à trouver une solution de bonne qualité (la plus proche possible de l'optimum). Il est vital pour déterminer si une solution est meilleure qu'une autre, que le problème introduise un critère de comparaison (une relation d'ordre). La plupart des problèmes d'optimisations appartiennent à la classe des problèmes NP-difficile classe où il n'existe pas d'algorithme qui fournit la solution optimale en temps polynomial en fonction de la taille du problème et le nombre d'objectifs à optimiser. Dans la littérature il existe des problèmes académiques utilisés comme des benchmarks : sac à dos, les fonctions de Schäfer, voyageur de commerce, Flowshop, . . . et des problèmes réels (applications industrielles) : télécommunications, transport, environnement,. . .

2.14 Les algorithmes génétiques

2.14.1 Introduction

Pour résoudre ces problèmes, nous utilisons des heuristiques afin de trouver la solution optimale, ou à défaut, la moins mauvaise, pour le problème.

L'idée principale des heuristiques est d'explorer l'espace des solutions en essayant de converger vers la meilleure solution. Cependant, il est important d'éviter une convergence prématurée de l'algorithme vers un extremum, ou optimum, local.

Un extremum local est la meilleure solution dans une zone restreinte, en opposition à l'extremum global, qui est la meilleure solution dans l'ensemble.

2.14.2 Principes

Les algorithmes génétiques utilisent la théorie de Darwin sur l'évolution des espèces. Elle repose sur trois principes : le principe de variation, le principe d'adaptation et le principe d'hérédité.

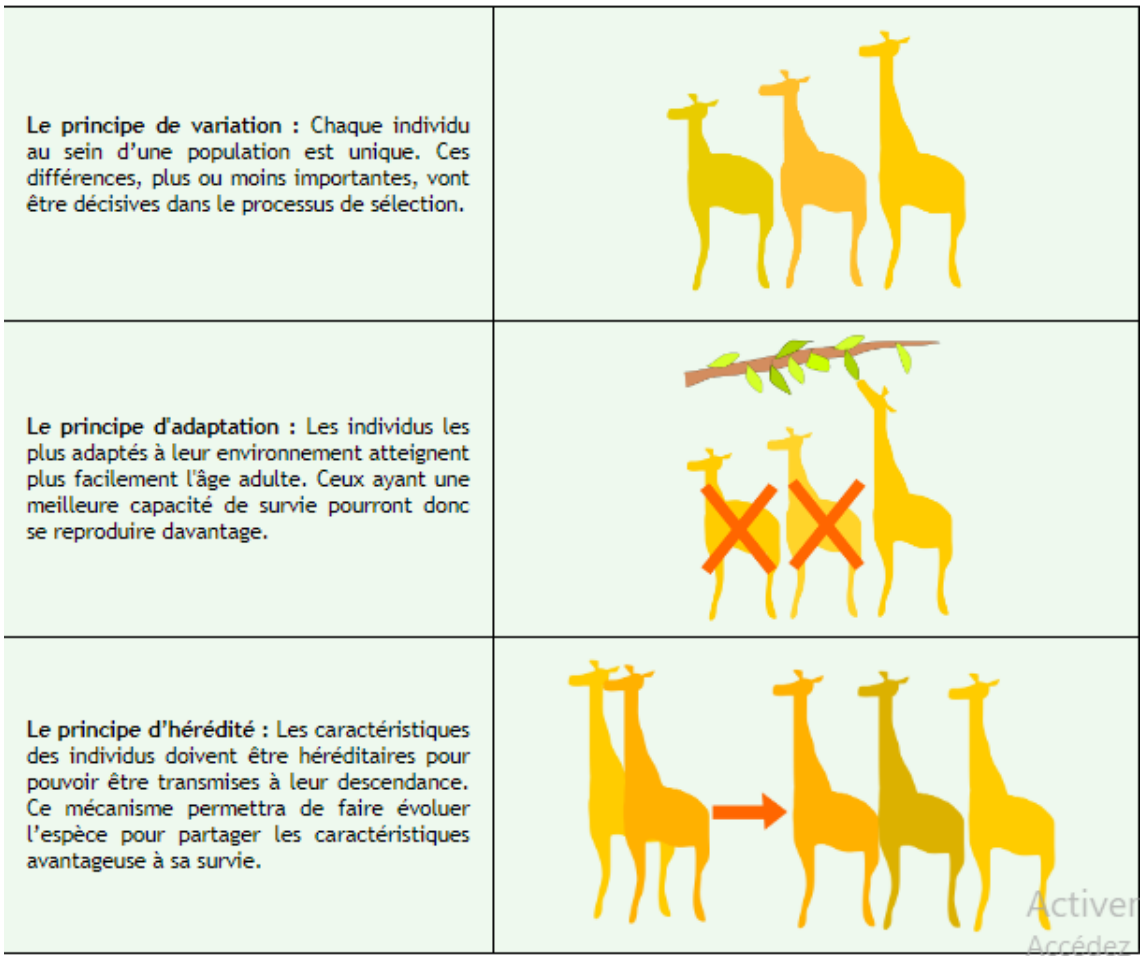

FIGURE 2.7 – Image montrant les principes de l'algorithme genetique [32]

2.14.3 Paradigmes

Ce paradigme, associé avec la terminologie de la génétique, nous permet d'exploiter les algorithmes génétiques : Nous retrouvons les notions de Population, d'Individu, de Chromosome et de Gène.

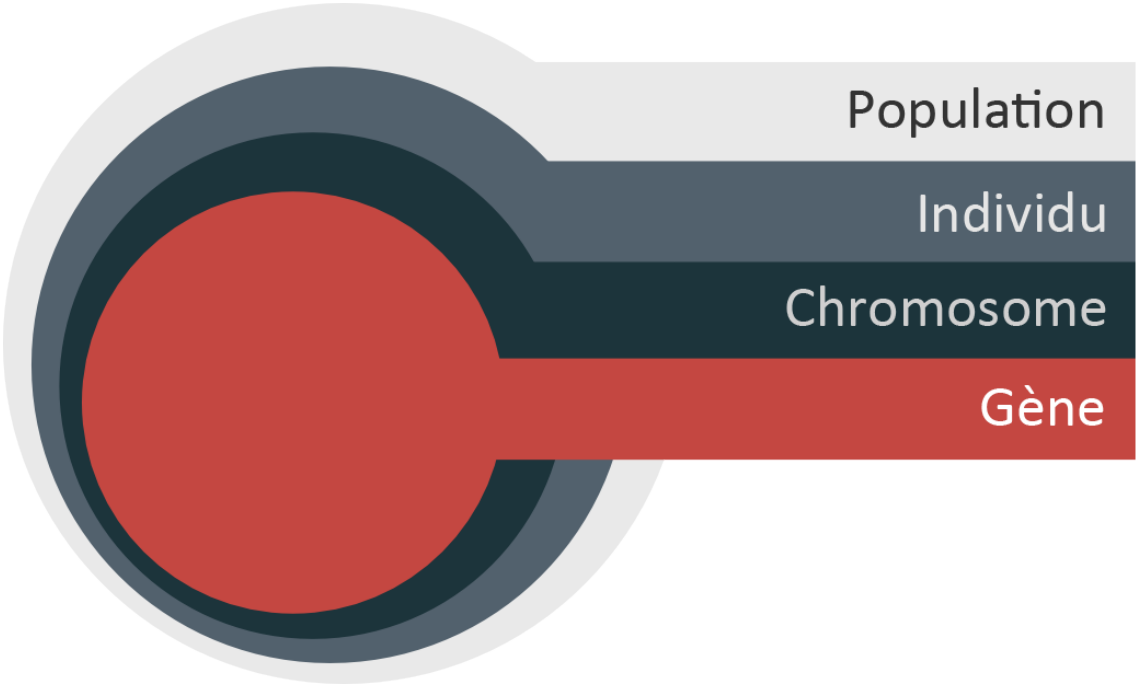

FIGURE 2.8 – Les paradigmes de l'algorithme génétique
[32]

- La population est l'ensemble des solutions envisageables.
- L'individu représente une solution.
- Le Chromosome est une composante de la solution.
- Le Gène est une caractéristique, une particularité.

Avec ces notions, nous obtenons trois opérateurs d'évolution...

2.15 Opérateurs d'évolution

Il y a trois opérateurs d'évolution dans les algorithmes génétiques selon [32] :

La sélection : Choix des individus les mieux adaptés.
Le croisement : Mélange par la reproduction des particularités des individus choisis.
La mutation : Altération aléatoire des particularités d'un individu.

Sélection :
La sélection consiste à choisir les individus les mieux adaptés afin d'avoir une population de solution la plus proche de converger vers l'optimum global.
Cet opérateur est l'application du principe d'adaptation de la théorie de Darwin.

Il existe plusieurs techniques de sélection. Voici les principales utilisées :

Sélection par rang : Cette technique de sélection choisit toujours les individus possédant les meilleurs scores d'adaptation.
Probabilité de sélection proportionnelle à l'adaptation : Technique de la roulette ou roue de la fortune, pour chaque individu, la probabilité d'être sélectionné est proportionnelle à son adaptation au problème.
Sélection par tournoi : Cette technique utilise la sélection proportionnelle sur des paires d'individus, puis choisit parmi ces paires l'individu qui a le meilleur score d'adaptation.
Sélection uniforme : La sélection se fait aléatoirement, uniformément et sans intervention de la valeur d'adaptation. Voici un exemple avec des individus en représentation binaire une fois la sélection effectuée :

Voici un exemple avec des individus en représentation binaire une fois la sélection effectuée :

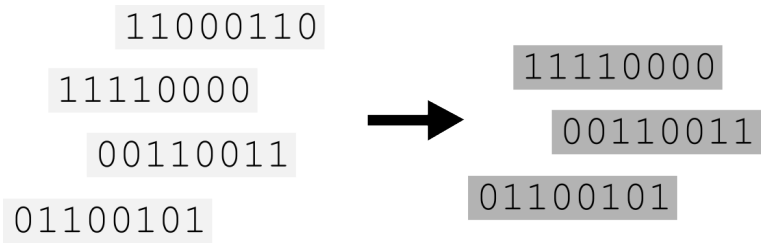

FIGURE 2.9 – Représentation binaire des individus
[32]

On réalise ensuite un croisement entre deux chromosomes parmi la population restante...
Croisement :
Le croisement, ou enjambement, crossing-over, est le résultat obtenue lorsque deux chromosomes partage leurs particularités. Celui-ci permet le brassage génétique de la population et l'application du principe d'hérédité de la théorie de Darwin.

Il existe deux méthodes de croissement : simple ou double enjambement.[32]

- Le simple enjambement consiste à fusionner les particularités de deux individus à partir d'un pivot, afin d'obtenir un ou deux enfants :

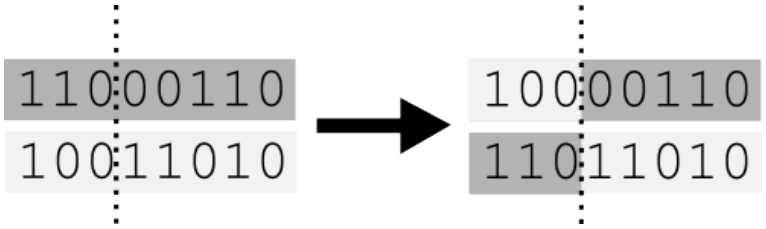

FIGURE 2.10 – Croisiement de deux individus

[32]

- Le double enjambement repose sur le même principe, sauf qu'il y a deux pivots :

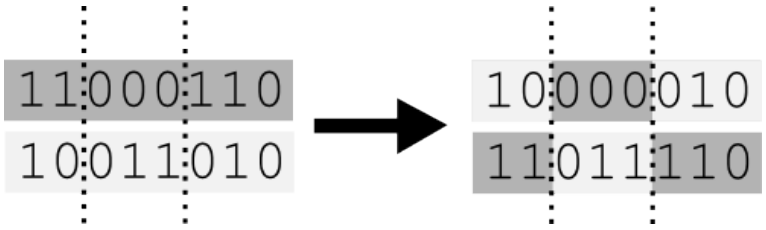

FIGURE 2.11 – Double Croisiement de deux individus

[32]

Mutation : La mutation consiste à altérer un gène dans un chromosome selon un facteur de mutation. Ce facteur est la probabilité qu'une mutation soit effectuée sur un individu.[32]
Cet opérateur est l'application du principe de variation de la théorie de Darwin et permet, par la même occassion, d'éviter une convergence prématurée de l'algorithme vers un extremum local.[32]
Voici un exemple de mutation sur un individu ayant un seul chromosome :

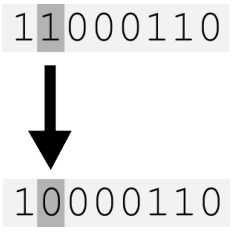

FIGURE 2.12 – Opérateur de mutation

[32]

Avec ces trois opérateurs d'évolution, nous pouvons appliquer les algorithmes génétiques.

2.15.1 Algorithme

Les principes de bases étant expliqués, voici le fonctionnement des algorithmes génétiques :

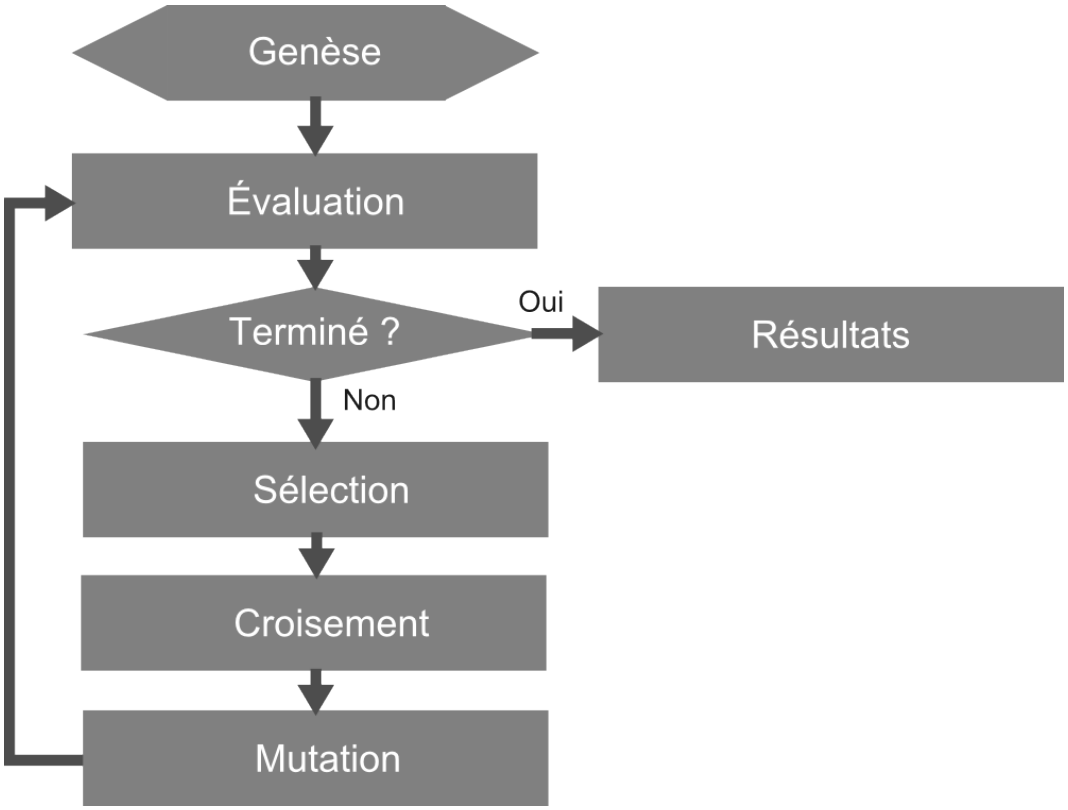

FIGURE 2.13 – Principes de base de l'algorithme
[32]

Quelques explications :

La génèse est l'étape de la création d'une population aléatoire. C'est le point de départ de notre algorithme
L'évaluation est l'analyse des individus pour analyser si une solution est disponible. Pour ceci, nous utilisons un fonction de coût, ou d'erreur, afin de définir le score d'adaptation des individus lors du processus de sélection.
Nous effectuons une boucle tant que l'évaluation estime que la solution n'est pas optimale.[32]

2.16 Conclusion

Dans ce chapitre nous avons présentés les problèmes liés à la composition de service, Nous avons aussi défini les modes, les exigences et les différentes étapes de la composition.
Dans le prochain chapitre nous allons présenter les différents travaux présents dans la littérature qui Abordent le problemes de la composition de service.

Chapitre 3

Chapitre III : Travaux connexes

3.1 Introduction

Les problèmes mentionnés dans le chapitre précèdent concernant la composition ont donnés naissances à plusieurs travaux dont le but est la résolution de la problématique de la composition dans un environnement multi-cloud. Dans ce chapitres nous présenterons ces travaux et nous discuterons les solutions proposées.

3.2 Etudes des travaux connexes de la composition de services

La composition de services est aujourd'hui un sujet de grand intérêt autant pour le monde de la recherche que pour le monde industriel. De nombreuses recherches visent à développer des modèles de composition de services et à fournir les outils nécessaires pour la composition de services [33].
Dans cette section nous nous intéressons aux travaux effectués ces dernières années, pour être en mesure de proposer une nouvelle solution optimale.

-**Kurdi et al. [34]**, ont proposé un algorithme d'optimisation combinatoire pour plusieurs cloud bases appelé COM2. Le principe de l'algorithme est de trier la liste des clouds dans l'ordre décroissant, pour assurer que le cloud avec le nombre maximal de services soit sélectionné en premier lieu. Il est clair que le résultat de l'algorithme dépend de la position des services demandés par l'utilisateur dans les clouds. En d'autres termes, si les services demandés sont dans les plus petits clouds, l'algorithme sera coûteux en terme de temps, et vice versa. La figure ci-dessous représente le pseudo code de l'algorithme COM2.

```
1: Input service request and MCE information
2: Output service composition sequence if available, otherwise, the algorithm terminates
3: Assumption clouds are sorted in decreasing order based on the number of services
4: //Initialize:
5: B ← ϕ //B is the Combiner List of clouds sorted in decreasing order of number of services
6: n ← number of clouds in the MCE
7: P ← ϕ //P is the Composer List of services
8: Get the users request R.
9: Select the cloud Cn that contains the largest number of services
10: //Check if Cn contains a new service that can fulfil the user request:
11: If ((Cn∩R) - P==ϕ) // subtracting P is important to ensure cloud contains new services
12:     then go to 20
13: Else //the cloud contains new services that are can fulfill the users request
14:     B = B + Cn  //add the cloud to the Combiner List
15:     P = P∪(Cn ∩ R) //add the services to the Composer List
16: //Check if P contains all required services that fulfill user request:
17: If (P == R)
18:     then generate composition sequence  // users request fulfilled
19: Else  // users request has not been fulfilled yet
20:     n = n - 1
21:     If (n > 0)//if some clouds in the MCE has not been checked yet
22:         then  go to 10  //check the next cloud
23:     Else //all clouds in the MCE have been checked
24:         Exit
```

FIGURE 3.1 – Pseudo code de l'algorithme COM2

L'algorithme COM2 proposé par les auteurs de ce travail, fonctionne comme suit :
• Après l'initialisations des variables (la requête de l'utilisateur et des informations concernent le multi cloud environnement), l'algorithme accepte la demande de l'utilisateur pour une composition de services et génère une suggestion en sélectionnant le premier cloud qui contient le plus grand nombre de services.

• Pour plus de simplicité, les auteurs ont supposé que les clouds sont triés par ordre décroissant en fonction du nombre de fichiers de services. Ce tri aide à identifier rapidement le cloud avec le plus grand nombre de fichiers de services.

• Par la suite, l'algorithme détermine si l'un des fichiers de cloud énumérés en premier peut répondre à la demande de l'utilisateur. Si aucun fichier de clouds n'est identifié, le cloud suivant est vérifié jusqu'à ce qu'un cloud approprié sont obtenu. Sinon, l'algorithme se termine lorsque le dernier cloud est atteint.

• Une fois un cloud approprié est trouvé, il est ajouté à Combiner List B, et ses fichiers de services sont ajoutés à la liste Composer P. Si la demande de l'utilisateur n'est pas satisfaite, le cloud suivant qui contient de nouveaux services et qui peut répondre à la demande de l'utilisateur est sélectionné.

• Pour s'assurer que le cloud sélectionné contient des services qui ne figuraient pas auparavant dans la liste des compositeurs, l'algorithme soustrait le contenu de la liste des compositeurs des nouveaux services du cloud sélectionné (Cn R). Si de nouveaux services ne sont pas disponibles, le cloud sélectionné est ignoré et le cloud suivant de la liste est pris en compte.

• Les étapes du processus sont répétées jusqu'à ce que la demande de l'utilisateur soit satisfaite. La Liste des Combinateurs est ensuite envoyée au compositeur.

Cette solution assure de trouver la solution rapidement si le service se trouve dans le cloud ayant le plus grand nombre de services car il sera sélectionné en premier, mais dans le cas contraire le temps d'exécution va augmenter très rapidement. Par contre, elle ne prend pas en compte ni l'aspect sémantique, ni le contexte ni les valeurs de QOS , ni l'aspect commercial(SLA).

Ghezouani et al. [35],ont proposé une solution qui vise à structurer les services du Cloud sous la forme d'une ontologie unifiée afin de résoudre le problème de l'hétérogénéité de description des services Cloud. C'est une approche basée sur l'ontologie pour la découverte et sélection de service, ou l'ontologie en question prend en compte les différents types de service Cloud (à savoir Saas,Paas et Iaas) , les différents mode de déploiement (à savoir Hybrid ,Public, Private,Community) et d'autre détails (voir figure ..).Pour bien comprendre l'utilité de cette ontologie nous l'avons réalisé en utilisant l'outil Protégé

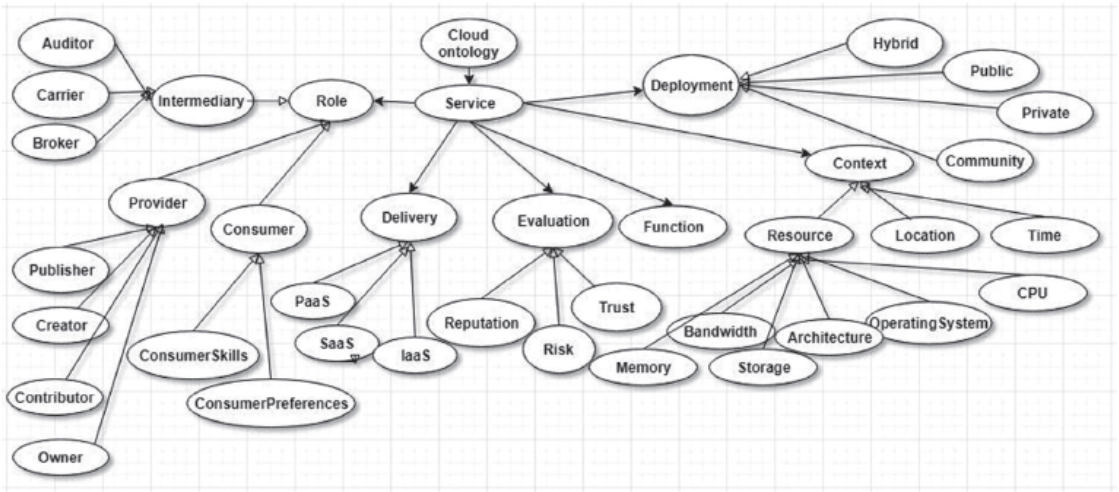

FIGURE 3.2 – Ontologie de la desription du Cloud unifié

Après avoir défini les services Cloud, les auteurs de [35] ont modélisé les QoS et les contraintes du Cloud à l'aide du Semantic Web Rule Language (SWRL) pour permettre un raisonnement sémantique lors des phases de sélection et de composition. De plus ils ont mis en place deux algorithmes, le premier pour le matching basé sur l'ontologie cloud proposée, pour assurer une sélection de service cloud sémantique.
Le deuxième algorithme est également défini pour renvoyer les séquences de composition horizontales/verticales demandées.
La figure ci-dessous représente le premier algorithme qui permet la sélection de la combinaison de service Cloud candidats :

Algorithm 1. Cloud combination selection

Input: Set of clouds $C = \{C_1, C_2, .., C_n\}$ with their hosted services, Requested services S^U with their concepts $O = \{cpt_1, cpt_2, \dots, cpt_x\}$.
Output: Cloud combination CC, Set of matched services S^M.
Assumption: Clouds are sorted in a decreasing order based on the number of hosted services.
Begin
1. $S^M \leftarrow \emptyset$
2. $CC \leftarrow \emptyset$
3. **For** each cloud $C_i \in C$ **do**
4. **If** $MatchC(C_i, C_u)$ **then**
5. $S^F = C_i \cap S^U$
6. **For** each service S_j in S_F **do**
7. **If** $MatchS(S_j, O) > T$ **then**
8. $S^M \leftarrow S^M \cup S_j$
9. $S^U \leftarrow S^U - S_j$
10. **End if**
11. **End for**
12. **If** $S^F \neq C_i \cap S^U$ **then** CC $\leftarrow CC \cup C_i$
13. **If** $S^U = \emptyset$ **then** break
14. **End if**
15. **End for**
16. Return CC
17. Return S^M

End

FIGURE 3.3 – Pseudocode de l'algorithme de sélection
[34]

L'algorithme 1 ci-dessus proposé par les auteurs fonctionne comme suit :
• Supposons que le Cloud avec le plus grand nombre de service est sélectionné en premier, l'algorithme vérifie si l'environnement cloud Ci correspond aux contrainte du client (ligne 4).

• Par la suite pour chaque service Si hébergé dans le cloud Ci satisfaisant les contraintes fonctionnelles du client (ligne 5-6) , une fonction de matching (ligne 7) est appliquée entre les concepts du service Si et les QOS définit par le client .Dans le cas où le dégrée de matching est supérieur à un seuil T (ligne 7),alors le service Si est ajouté à l'ensemble Sm des services similaires et est enlevé de l'ensemble Su (ligne 9).

• Ce concept de matching est appliqué à tous les services du cloud Ci,par la suite les services qui respectent les attentes du client sont ajoutés à l'ensemble des combinaisons CC

(ligne 12). Autrement l'Algorithme 1 va traiter le prochain cloud (ligne 3).

• La combinaison de service cloud CC est validé, si un ensemble de services similaires pour chaque tâche est déterminé (ligne 13).

• Finalement, l'Algorithme 1 retourne la combinaison de cloud CC (ligne 16) ainsi que les services candidats Sm.

Les services cloud résultant de l'Algorithme 1, sont combinés pour générer les plans de compositions susceptibles de satisfaire la requête du client. Pour ce faire les auteurs de [11] ont proposé un autre algorithme de composition illustré dans la figure ci-dessous, qui prend en entrée l'ensemble de services cloud résultant de l'algorithme 1.

Algorithm 2. Composition plans generation

Input: Candidate services $S^M = \{G_1, G_2, \dots, G_m\}$ in the cloud combination CC, User request S^U.
Output: Set of composition plans $L^P = \{P_1, P_2, \dots, P_y\}$
Begin
1. Sort clouds in CC in a descending order of the number of relevant services
2. Sort services in each cloud according to their matching degrees
3. $P \leftarrow \emptyset$
4. $L^P \leftarrow \emptyset$
5. **For** each task $t_i \in T$ **do**
6. **For** each $S_j^i \in G_i$ **do**
7. **If** $MatchComp(S_j^i, G_{i+1})$ **then**
8. add S_j^i to P
9. remove S_j^i from G_i
10. break
11. **End if**
12. **End for**
13. **End for**
14. **If** $|P| = |S^U|$ **then**
15. $L^P \leftarrow L^P \cup P$
16. $P \leftarrow \emptyset$
17. Go To Step 5
18. **End if**
19. Return L^P

End

FIGURE 3.4 – Pseudocode de l'algorithme de composition [34]

L'algorithme 2 ci-dessus proposé par les auteurs fonctionne comme suit :
• Pour commencer l'algorithme 2 trie les clouds par rapport au nombre de services candidats (ligne 1) et par rapport au degré de similarité avec la requête du client (ligne 2). Ce trie permet de combiner les services les plus pertinents venant d'un nombre de cloud minimal pour éviter l'hétérogénéité de politiques des fournisseurs.

• La prochaine étape consiste à sélectionner le meilleur service pour chaque tache ti dans l'ensemble Sm qui regroupe les services candidats pour chaque tache, et vérifie si le service Si pour la tache ti satisfait les contraintes du client (ligne 7).Dans le cas où cette condition est vérifiée le service Si est ajouté à l'ensemble de composition P (ligne 8) ,par la suite les étapes précédentes sont répété pour les taches suivantes de l'ensemble des taches (ligne 10).

• Une fois le plan de composition généré, il est ajouté à l'ensemble Lp (ligne 15). Le processus de compositions est répété pour tous les services candidats (ligne 17).

• Si les étapes précédentes n'aboutissent pas et ne sont pas suffisantes pour construire un nouveau plan (ligne 14), l'algorithme 2 retourne une liste de plan de composition (ligne 9) qui sera évalué en utilisant une fonction qui calcule le score.

Cette solution prend en compte l'aspect sémantique grâce à l'utilisation d'une ontologie pour unifier la description des services enregistrés par le fournisseur et une BDR pour le stockage des données, ainsi les valeurs de QOS.Par contre elle oblige le client à introduire les valeurs de QOS pour chaque service du service composite, ainsi que l'absence de l'aspect commercial(SLA).

- **Ali Bentaleb et al.[36]**,ont proposé une approche de composition de service cloud qui prend en considération et les QOS imposé par le client et l'environnement du cloud , cette approche est basé sur l'utilisation d'un algorithme génétique .Ils ont mis en œuvre une application Saas intermédiaire entre le client et les fournisseurs du cloud

• L'application Saas va agir comme un client et va demander les services cloud au Cloud, cela va permettre d'avoir les informations en mode offline sans impacter le temps d'exécution. L'application Saas applique l'opérateur skyline qui a pour but de définir les services qui en dominent d'autres, en d'autres termes si un service domine un autre cela veut dire qu'il est meilleur. Donc l'opérateur skyline va enregistrer les services localement en supposant que le client aura besoin d'un ou plusieurs services des services existants.

• Le client va soumettre sa requête complexe à l'application Saas qui va l'interpréter et va chercher des services candidats susceptibles de satisfaire la requête du client.

• L'application Saas va transmettre les services candidats au cloud que le cloud doit considérer. Par la suite l'application Saas va sélectionner les services en utilisant un algorithme génétique, faire la composition puis les transmettre au cloud. Ainsi Le cloud soumet sa réponse à l'application Saas quand tout est ok.

• Pour finir l'application Saas retourne le résultat de la composition au client.

Cette solution garantit la centralisation des données en faisant appel à l'application Saas qui va récupérer les données du cloud et les sauvegarder localement, ce qui va permettre de respecter l'aspect commercial (SLA) et diminuer sa violation ainsi que de diminuer le temps de réponse. Cela va faire augmenter la sécurités suite à la centralisation et va faciliter la maintenance. Par contre l'utilisation de l'algorithme génétique peut entrainer un temps de réponse relativement lent avec l'augmentation du nombre de service cloud.

- **Miao Zhang et al.[37]**,ont proposé une approche de composition de services cloud dans un environnement multi-cloud .Ils ont mis en œuvre un algorithme génétique customisé , en proposant un opérateur de mutation et de croisement amélioré .Ce qui va permettre au client d'introduire ses exigences fonctionnelles et non fonctionnelles ,pour trouver un plan de composition optimal et optimisé .

Cette solution prend en considération les valeurs de QOS indiquées par le client et les transmet à l'algorithme génétique ayant l'opérateur de mutation et croisement amélioré. Grace à l'utilisation de ses operateurs le temps d'exécutions diminue et il y a plus de chance que l'algorithme converge vers une solution optimale. Cependant plus le nombre de service augmente plus le temps d'exécution et les risque de ne pas converger vers une solution optimale augmentent. La solution ne prend pas en compte l'aspect sémantique, ni le contexte.

Guobing Zou et al [38],ont proposé une approche qui permet convertir ce problème en un ensemble de modèles de couverture, et de trouver une combinaison sous-optimale de clouds en utilisant un algorithme d'approximation. En utilisant IA planification pour composer des services Web. Plus précisément, l'idée-clé modélise la multiple base de clouds comme un arbre, et utilise un algorithme d'approximation pour optimiser la sélection du clouds, en employant une IA système de planification pour la composition du service.

Les auteurs de [38] ont décomposé leur solution en quatre algorithmes comme suit :
1-**All Clouds Combinitons :** Dans cet algorithme, ils couvrent d'abord tous les services Web impliqués dans MCB (multiple cloud bases) et la demande de composition dans un domaine de composition et un problème de composition, respectivement. Ensuite, il exécute un planificateur de composition pour trouver une solution de planification. S'il existe une séquence de plan de composition, elle est renvoyée au demandeur de service.

2-**Base Cloud Combination :** cet algorithme permet de trouver une combinaison optimale de clouds avec un nombre minimum de clouds impliqués dans la composition. Cet algorithme, énumère récursivement toutes les possibilités de combinaison de clouds jusqu'à ce qu'une solution de composition soit trouvée dans une combinaison.

3-**Smart Cloud Combination :** La méthode de Smart Cloud combinaison de clouds commence par modéliser d'un MCB en tant qu'arborescence, puis de trouver un ensemble minimum de clouds en effectuant une recherche dans l'arborescence MCB.

Cet algorithme est constitué de quatre étapes : la première étape consiste à trouver MRS (Minimum Request Set), s'il n'existe pas de combinaison de clouds satisfaisant la requête de l'utilisateur si son MRS ne peut pas être trouvé dans cette étape. Dans la deuxième étape, il faut prétraiter chaque cloud dans MCB pour réduire l'espace de ses fichiers de service et définir son coût. Plus précisément, pour chaque fichier de services sf impliqué dans un cloud C, s'il est également inclus dans MRS, alors sf est ajouté dans le cloud réduit C'. Autrement, sf est retiré du cloud réduit et le nombre de services impliqués dans sf est ajouté au coût du cloud réduit C'. Après que, tous les clouds sont transformés en leurs correspondants clouds réduits, qui sont stockés dans un ensemble réduit de réduction de clouds. Dans le même temps, chaque cloud réduit C' a un coût de cloud SN(C').
La troisième étape consiste à trouver une combinaison sous-optimale de clouds en utilisant l'ensemble réduit de clouds et les coûts de clouds réduits. Ils implémentent un algorithme approximatif en considérant le coût réduit du cloud dans le processus de sélection de clouds. Chaque cloud réduit doit être vérifier, si le nombre de ses fichiers de services inclus est plus grand que le cloud réduit maximum actuel, alors il est sélectionné comme cloud réduit maximum actuel. Pendant ce temps, si le nombre de fichiers de services est égal au cloud réduit maximal actuel.

3.3 Comparaison des différents travaux

Le tableau ci-dessous présente un résumé des différentes solutions que nous avons implémentées et examinées et que nous avons présentées en détails dans la section précédente, selon les critères suivants :

C1 : Approche proposé dans un environnement mono-cloud.
C2 : Approche proposé dans un environnement multi-cloud.
C3 : Solution qui prend en compte les valeurs de qualité de service QOS.
C4 : Solution à base de sélection locale.
C5 : Solution à base de sélection globale.
C6 : Solution qui prend en compte l'aspect commercial (SLA).
C7 : Solution qui prend en compte l'aspect sémantique (ontologie).
C8 : Solution méta-heuristique.

Approches	C1	C2	C3	C4	C5	C6	C7	C8
[33]	✗	✓	✗	✓	✗	✗	✗	✓
[34]	✗	✓	✓	✓	✗	✗	✓	✗
[35]	✗	✓	✓	✗	✓	✓	✗	✓
[36]	✗	✓	✓	✗	✓	✓	✗	✓
[37]	✗	✓	✗	✗	✓	✗	✗	✗

FIGURE 3.5 – Comparaison entre les différentes approches

Discussions et Avantages et inconvénient :
Cette étude comparative nous a permis de conclure que ces travaux ont utilisés un environnement multi-cloud pour la résolution du problème de composition de service. Le but de l'utilisation d'un environnement multi-cloud est d'être quasiment assuré de trouver le service qui répond au exigences du client qui ne cesse de devenir complexe de jour en jour, et donc un seul cloud peut ne pas satisfaire la requête du client .
Nous remarquons aussi l'utilisation de la sélection locale pour la composition dans le travail de kurdi [34] ou les valeurs de qualités de service QoS sont inexistantes ,ainsi que le travail de Ghezouani [35] qui oblige le clients à introduire les valeurs de qualités de services pour toutes les instances de service composés.
Quant à l'utilisation de la sélection globale qui est la méthode d'optimisation de sélection, qui prend en considération les valeurs de QoS globaux pour le service composé pour la composition dans les travaux [36] [37] [38],qui ont proposé des approches basés sur l'optimisation en prenant

en compte les valeurs de qualités de services dans [36] [37] mais pas dans [38] .

Le tableau suivant résume la comparaison des quelques approches en mentionnant les avantages et inconvénients de chaque une :

	Avantages	Inconvénients
A combinatorial optimization algorithm for multiple cloud service composition[34]	-Implémentation très faciles. -Code simple et concis. - Temps de réponse rapide si les services se trouvent dans le cloud avec le plus grand nombre de service.	-Absence de valeurs de QoS - Temps de réponse lent dans le cas grand nombre de cloud.
Bringing semantics to multicloud service compositions[35]	-Présence de l'aspect sémantique en utilisant une ontologie unifiée -Présence de valeurs de Qos pour la composition	-Absence de l'aspect commercial SLA. -Oblige le client à introduire les valeurs de QoS pour chaque service.
Toward Cloud SaaS for web service composition optimization based on genetic algorithm composition[36]	-Centralisation des donnés donc une meilleure maintenance et plus de sécurité.	-Temps de réponse relativement lent dans le cas d'un grand nombre de cloud.
Genetic Algorithm based QoS-aware Service Composition in Multi-Cloud[37]	-Solutions réalisables, et évolutives.	- Temps d'exécution élevé. - Convergence lente. - Haute complexité temporelle

TABLE 3.1 – Avantages et inconvénients des différentes approches

3.4 Conclusion

Dans ce chapitre nous avons présentés les travaux existant qui portent sur les problèmes de la composition des services cloud cela nous a permis de découvrir de nouvelles idées et de bien comprendre les aspects décrit dans le chapitre 2. Nous avons conclu ce chapitre avec une étude comparative pour mette en exergue les avantages et inconvénients de chaque approche.
Cette étude nous a permis de fixer les objectifs et de définir les besoins de notre travail qui sera présenté en détail dans les prochains chapitres.

Chapitre 4

Chapitre IV : Conception de la solution

4.1 Introduction

Dans le chapitre précédent, nous avons comparé et implémenté les différentes solutions présentées dans la littérature, ce qui nous a permis d'élaborer une solution basée sur les algorithmes génétiques pour la sélection globale et le calcul de similarité pour la sélection locale.
Dans ce chapitre, nous allons présenter notre démarche de travail, l'analyse des besoins ensuite nous allons présenter notre solution proposée en spécifiant l'architecture de notre système en présentant les différents scénarios possibles, ainsi que les pseudocodes.

4.2 Démarches de travail

Tout au long des différentes phases de réalisation de notre projet, nous avons adopté la méthode eXtreme Programming (XP) [1], qui est une méthode agile, utilisée en génie logiciel, plus particulièrement orientée sur l'aspect réalisation d'une application.
Cette méthode est mieux adaptée pour des équipes de tailles petites ou moyennes, des projets avec de nombreuses zones d'incertitude (risques) et lorsque les besoins sont vagues ou évoluent rapidement et nécessitent une amélioration chaque fois. L'Extreme programming a pour but principal la réduction des coûts du changement, et elle repose sur des cycles et des itérations rapides de développement dont les règles sont les suivantes [39] :

• **Planification :** consiste à diviser le projet en plusieurs itérations, et de crée un calendrier de livraison.

• **Gestion :** consiste à dédier un espace de travail au membre, organiser des réunions quotidiennement, et surveiller l'état d'avancement du projet.

1. http ://www.extremeprogramming.org./

• **Conception :** utilisation des diagrammes et des schémas pour simplifier la représentation des différentes étapes du projet, avec possibilité de les reformuler chaque fois que possible.
• **Codage :** ici un ordinateur doit être dédier pour les tests unitaires des différentes parties du code, qui doit être rédigé conformément aux normes convenues.

• **Testes :** ici tous les codes doivent avoir des tests unitaires, et tout le code doit réussir tous les tests unitaires avant de pouvoir être remit.

Le Cycle de développement : La programmation extrême repose sur des cycles rapides de développement (des itérations de quelques semaines) dont les étapes sont les suivantes :

• Une phase d'exploration détermine les scénarios « client » qui seront fournis pendant cette itération ;
• L'équipe transforme les scénarios en tâches à réaliser et en tests fonctionnels ;
• Chaque développeur s'attribue des tâches et les réalise avec un binôme ;
• Lorsque le produit satisfait aux tests fonctionnels, il est livré.

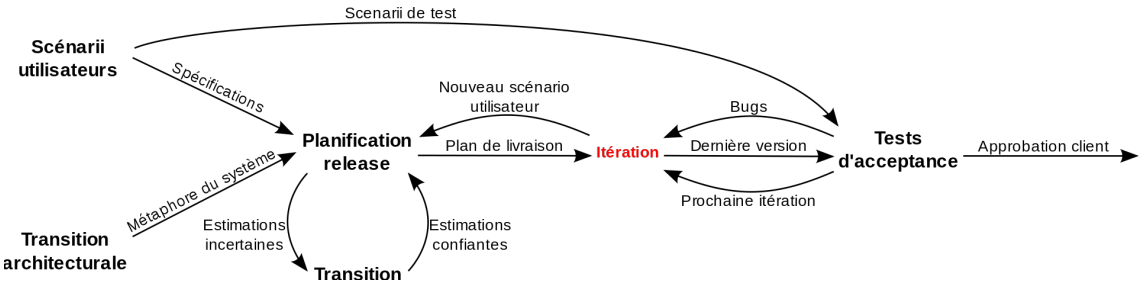

FIGURE 4.1 – Le cycle de l'Extreme Programming

[39]

4.3 Spécification des besoins

Pour spécifier les besoins de notre système, nous devons spécifier les acteurs intervenants avec ce dernier, qui sont :
• **Client :** L'acteur qui consomme les services.
• **Fournisseur des services :** L'acteur qui fournit les services.
• **Administrateur :** L'acteur qui fait la maintenance, la mise à jour de la base de données et certaines modifications.
La figure ci-dessous présente le diagramme de cas d'utilisation de notre système, qui met en avant les acteurs du système et leurs différentes interactions comme suit :

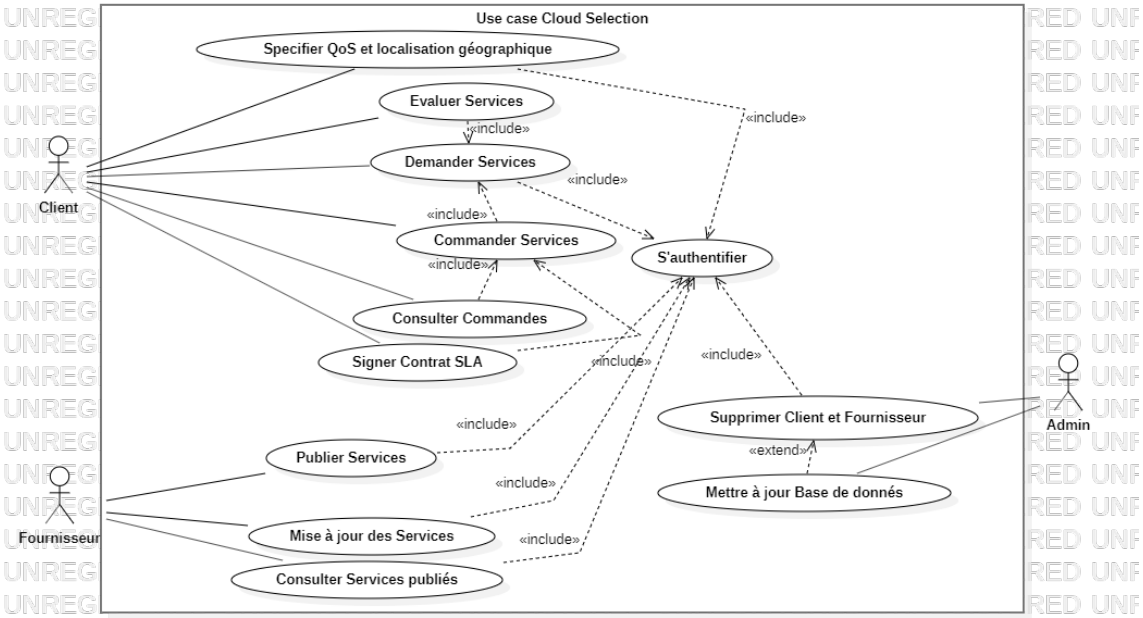

FIGURE 4.2 – Diagramme de cas d'utilisation du système

-Le client doit s'authentifier pour pouvoir demander, commander et consulter les services publiés par les fournisseurs.

-Le fournisseur de service cloud doit d'abord s'inscrire dans le système, puis il pourra s'authentifier et publier des services avec leurs informations. Il pourra aussi mettre à jour les services donc les modifier ou les supprimer.

-L'admin du système a pour but de vérifier les services, les fournisseurs et les clients présent dans la base de données, il pourra donc les modifier ou les supprimer.

4.4 Processus de composition général

Après l'étude et l'analyse des solutions existantes dans le chapitre précedents, nous pouvons désormais définir notre architecture de système. En définissant les étapes nécessaires pour la composition de services cloud.
Notre approche se compose de plusieurs étapes, on commence par l'introduction des services par les fournisseurs où ils doivent spécifier les valeurs de qualité de service QoS et la localisation géographique pour chaque service,par la suite on doit récuperer et définir les taches nécessaires pour la réalisation de la requete du client.Une fois les taches nécessaires définies on doit spécifier les services qui réalisent chaque tache.
Par la suite on doit normaliser les valeurs de QoS des services qu'on a specifié à partir de la requete du client où nous devrons calculer la distance géographique pour chaque service.On

elabore la sélection avec deux algorithme un avec la sélection locale et un avec la sélection globale pour élaborer la composition.
Nous avons modélisé le problème de composition comme suit :

4.4.1 Introduction des services par les fournisseurs

Les fournisseurs introduisent leurs services à partir d'une interface fournisseur, en respectant la description unifié grâce à l'utilisation d'une ontologie de description et son matching avec la base de donnée relationnel (pour éviter l'hétérogénéité entre les services).

4.4.2 Récupération et définition des taches nécessaires pour la réalisation de la requête du client

Le client introduit sa requête à partir d'une interface client, en spécifiant la fonctionnalité du service qu'il désire, les valeurs de qualité de service QoS ainsi que le contexte qui est sa localisation géographique.
Une fois la requête du client récupéré, le système va extraire les tâches **Ti ; i= 1, 2, …. , n**, avec **Ti** représente la i-ieme tâche nécessaire pour la réalisation de service composé et n représente le nombre des tâches qui sont nécessaires pour la réalisation de la requête du client.

4.4.3 Définir les services pour réaliser chaque tache

A partir des services publiés par les fournisseurs, le système classifie la liste des services par tache ou chaque tâche peut être réalisé par un seul service de la liste des services candidats.
Donc chaque tâche Ti représente un ensemble des services qui ont la même fonctionnalité **Sij ; Si1, Si2, ..., Sij, …, Sim**, avec **Sij** représente le **j-ieme** service qui réalise la i-ème tâche et m c'est le nombre des services pour chaque tâche. La tâche Ti nécessite un seul service Sij pour être réalisé. Les services qui réalisent la même tâche peuvent avoir de différentes valeurs de qualité de service QoS.

4.4.4 Normalisation des services en fonction de la requête du client

Chaque service publié par un fournisseur contient sa localisation géographique, le lieu où le service en question est stocké. Donc après avoir récupéré la requête du client nous pouvons en extraire la localisation géographique du client qui va nous permettre de calculer la distance entre la position de l'utilisateur et la position du service, pour assurer de sélectionner les services les plus proches du client en terme de distance.

4.4.5 Élaboration de la sélection

Une fois que les services pour la réalisation de chaque tache sont spécifiés, un algorithme de sélection est exécuté pour sélectionner un plan de composition qui contient une séquence des services où chaque service réalise une tâche selon les exigences posées par le client.
La sélection va permettre de trouver la séquence des services nécessaires pour exécuter le service composé SC, en respectant les exigences de QoS et la localisation géographique imposés par client avec :
SC= S1j,S2j,...,Smj avec **Sj** c'est le **j-ième service** qui réalise la **i-ième** tâche et **m** le nombre des tâches nécessaires pour exécuter le service compose **SC**.

4.4.6 Élaboration de la composition

Après la sélection et la génération des plans de composition, le système évalue les plans de composition et retourne le meilleur plan au client en prenant en compte les valeurs de qualité de service QoS et la localisation géographique.
Nous avons proposé cette solution en créant l'ontologie qui nous permet d'ajouter l'aspect sémantique. Nous allons par la suite faire le matching entre ce qui se trouve dans l'ontologie et la base de données.
La figure ci-dessous représente l'architecture globale de notre système.

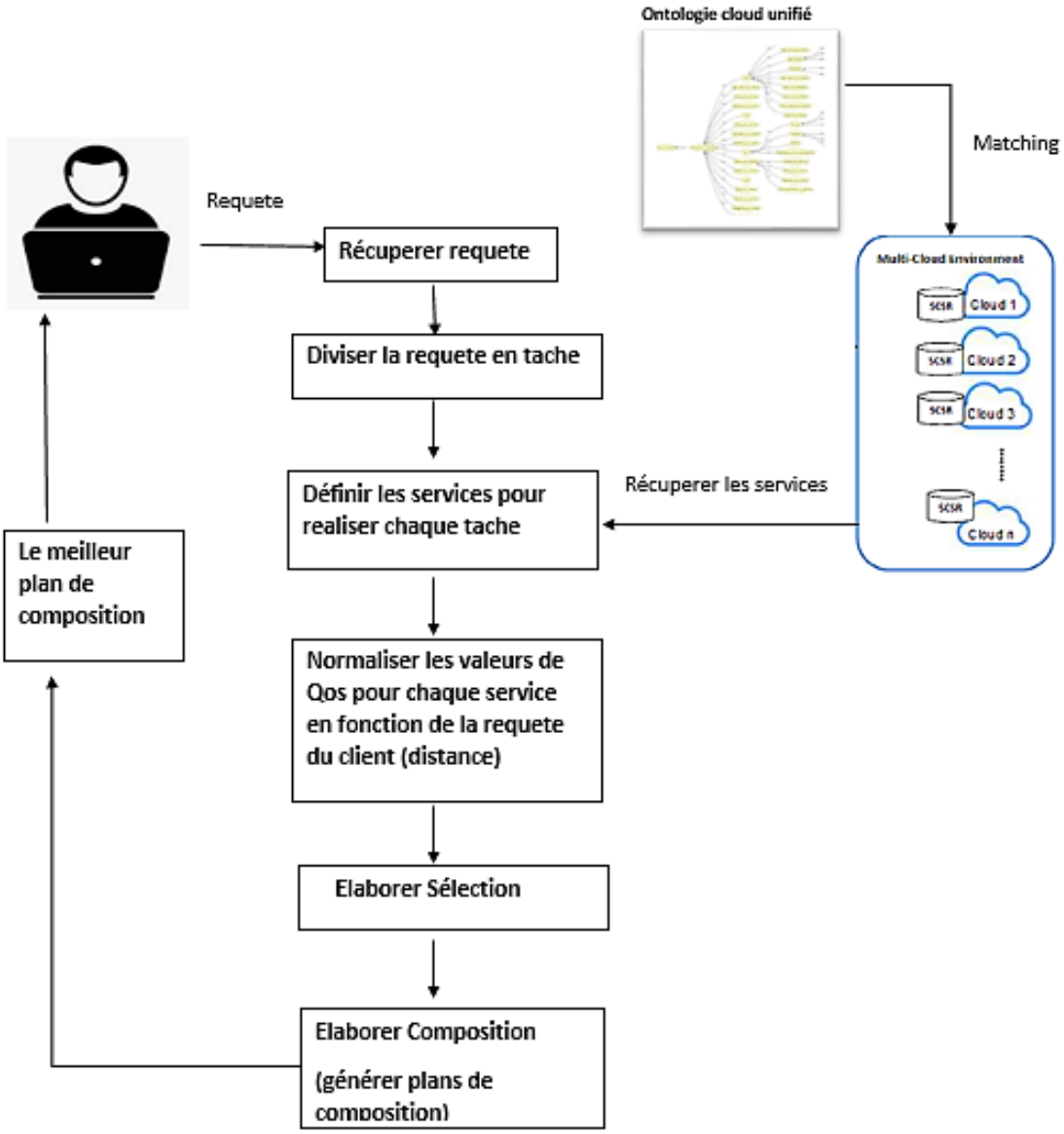

FIGURE 4.3 – L'architecture globale du système de composition.

4.5 Définition des critères de QoS

Nous avons défini les valeurs de qualités de service nécessaires pour cette solutions, et l'optimisation de ces valeurs entraine l'optimisation de la solution .
Un services Sij est défini avec ses propriétés non fonctionnelles (critères de QoS) ou pour chaque service Sij nous définissons un vecteur **Qk(Sij)** ; **k= 1, 2, . . ., q** avec Qk représente le k-ieme critère de QoS et q représente le nombre des critères de qualité de services à utiliser.

Dans notre Système nous avons utilisé quatre attributs de QoS **Qk(Sij)**, **k= 1,2,3,4,5** Avec **q=5** ainsi qu'un critère de contexte qui est la localisation géographique et : **Q1(Sij) : représente le cout(prix). Q2(Sij) : représente le temps d'exécution. Q3(Sij) : représente la disponibilité.**

Q4(Sij) : représente la réputation. Q5(Sij) : représente la Localiation géographique.

Les attributs de QoS sont catégorisés en critères positifs à maximiser (disponibilité et réputation), ou bien des critères négatifs à minimiser (temps d'exécution, cout et distance géographique).

La figure suivante illustres les catégories de QoS utilisé dans notre système :

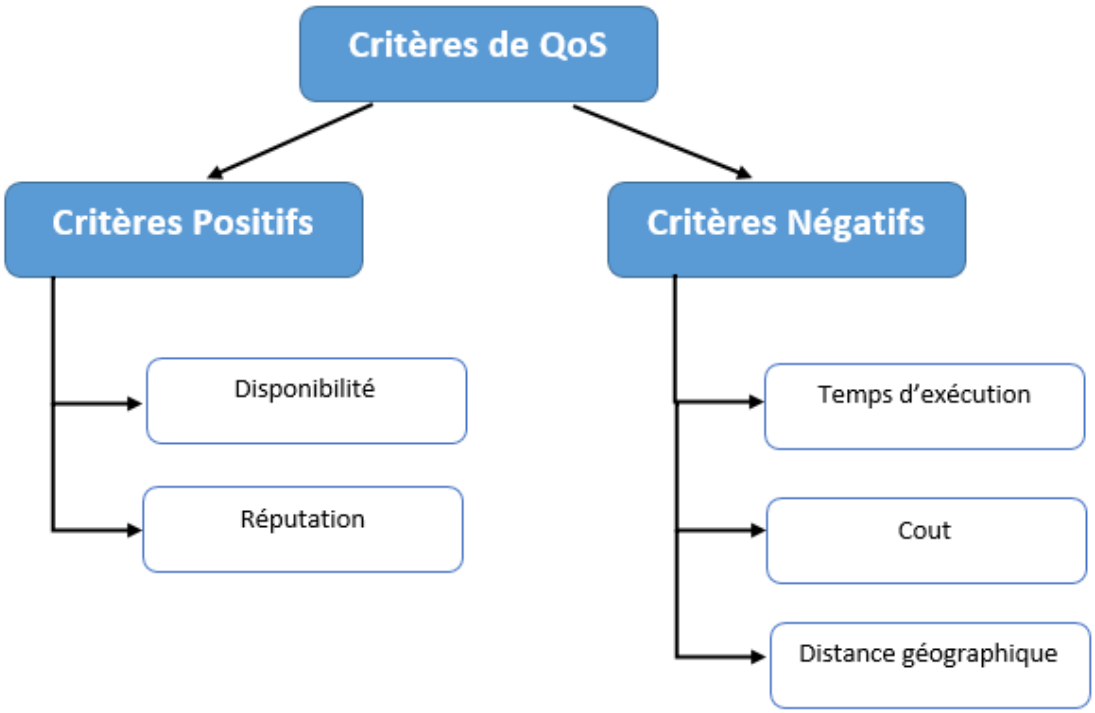

FIGURE 4.4 – Types de critères de QoS utilisés.

4.6 Correspondance entre la requête et les services disponibles

Pour définir la correspondance entre la requête utilisateur et les services disponibles, le système propose une liste bien définie des services disponibles après avoir effectué le matching avec l'ontologie de description :

- Pour commencer le système présente une interface utilisateur qui contient les services disponibles qui peuvent être réalisés par la composition.
- Par la suite une correspondance entre la requête et les services qui réalisent chaque tâche qui satisfait la requête d'un service composé. Par exemple un check box qui fait référence vers le service nommé « voyage organisé » le système conclu automatiquement que le service est composé des tâches réservation d'avion, réservation de bus et réservation d'hôtel.

- Les fournisseurs des services sont obligés de publier les services, en respectant la description unifiée, où le système informe les fournisseurs des services de la liste des tâches (propriétés fonctionnelles), afin de les publier en toute sécurité.

4.7 Les scenarios de composition

Notre système prend en charge deux types de scenarios, un dans le cas d'une sélection locale et un autre dans le cas d'une sélection globale.La séléction est l'étape la plus importante pour effectuer la composition ,car elle va permettre de choisir les meilleurs services satisfaisant les exigences de QoS et la localisation géographique exigés par le client.

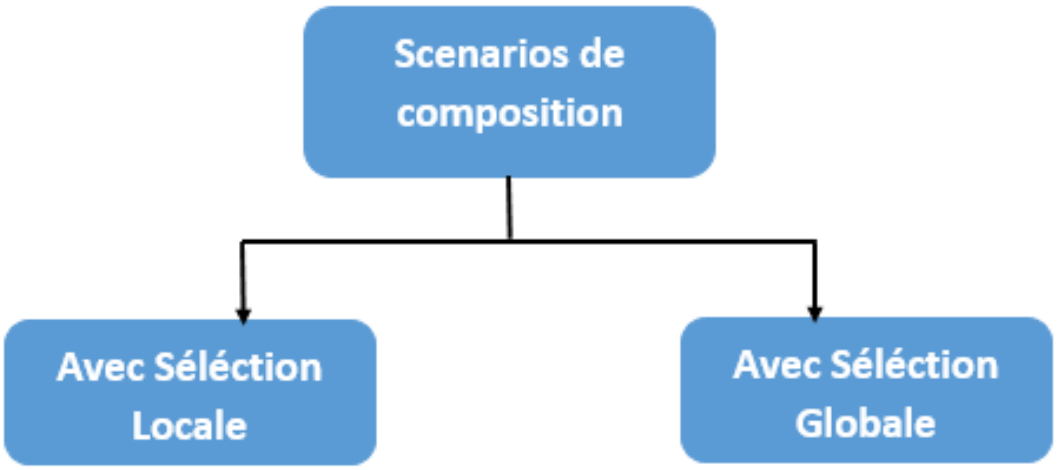

FIGURE 4.5 – Les scenarios de compositions

4.7.1 La sélection locale Scénario 1

Supposons que le client souhaite réserver un avion et un bus, tout en spécifiant les valeurs de QoS et de contexte (Localisation géographique) pour chaqu'un des deux separement. Dans ce cas nous parlons de sélection locale.
Le figure si-dessous représente le pseudo-code de l'algorithme 1 de la sélection locale :

Algorithm 1: Sélection Locale

Result: Plan de composition
C_n : L'ensemble des Clouds ordonné, ou n est le nombre de Clouds ;
Req : La requête du client pour une fonctionnalité donnée;
Précondition: Nous supposons que les Clouds sont triés par ordre décroissant du nombre de services qu'ils contiennent;
ListeServicesTrouvés: La liste des services correspondants à la requête de l'utilisateur.;
$n := 0$;
LS := Liste des services de la fonctionnalité qu'a spécifié l'utilisateur dans le Cloud C_n;
while *Pas trouvé de service dans* ***LS*** *de* C_n *avec* ***Req*** **do**
    CalculerSimilarité(**Req**, **LS**);
    if *CalculerSimilarité(**Req**, **LS**) = RemplitCondition* **then**
        ServiceTrouvé = vrai;
        Ajouter(ListeServicesTrouvés, ServiceTrouvé);
    else
        $n := n + 1$;
    end
end

FIGURE 4.6 – Pseudo code sélection locale .

Pour effectuer cet algorithme nous nous sommes inspirés des solutions proposées par [35], l'idée clé de ces deux solutions et de commencer la sélection par le cloud qui contient le plus grand nombre de service. Nous avons gardé le même principe en y ajoutant l'aspect sémantique qui est la localisation géographique, pour s'assurer de proposer au client des services qui soient le plus proches de lui.
Les étapes de l'algorithme sont comme suits :
- On commence par ordonner les cloud par ordre décroissant du nombre de service disponible, pour pouvoir parcourir le cloud qui contient le plus grand nombre de service en premier .Cela va nous permettre d'éviter l'hétérogénéité des politiques des fournisseurs ,car on risque de trouver les cloud correspondant dans le premier cloud et donc gagner beaucoup de temps.

- Par la suite ,on va récupérer la requête du client contenant ses contraintes, et suivant la requête du client nous récupérons la liste des services exécutant la tache souhaitée par client.

-On normalise les valeurs de qualité de service QoS pour chaque service en prenant en compte la localisation géographique exigée par le client et la localisation géographique du service en question , donc une distance sera calculé pour chaque service.

-Une fois les valeurs de QoS de chaque service correspondant à la fonctionnalité désirée par le client une fonction de similarité est appliqué entre le service et les contraintes du client comme suit :

$$\mathbf{S}im(X,Y) = \sum_{i=0}^{n} Wi * Sim(Xi,Yi).....(1)$$

Cette équation (1) fait la somme des similarités calculé dans (2) .Ici Wi représente le poids assigné par l'utilisateur appartenant à l'intervalle [1,0] pour chaque critères Xi. L'équation (2) quant à elle calcule la similarité entre le critère Xi du service donné et le critère Yi de la requete du client comme suit :

$$Sim(Xi,Yi) = p.(\frac{(\alpha(Xi)) \cap (\alpha(Yi))}{(\alpha(Xi))} + (1-p).(\frac{(\alpha(Xi)) \cap (\alpha(Yi))}{(\alpha(Xi))}.....(2)$$

Ou p est une valeur aléatoire appartenant à l'intervalle [1,0], a(Xi) et a(Yi) est le nombre de critère disponible dans le vecteur du service et de la requête client, quant à (a(Xi)inter a(Yi)) est le nombre de critères communs entre la requête du client et le service donné.

-On refait les étapes précédentes jusqu'à trouver trois service qui répondent au contrainte du client et on l'ajoute à la liste des plans des plans de compositions, si ce n'est pas le cas on passe au prochain cloud en incrémentant la variable n.

-L'algorithme est exécuté autant de fois que le client à spécifier de fonctionnalités. A la fin de l'algorithme nous avons une liste de composition à laquelle on va calculer le score avec l'équation suivante et on va la retourner au client avec l'équation (3) comme suit :

$$ScoreS(Si) = W1 * Cr1 + W2 * Cr2 + W3 * Cr3 + ...Wn * Crn.....(3)$$

Ce score est calculé pour chaque service composant le service composé pour chaque tache, par la suite on calcule le score pour chaque plan de composition avec l'équation (4) suivante :

$$Sim(X,Y) = \frac{1}{n} * \sum_{i=0}^{n} ScoreS(Si).....(4)$$

Enfin, les plans de composition candidats sont triés par ordre décroissant de leur score, et le plan avec le meilleur score est renvoyé à l'utilisateur du cloud.

4.7.2 Exemple de scénario 1

Supposons que notre client souhaite avoir deux fonctionnalités dans notre application et que les valeurs de QoS qu'il a introduit avec leurs poids de préférences respectifs sont :

Disponibilité	Temps d'execution	Réputation	Prix
0.43 , avec un poids de 0.8	8.39 s, avec un poids de 0.9	3.58, avec un poids de 0.98	60.81 euros, avec un poids de 1

TABLE 4.1 – Exemple de la première fonctionnalité du scénario 1

Disponibilité	Temps d'execution	Réputation	Prix
0.7 , avec un poids de 0.87	30 s, avec un poids de 0.95	4.9, avec un poids de 0.84	40 euros, avec un poids de 0.81

TABLE 4.2 – Exemple de la deuxième fonctionnalité du scénario 1

Notre client a aussi introduit sa localisation géographique (Pour cet exemple, nous supposons qu'il se trouve en Algerie)
Après la normalisation des valeurs des besoins fonctionnels, nous calculons la distance entre la localisation du client, et la localisation de tous les services de la fonctionnalité qu'il a choisi.
Nous nous retrouvons alors avec les mêmes valeurs de QoS avec en plus, comme critère la distance qui dépendra du lieu ou le service se trouve.
Nous calculons donc la distance entre le client et où se trouve le service. Ce qui nous donne dans notre cas 0, (Distance entre 'Algerie' et 'Algerie') Nous nous retrouvons donc avec les mêmes valeurs de QoS précédentes avec en plus, la distance comme critère :

Disponibilité	Temps d'execution	Réputation	Prix	Distance
0.43 , avec un poids de 0.8	8.39 s, avec un poids de 0.9	3.58, avec un poids de 0.98	60.81 euros , avec un poids de 1	0 km, avec un poids de 1

TABLE 4.3 – Exemple de la première fonctionnalité du scénario 1 avec distance

Disponibilité	Temps d'execution	Réputation	Prix	Distance
0.7 , avec un poids de 0.95	30 s, avec un poids de 0.85	4.9, avec un poids de 0.84	40 euros , avec un poids de 0.81	0 km, avec un poids de 1

TABLE 4.4 – Exemple de la deuxième fonctionnalité du scénario 1 avec distance

Nous allons ensuite calculer la similarité entre la requête du client et les services, en utilisant l'équation **(1)**.

Nous trouvons alors deux services **S1** et **S2** pour chacune des fonctionnalités avec des valeurs de similarité suivantes :

$$Sim(Req1Client, S1) = 8.07$$

$$Sim(Req2Client, S2) = 9.19$$

Nous calculons par la suite le score de ce plan de composition contenant le service **S1** et **S2** en utilisant l'équation **(4)**, nous obtenons alors :

$$ScoreS(S1/S2) = 1.07$$

Pour finir, nous comparons les valeurs de QoS des services que nous avons obtenus avec la requête du client :

Pour le service **S1** :

Disponibilité	Temps d'execution	Réputation	Prix	Distance
0.84	11.05 s	4.25	56.11 euros	0 km

TABLE 4.5 – QoS de S1

Pour le service **S2** :

Disponibilité	Temps d'execution	Réputation	Prix	Distance
0.79	36.94 s	4.64	36.22 euros	0 km

TABLE 4.6 – QoS de S2

C'est donc ce plan de composition qui est effectivement le meilleur qu'on puisse obtenir, et ceci, on peut le déduire des valeurs obtenues qui peuvent être meilleures que ce que l'utilisateur a demandé.

4.7.3 La sélection globale Scénario 2

Supposons que le client souhaite le service voyage organisé qui est un service composé de trois taches (réservation avion, réservation bus, réservation hôtel) et qu'il désire introduire les valeurs de QoS et de contexte (Localisation géographique) une seule fois, on parle ici d'une sélection globale.
Pr exemple un client souhaite un service voyage organisé avec un cout qui ne doit pas dépasser 150 €, étant donné que c'est un critère à minimiser.
Si un plan de composition contient les trois services (Réservation Avion, Bus,Hotel) qui réalisent les tâches du service demandé et les couts de chaque service sont (80,30,70) respectivement.
Si nous calculons le cout du service global selon la formule définie dans le tableau (chapitre 2) nous trouvons que le cout est égal à 180 €, et donc cela dépasse le cout exigé par le client qui est égale à 150 €. Si nous remplaçons le service de réservation d'avion avec un autre service pour la réservation d'avion qui a un cout de 40€,alors le cout cumulé devient 140 € qui ne dépasse pas 150 €,donc qui satisfait l'exigence de client.

4.8 Mesure de performance du service composé

La mesure de performance d'un service composé est très importante dans la sélection globale, pour identifier les services composés qui répondent aux exigences du client. Et pour cela, il faut une normalisation des QoS, qui se fait comme suit :
- Pour mesurer la performance d'un service composé nous utilisons les valeurs de QoS des services atomiques dans le service composé.
-Nous commençons par le calcul des valeurs de QoS cumulé du service composé en utilisant le tableau du chapitre 2.
-Par la suite nous avons besoin de normaliser les valeurs de QoS selon les équations suivantes :
- Pour les critères positifs : (5)

$$QNn(SC) = \frac{Qmax(n) - Qn(SC)}{Qmax(n) - Qmin(n)} \text{.....(5)}$$

- Pour les critères négatifs : (6)

$$QNn(SC) = \frac{Qm(SC) - Qmin(m)}{Qmax(m) - Qmin(m)} \text{.....(6)}$$

4.9 La fonction fitness

Après la normalisation des valeurs de QoS du service composé nous pouvons désormais mesurer la performance du notre service composite en utilisant la fonction du fitness. La fonction du fitness calculé avec la formule suivante décrite dans [29] : (7)

$$f(SC) = \sum_{i=i}^{2} QNni * Wi + \sum_{j=i}^{2} QNmi * Wj \text{.....(7)}$$

Avec wi et wj représentent les poids pour chaque critère de qualité de service posé par l'utilisateur avec :

$$\sum wipi = 1$$

Un algorithme d'optimisation permet de trouver le plan de composition qui répond aux demandes des clients. Le problème est encodé selon un problème d'optimisation pour permettre l'exécution de cet algorithme, afin de minimiser le cout, le temps d'exécution et la distance géographique ainsi que de maximiser la disponibilité et la réputation.
Avant l'exécution de l'algorithme, un encodage pour faire la correspondance entre le regroupement des services pour chaque tâche et le lien qui fait la référence pour les services dans la liste pour chaque tâche afin de choisir des combinaisons des services composites qui sont candidats d'être choisi dans le processus de sélection.
Dans notre algorithme, une solution candidate (individu) est encodé par un tableau d'entiers, ou chaque entier fait référence vers un service concret dans l'ensemble des services pour chaque tâche. La figure suivante représente l'encodage du problème, en d'autres termes la création d'individu pour l'algorithme génétique de la sélection globale :

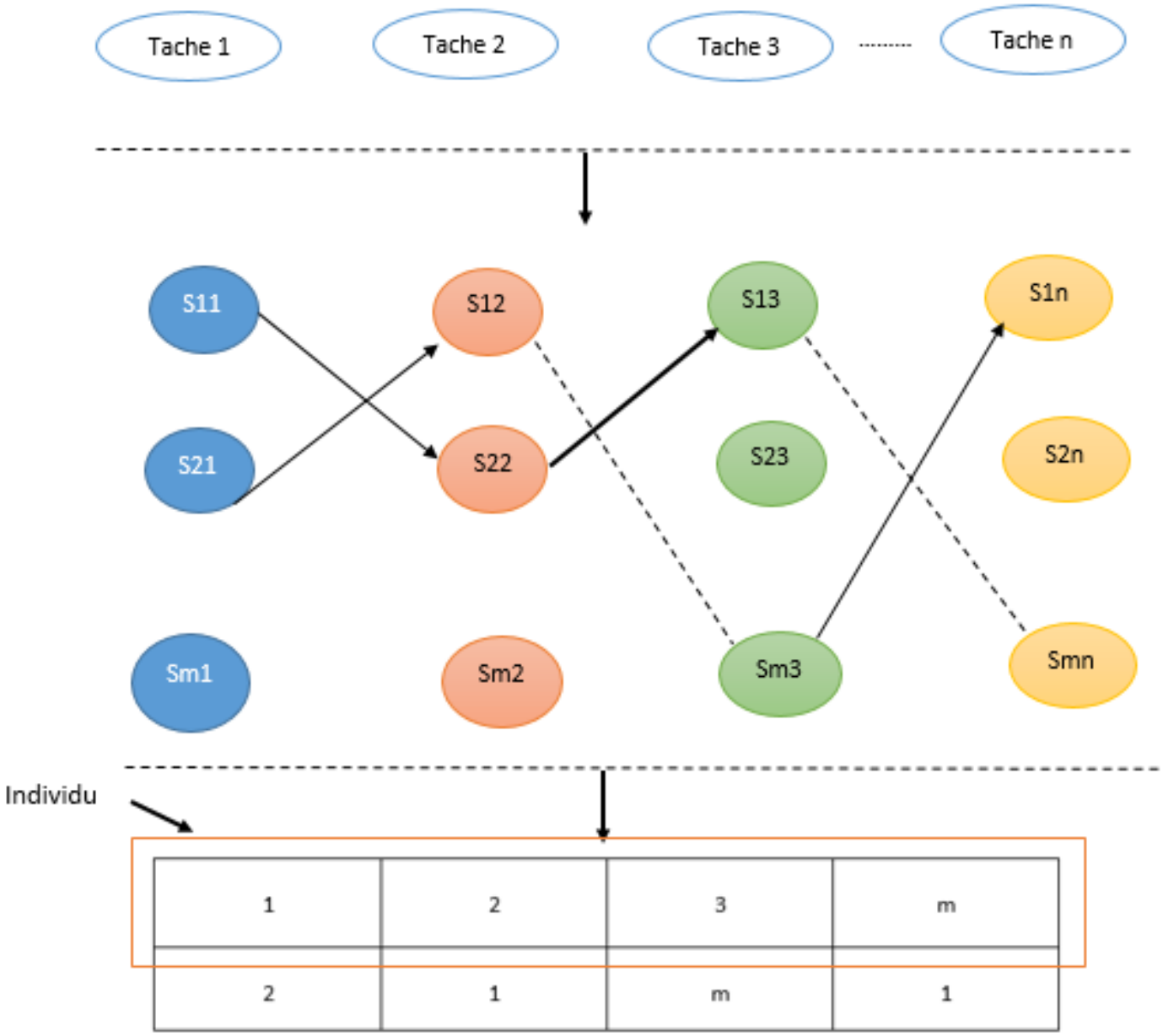

FIGURE 4.7 – Encodage du problème en individu

L'idée de l'algorithme pour la sélection globale a été inspiré de la solution proposé dans [29] qui est un algorithme méta-heuristique d'optimisation, notre amélioration a été de diviser les services de chaque tache en groupe pour minimiser le temps d'exécution. Le pseudo code suivant montre les étapes de notre algorithme pour la sélection locale.

Algorithm 1: Sélection Globale

Result: Plan de composition
C_n : L'ensemble des Clouds
Req_m : La requête du client ou m est le nombre de fonctionnalités;
ListeServicesTrouvés: La liste des services correspondants à la requête de l'utilisateur.;
LS := Liste des services de la fonctionnalité qu'a spécifié l'utilisateur dans le Cloud C_n;
begin
fitness := 0;
while *Condition de convergeance non satisfaite* **do**
  DécomposerServices(LS_i, nombreAléatoire);
  NormaliserValeursQoS(ValeursQoS(LS_i))
  CalculerFitness(ValeursNoramlisées)
  trierPlans(planDeComposition);
  if *planDeCompositionTrouvé = vrai* **then**
    Ajouter(PlanDeComposition, **LS**); ConditionDeConvergeanceSatisfaite = vrai;
    return listeServicesTrouvés;
  else
    i := i + 1 ;
    croiserPopulation(LS_i, opérateurAléatoireDeCroisement);
  end
end

FIGURE 4.8 – Pseudocode de l'algorithme de sélection globale

Les étapes de l'algorithme se présente comme suit :
-D'abord on commence par récupérer les ids des services pour chaque tâche ainsi que la requête du client.
-Par la suite on décompose les ids des services par groupes et on génère les plans de composition pour les premiers groupes qui vont représenter la population initiale.
-On normalise les valeurs de QoS en utilisant les équations (5) et (6) en fonction de la requete du client avec ses valeurs de QoS et la localisation géographique.
-Par la suite on calcule la fonction fitness pour chaque individu en utilisant l'equation (7) avec les valeurs normalisées auparavant.
-On trie les plans de composition par ordre croissant selon la valeur de la fonction du fitness pour chaque individu.
- On vérifier si les conditions de convergence sont vérifiées donc s'il existe une solution qui satisfait la demande du client, si oui l'algorithme termine son exécution et il retourne l'individu qui a la meilleure valeur de fonction du fitness, sinon l'algorithme continue son exécution en allant vers l'étape suivante.

-L'étape suivante consiste à passer au deuxièmes groupes et d'améliorer les premiers plans de composition en appliquant un opérateur de croissement, comme illustré dans la figure suivante :

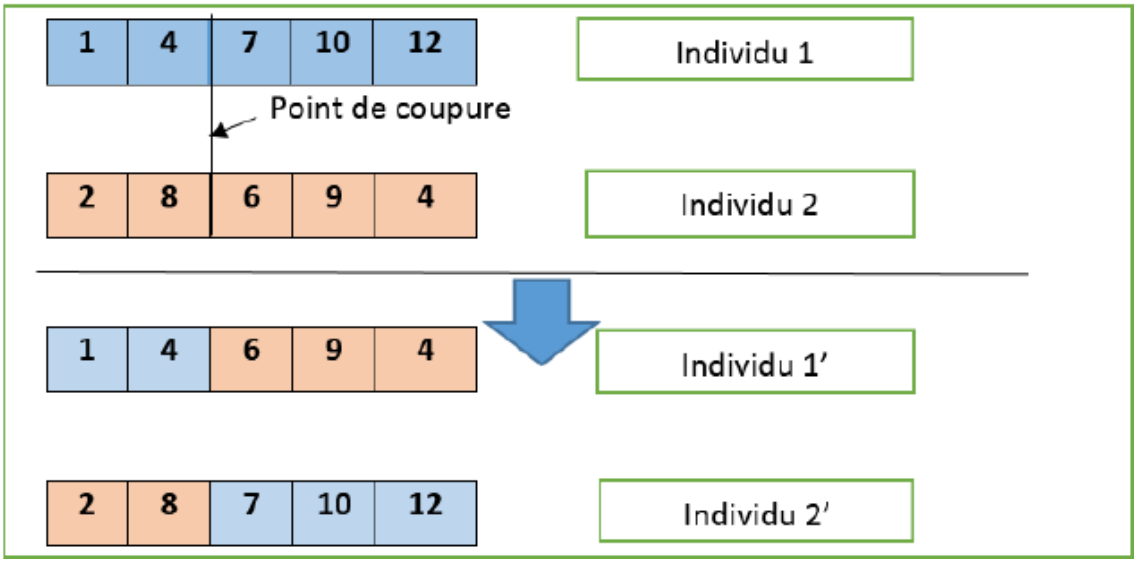

FIGURE 4.9 – Operateur de croisement

-Par la suite on refait les étapes de fonction fitness et d'évaluation du plan de composition, on revérifie s'il existe un plan de composition qui satisfait la requete du client , si c'est le cas l'algorithme termine son exécution et il retourne l'individu qui a la meilleure valeur de fonction du fitness, sinon retourne à la phase de croisement. -On refait le processus jusqu'à aboutir à une solution satisfaisante.

4.9.1 Exemple de scénario 2

Supposons que notre client souhaite avoir deux fonctionnalités comme dans le scénario 1 dans notre application mais que cette fois ci, il voudrait introduire les valeurs de QoS du service total qu'il va recevoir, voici ses valeurs :

Disponibilité	Temps d'execution	Réputation	Prix	Localisation
0.7, avec un poids de 0.95	40 s, avec un poids de 0.85	4.2, avec un poids de 0.84	80 euros, avec un poids de 0.81	Algerie, avec un poids de 1

TABLE 4.7 – Exemple de fonctionnalité pour le scénario 2

Tout d'abord, nous devons normaliser les valeurs de QoS et la distance géographique, en utilisant l'équation **(5)** pour les critères positifs et l'équations **(6)** pour les critères négatifs, on obtient alors une requête qui ressemble à ceci :

$$RequteClient = [0.499, -0.002, 0.68, 0.22]$$

Ensuite, nous passons en vue la totalité services et on les normalise en utilisant les mêmes équations **(5)** et **(6)** ou nous obtenons des valeurs comparables à celle de la requête du client. Nous obtenons alors deux services qui répondent à ses besoins, en utilisant la fonction fitness de l'équation **(7)** ou nous allons soustraire la fitness de l'utilisateur et celle des services trouvés, on trouve alors une valeur :

$$FitnessServiceRequte = 35.61$$

Les valeurs de QoS du service composé après aggrégation sont :

Disponibilité	Temps d'execution	Réputation	Prix	Distance
0.74	36.17 s	3.7	41.46 euros	0 km

TABLE 4.8 – Exemple de fonctionnalité pour le scénario 2 après aggrégation

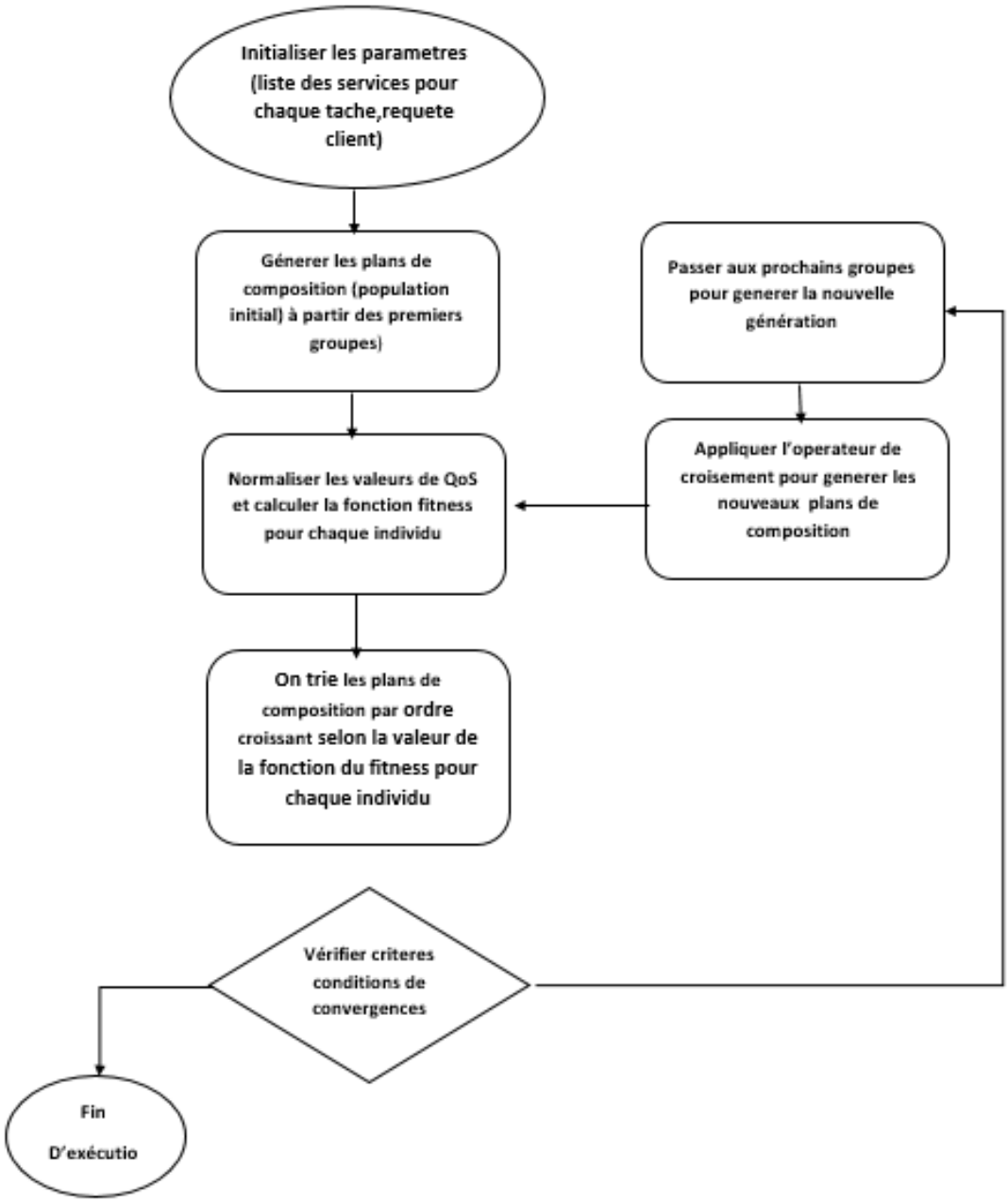

FIGURE 4.10 – Architecture globale de la sélection globale

4.10 Conclusion

Dans ce chapitre nous avons présenté la conception de notre solution en utilisant une sélection locale et globale pour la composition de service cloud ,en répondant aux exigences du client en terme de valeurs de qualité de service QoS et de localisation géographique.
Nous avons commencé par définir notre démarche de travail, par la suite nous avons fait une analyse des besoins pour identifier les acteurs du système et leurs actions respectives.

Nous avons présenté les différents modules et étapes de notre solution, et nous avons fait l'expérimentation et simulation avec des exemples dans le but de conclure les résultats qui confirme la validité de notre solution.
Dans le prochain chapitre, nous allons présenter notre solution avec les différentes expérimentations et la simulation pour tester tous les modules de notre système, et nous allons présenter et discuter cette solution, et la comparer avec quelques travaux connexes.

Chapitre 5

Chapitre V : Expérimentation

5.1 Introduction

Après la modélisation de notre solution, nous arrivons dans ce dernier chapitre ou nous allons parler de la mise en oeuvre jusqu'au développement de la solution proposée, la comparaison des resultats avec d'autres solutions existantes, une analyse et pour finir une discussion

5.2 Outils et envrionnement de développement

5.2.1 Outils

Pour aboutir au résultat voulu, nous avons utilisé un ordinateur avec un système d'exploitation Windows 10, alimenté par un processeur Intel Core i5, et une RAM de 8 GO, aussi nous avons utilisé des outils de développement qui nous ont semblé être un bon choix vu ce qu'ils nous apportent.

• Ordinateur portable HP 6550b.
• Intel(R) Core(TM) i5-8265U CPU @ 1.60GHz 1.80 GHz.
• 8,00 Go (7,87 Go utilisable).
• Système d'exploitation 64 bits, processeur x64 .
• Windows 10 Professionnel.

5.3 Environnement de développement

5.3.1 Netbeans

NetBeans IDE est un environnement de développement intégré gratuit et à code source ouvert destiné au développement d'applications sous Windows, Mac, Linux et Solaris [1]
Ce logiciel vous permet de mettre à jour vos applications pour utiliser le nouveau langage Java 8. Avec ses analyseurs de lots et ses convertisseurs sophistiqués, il effectue simultanément des recherches dans plusieurs applications afin de trouver des modèles correspondants à convertir en nouveau langage Java 8. [2]

5.3.2 JDK et JRE

Java SE JDK est le kit de développement JAVA officiel édité par Oracle. Java est un langage de programmation orienté objet très populaire, qui permet la réalisation d'applications performantes. Les programmes JAVA peuvent être exécutés sur n'importe quel système d'exploitation équipé d'une machine virtuelle JAVA. Pour y parvenir, le langage est compilé sous forme de bytecode qui est ensuite interprété par la machine virtuelle Java [3]
Java Runtime Environment 64-bit est une famille de logiciels gratuits qui servent à faire fonctionner tous les programmes écrits en Java, un langage de programmation. Cette version est destinée aux utilisateurs de Windows 64-bit. Des logiciels utilitaires au pur divertissement, JRE permet d'accéder à un contenu très divers pour vivre une expérience [40]

5.3.3 Protégé

Protégé est un système auteur pour la création d'ontologies. Il a été créé à l'université Stanford et est très populaire dans le domaine du Web sémantique et au niveau de la recherche en informatique.
Protégé est développé en Java. Il est gratuit et son code source est publié sous une licence libre (la Mozilla Public License).
Protégé peut lire et sauvegarder des ontologies dans la plupart des formats d'ontologies : RDF, RDFS, OWL, etc. [4]

5.3.4 SQL

SQL [5] (Structured Query Language) est un langage de requête à usage général et standardisé pour accéder et manipuler des bases de données au sein des SGBD et plus particulièrement

1. https ://www.oracle.com/dz/tools/technologies/netbeans-ide.html
2. https ://www.lesnumeriques.com/telecharger/netbeans-ide-20074
3. https ://www.lesnumeriques.com/telecharger/java-se-development-kit-jdk-20544
4. https ://fr.wikipedia.org/wiki/Protégé(logiciel)
5. https ://docs.oracle.com/en/database/oracle/sql-developer/index.html

des SGBDR. Les instructions SQL sont des instructions destinées à une base de données qui permettent aux utilisateurs d'effectuer diverses tâches. Les instructions SQL peuvent être utilisées directement par le système de gestion de base de données, mais elles peuvent également être intégrés dans du code écrit dans des langages tels que Java.

5.3.5 XAMPP

XAMPP est un projet open source à but non lucratif développé par Apache Friends. Son nom est un acronyme pour Cross-Platform (X), Apache, MySQL, PHP et Perl.
Il regroupe donc tout naturellement les outils libres suivants :
- **Apache :** un serveur Web HTTP multiplateforme ;
- **MariaDB :** un serveur de gestion de bases de données relationnelles MySQL pouvant être manipulées avec phpMyAdmin ;
- **PHP :** un langage de programmation back-end utilisé pour créer des sites et applications dynamiques ;
- **Perl :** un langage de programmation générique adapté au traitement et à la manipulation de fichiers texte (comme HTML ou XML).[41]

5.3.6 JDBC

JDBC est l'acronyme de Java DataBase Connectivity et désigne une API pour permettre un accès aux bases de données avec Java.
La connexion à une base de données requiert au préalable le chargement du pilote JDBC qui sera utilisé pour communiquer avec la base de données. Une fabrique permet alors de créer une instance de type Connection qui va encapsuler la connection à la base de données. [42]

5.4 Présentation des interfaces de l'application

Dans cette section nous allons présenter les interfaces de notre système,en expliquant le rôle de chaqu'une d'entre elle en details.

L'interface d'acceuil :

Quand on ouvre l'application la premiere interface qui apparait est l'interface d'acceuil qui va permettre aux fournisseurs et aux clients de s'inscrire ou de s'authentifier s'ils sont déjà inscrits. Ainsi qu'un bouton pour l'admin afin de lui permettre de faire la maintenance du système.

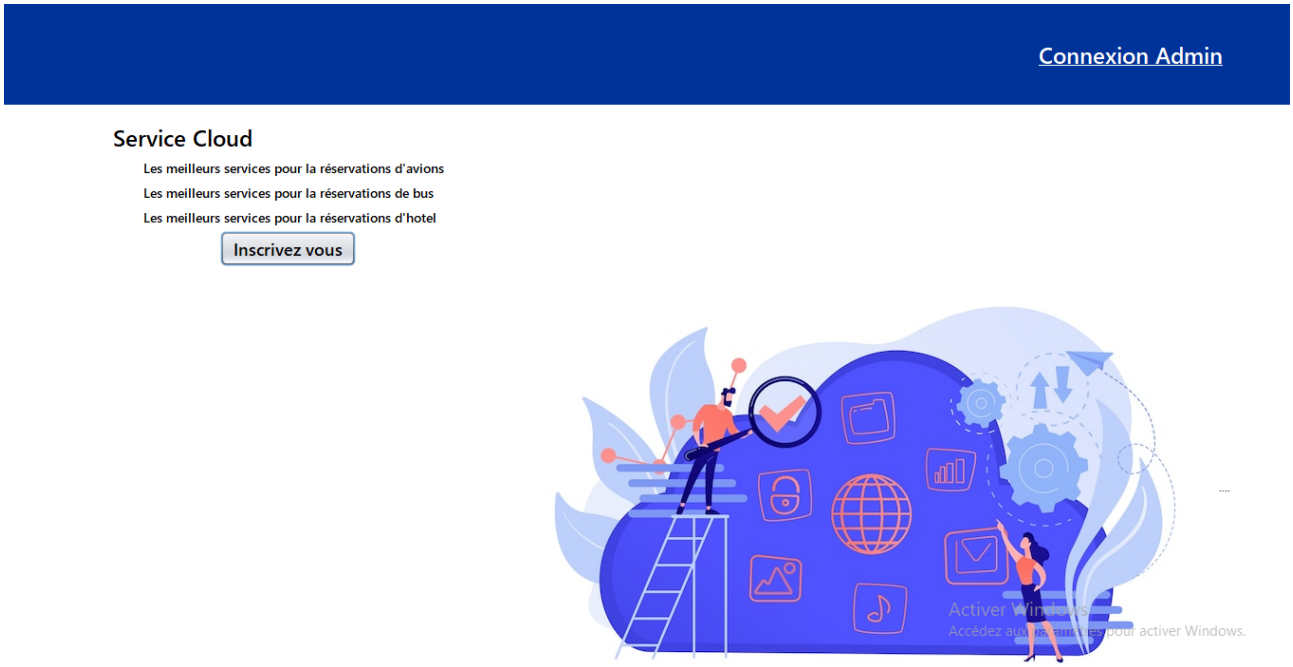

FIGURE 5.1 – Interface d'inscription de l'application

- Interface de connexion de l'admin :

Cette interface va permettre à l'admin de s'authentifier en introduisant ses informations (Adresse mail, Mot de passe) et cliquant sur le bouton Login,pour lui permettre d'effectuer ses opérations.

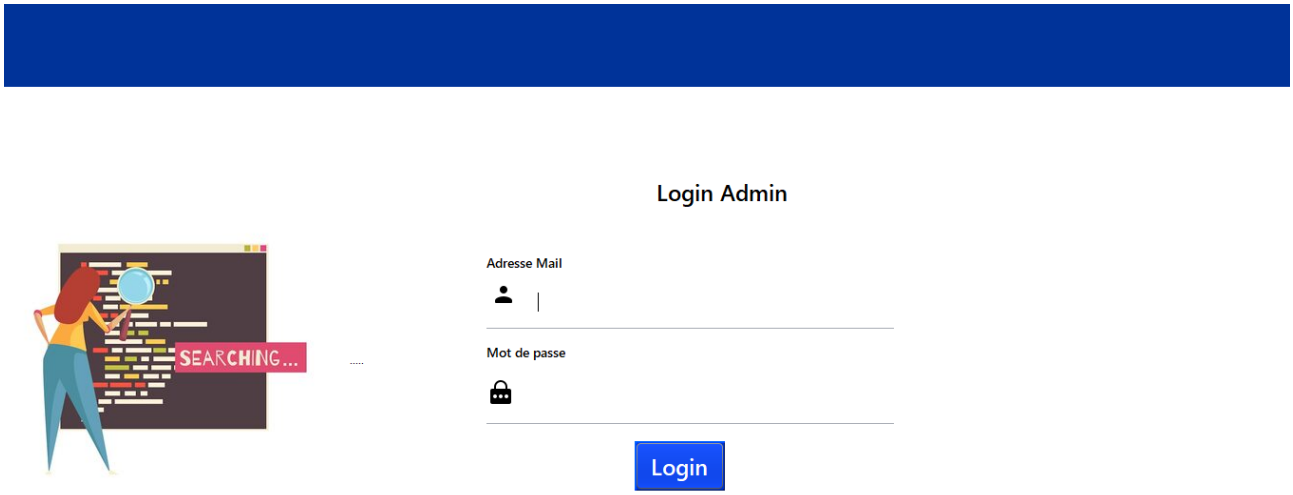

FIGURE 5.2 – Interface de login de l'admin

- Interface d'accueil de l'admin :

Apres s'être authentifier l'admin va se retrouver dans la page d'acceuil,ou il pourra mettre à jour la table Client (supprmier,modifier,ajouter Client),mettre à jour table fournisseur (supprmier,modifier,ajouter fournisseur), mettre à jour table service (supprmier,modifier,ajouter service).

- Interface d'inscription :

Il existe deux interface d'inscription, une dédié au client ou il devra introduire ses coordonnés (Nom,Adresse mail,Mot de passe,Numéro de téléphone) et une autre interface d'inscription dédié au fournisseur ou il devra introduire ses coordonnés (Nom,Adresse mail,Mot de passse,Numéro de téléphone ,Cloud auquel il appartient). Une fois les données introduite et une fois la vérification de chaque champs , l'utilisateur devra appuyer sur le bouton Sign up.

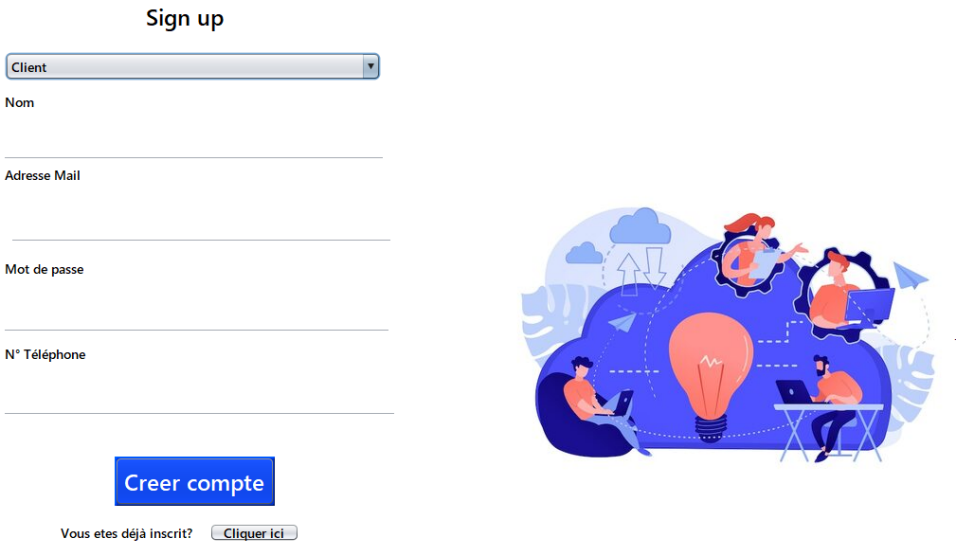

FIGURE 5.3 – Signup client

FIGURE 5.4 – Sign up fournisseur

- Interface de connexion :

Si le client ou le fournisseur sont déjà inscrit ils devront s'authentifier en introduisant leurs email et mot de passe pour pouvoir accéder à leurs interfaces dédiées respectivement.

Espace fournisseur :

Après s'etre authentifier le founisseur va se retrouver dans son espace fournisseur ou il pourra Ajouter des services,Modifier ses services,Supprimer ses services,Visualiser ses services déjà publié en appuyant sur le bouton remplir tableau et il pourra aussi afficher la description en format XML du service de son choix

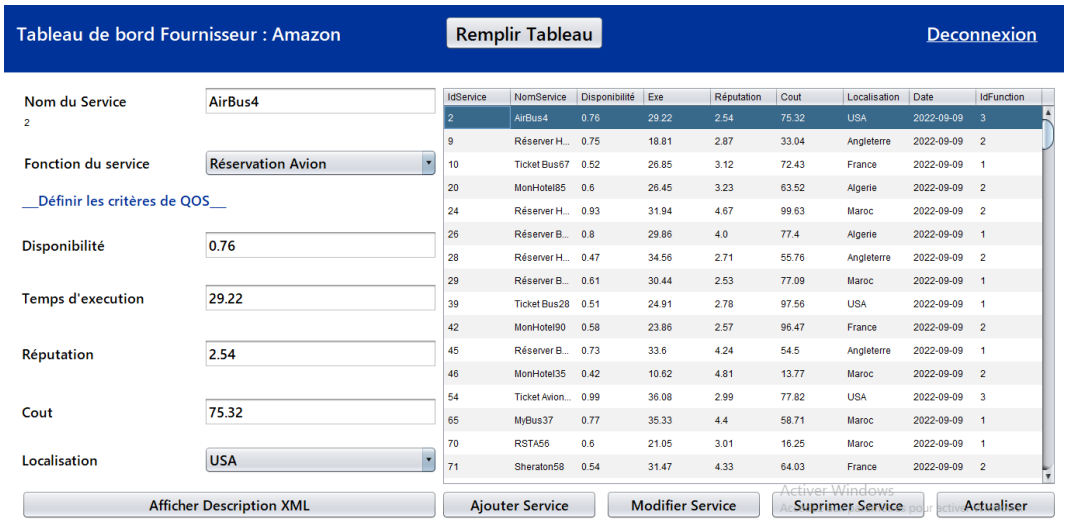

FIGURE 5.5 – Espace fournisseur

En cliquant sur le bouton Afficher Description XML ,la description du service séléctionné sera affiché comme-suit :

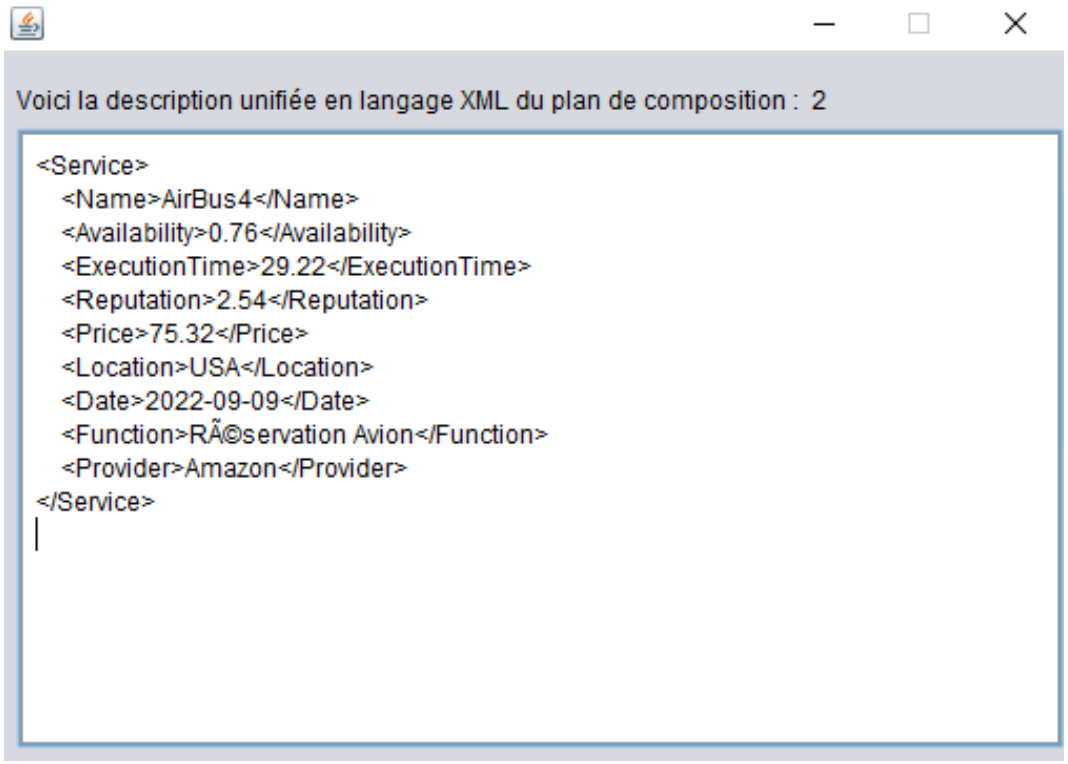

FIGURE 5.6 – Données XML du service

- Espace client :

Après s'etre authentifier le client va se retrouver dans son espace client ou il aura le choix entre introduire ses exigences pour les services un par un (sélection locale) ou introduire ses exigences pour les services une seule fois (selection globale).

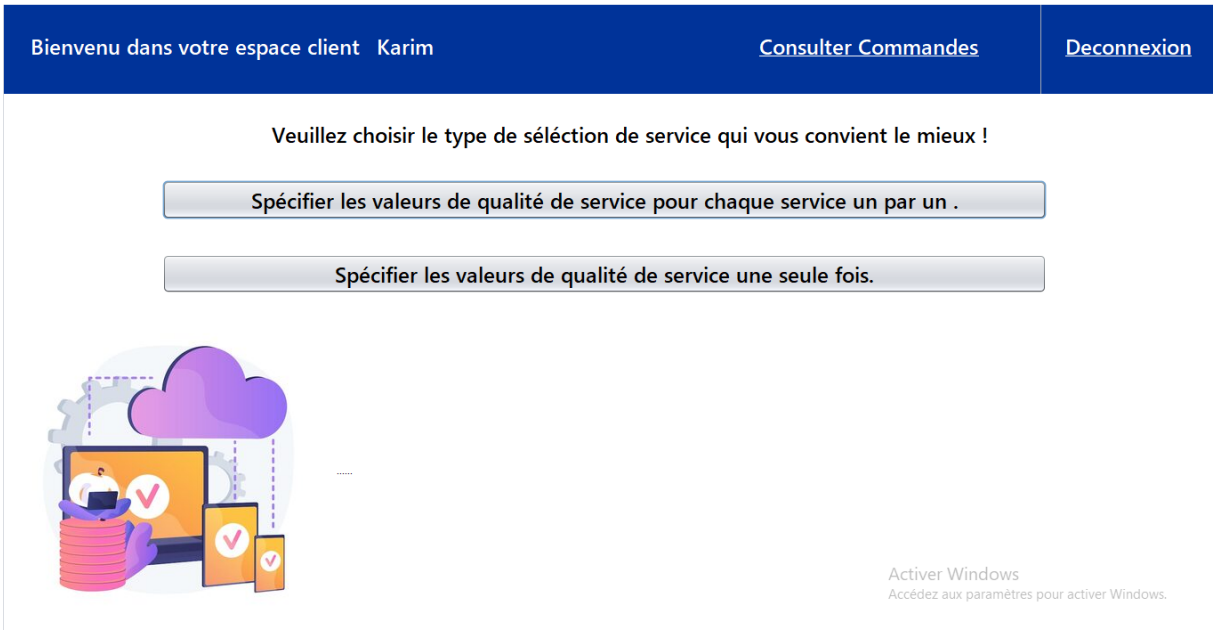

FIGURE 5.7 – Espace client

Si le client choisit de specifier ses exigences une seule fois alors il va se retouver dans l'interface suivante :

FIGURE 5.8 – Espace client

Ou il va introduire ses valeurs de qualité de qualité de service avec leurs poids ainsi que sa localisation géographique avec son poids , puis il devra cliquer sur calculer pour que l'algorithme de la sélection globale avec l'algorithme génétique lui génère le meilleur plan de composition , une fois qu'il aura cliquer sur OK il pourra cliquer sur commander ou une interface s'ouvrira contenant toutes les informations sur le services composé comme suit :

FIGURE 5.9 – Espace commande client

Une fois sur cette interface le client a le choix de signer ou pas le contrat client SLA en cliquant sur le bouton Contrat Client comme suit :

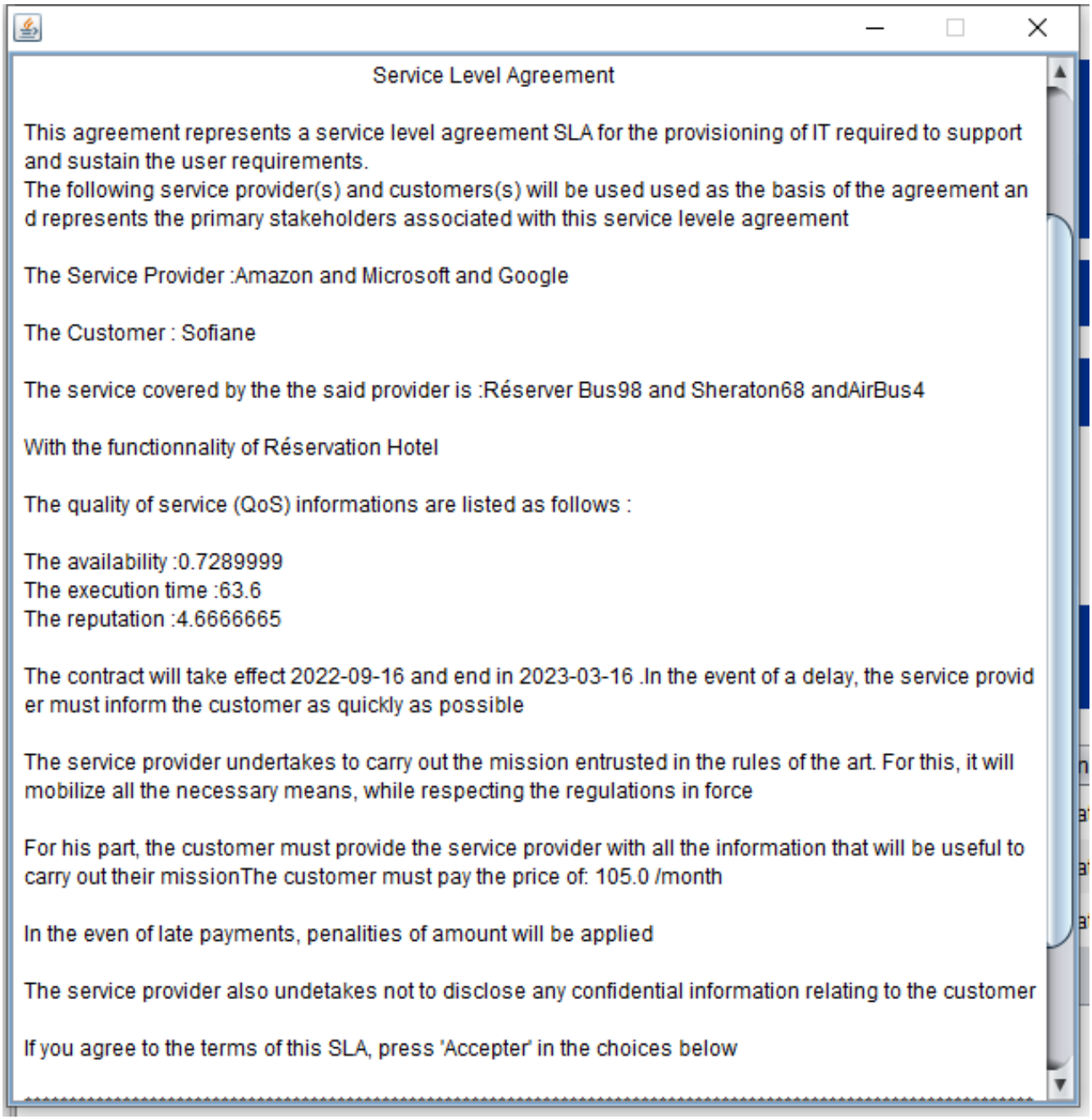

FIGURE 5.10 – Contrat du client SLA

Quand il aura signé le contrat il pourra appuyer sur le bouton commander et donc il se retrouvera dans son espace de consultation des commandes ,ou il pourra est consulter ses commande et noter les services qu'il a consommé :

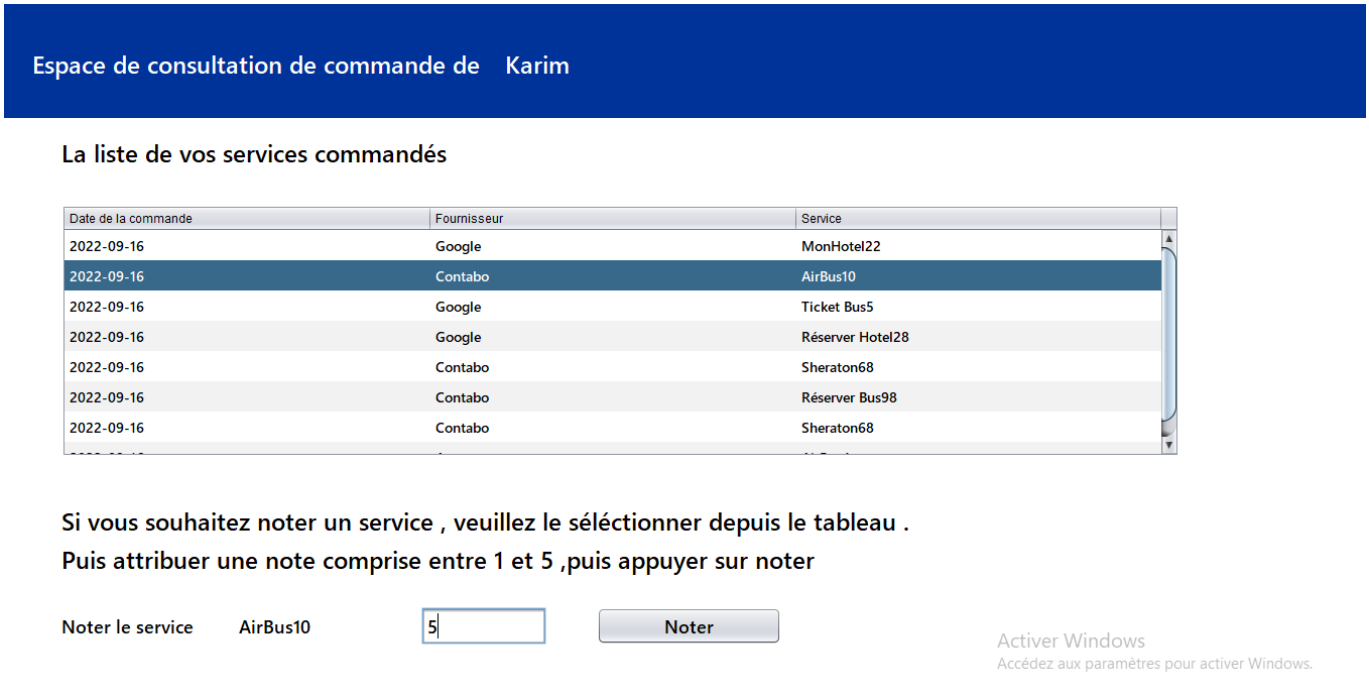

FIGURE 5.11 – Espace de consultation de la commande

5.5 Matching entre l'ontologie de description unifié et la base de données relationnelle

Dans notre solution nous avons créer une ontologie avec l'outils protégé(bas de page)

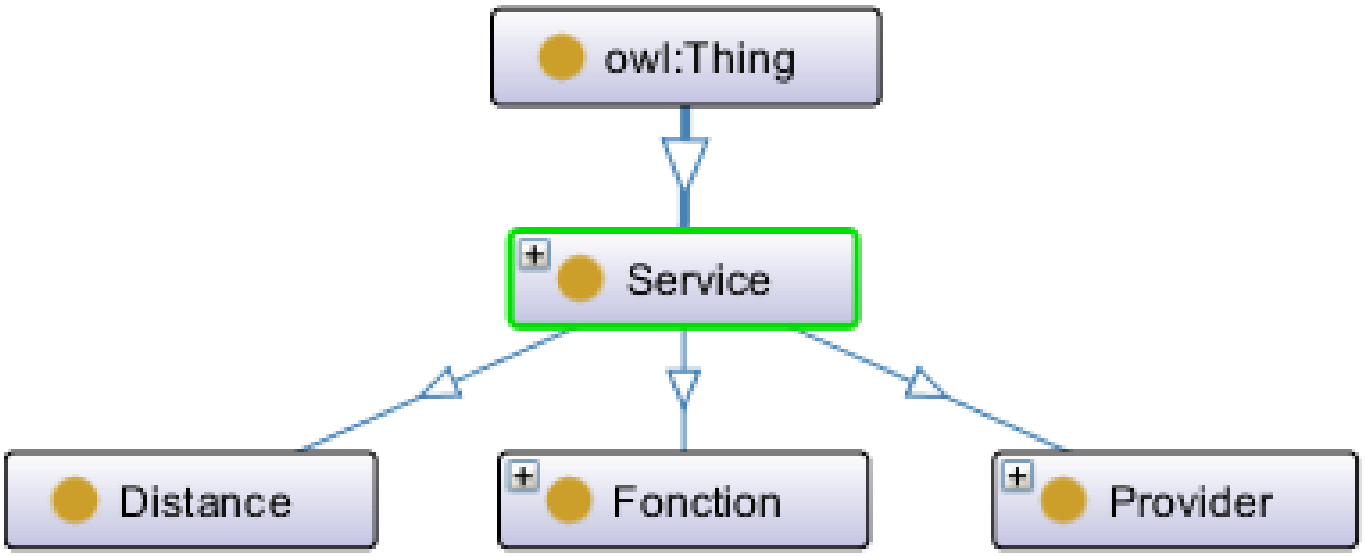

FIGURE 5.12 – L'ontologie utilisée

Pour unifier la description de service publié par les fournisseur , pour se faire nous avons utilisé obda.

5.5.1 OBDA

Le paradigme Ontology-Based Data Access (OBDA) consiste à exposer, à des fins d'interrogation, une vue conceptuelle du domaine d'intérêt, donnée sous la forme d'une ontologie qui masque la structure des sources de données. Des requêtes peuvent alors être posées sur cette vue conceptuelle de haut niveau, et les utilisateurs finaux n'ont plus besoin de comprendre les sources de données, la relation entre elles ou l'encodage des données. Les requêtes des utilisateurs sont traduites par le système OBDA en requêtes sur une ou plusieurs sources de données. L'ontologie est connectée aux sources de données via des mappages, une spécification déclarative qui relie les symboles de l'ontologie (classes et propriétés) aux vues SQL sur les données. La norme W3C R2RML a été créée dans le but de fournir un langage pour la spécification des mappages dans le cadre OBDA. L'ontologie ainsi que les mappages exposent un graphe RDF virtuel, qui peut être interrogé à l'aide de SPARQL, le langage de requête standard de la communauté du Web sémantique.[43].
Avant de faire le mapping entre l'ontologie et la base de donnés nous devons d'abord établir la connexion comme suit :

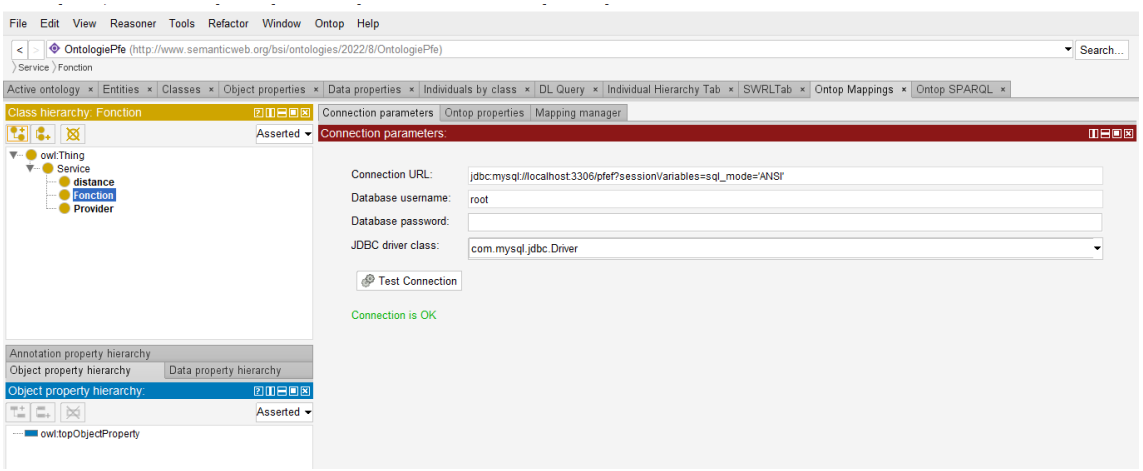

FIGURE 5.13 – Etablissement de la connexion entre l'ontologie et la base de données

Ensuite nous devons spécifier les classes de l'ontologie ainsi que les propriétés (data properties) comme suit :

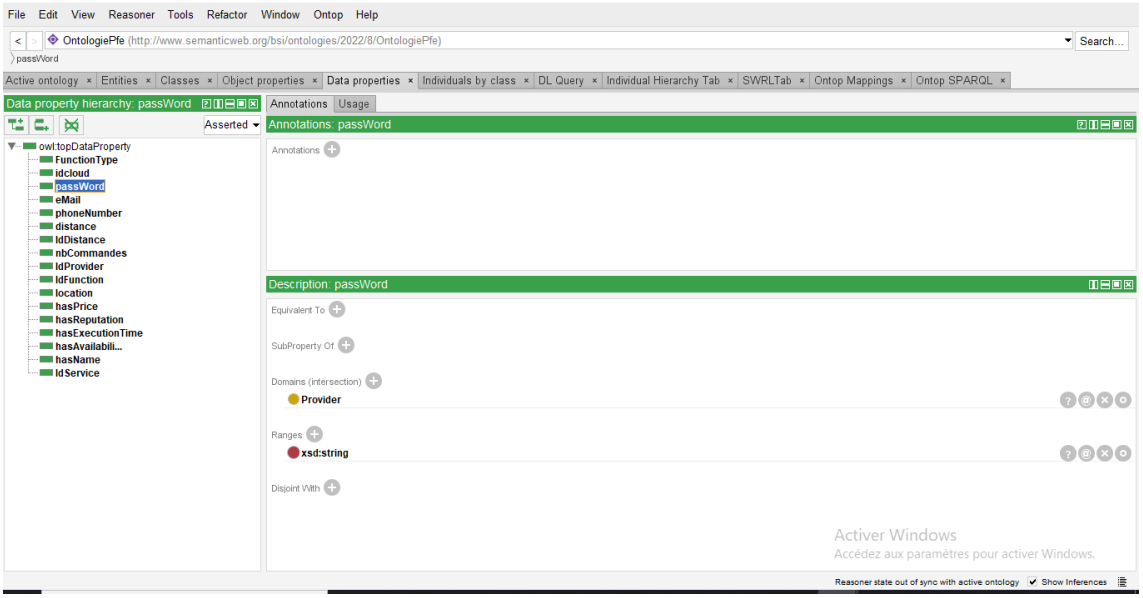

FIGURE 5.14 – Spécification des classes et des propriétés

La figure suivante représente les mapping créer pour pouvoir récupérer les instances disponibles dans la base de données vers l'ontologie :

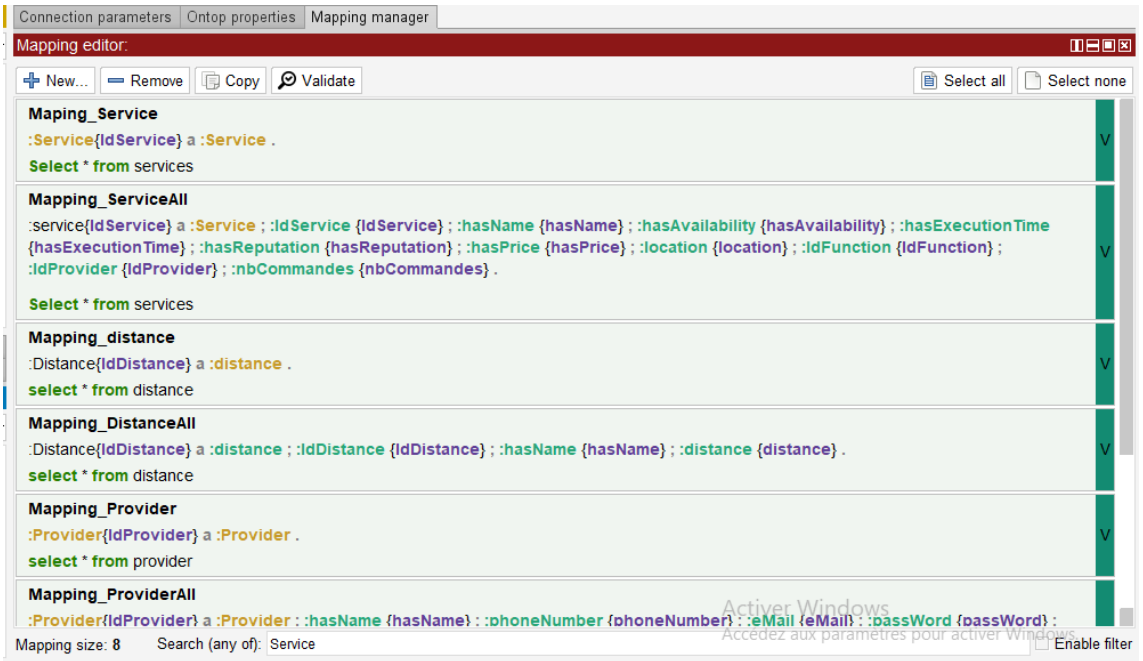

FIGURE 5.15 – Représentation des règles de matching

Une fois la mapping effectué nous devons actionner un bouton pour récupérer les valeurs présentes dans la base de données :

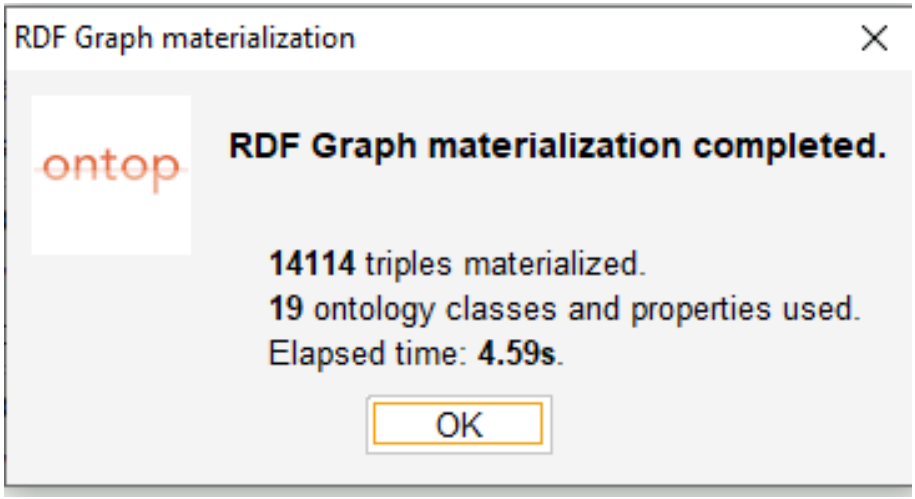

FIGURE 5.16 – Mapping effectué

Par la suite nous pouvons vérifier que le matching et mapping a bien été effectué comme le montre la figure ci-dessous :

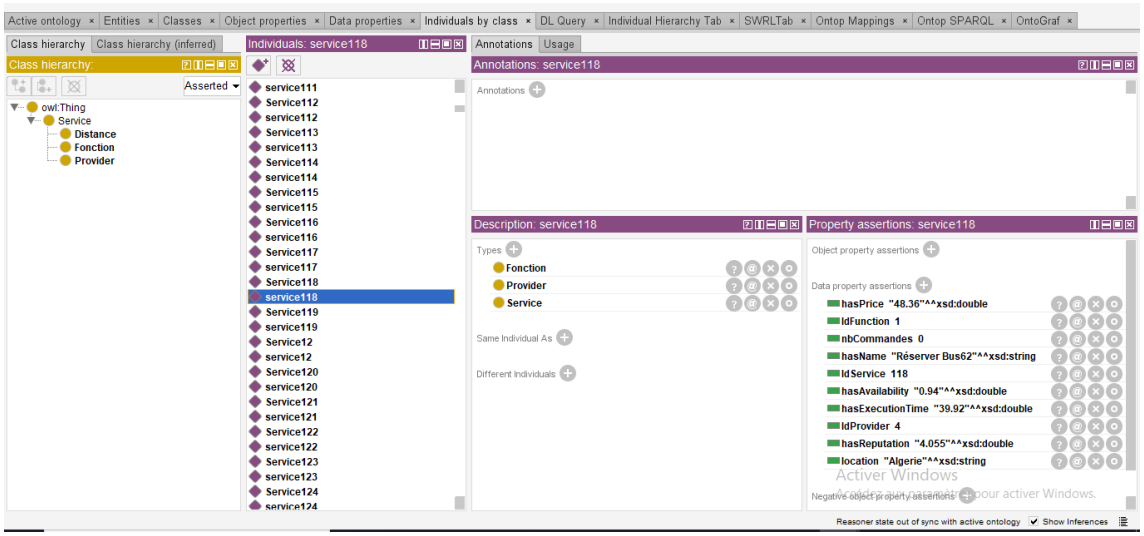

FIGURE 5.17 – Instances importées

Les figures ci-dessous montrent le temps d'execution de notre solution en comparaison avec Ghezouani et Merizig.

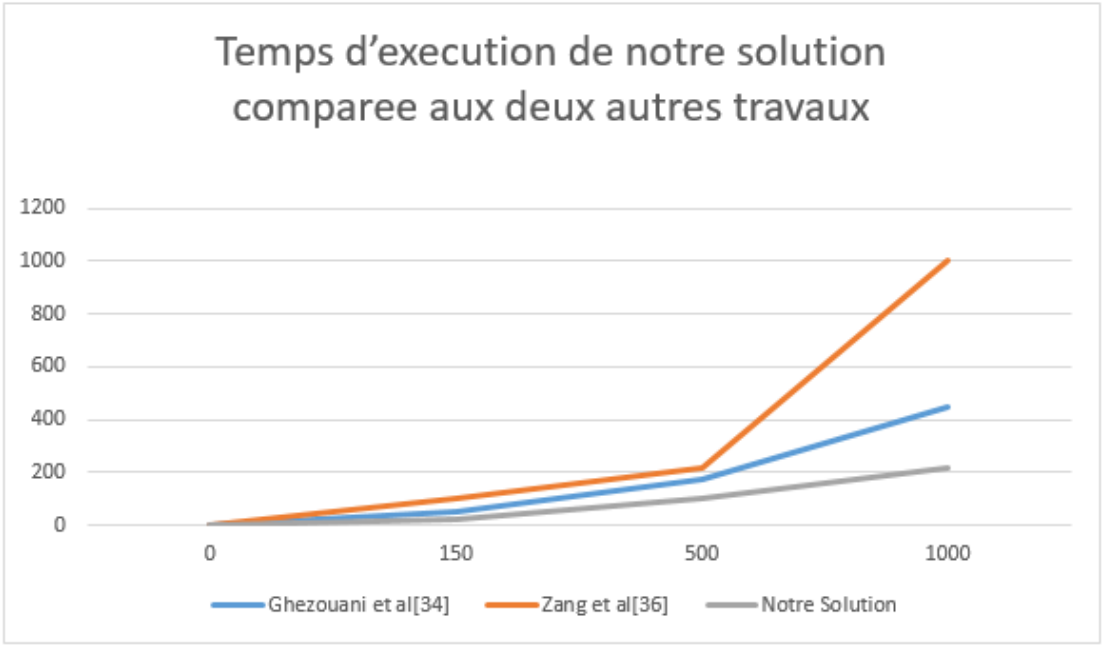

FIGURE 5.18 – Courbe du temps d'execution de notre solution comparée aux deux autres (Ghezouani et Merizig en fonction du nombre de service)

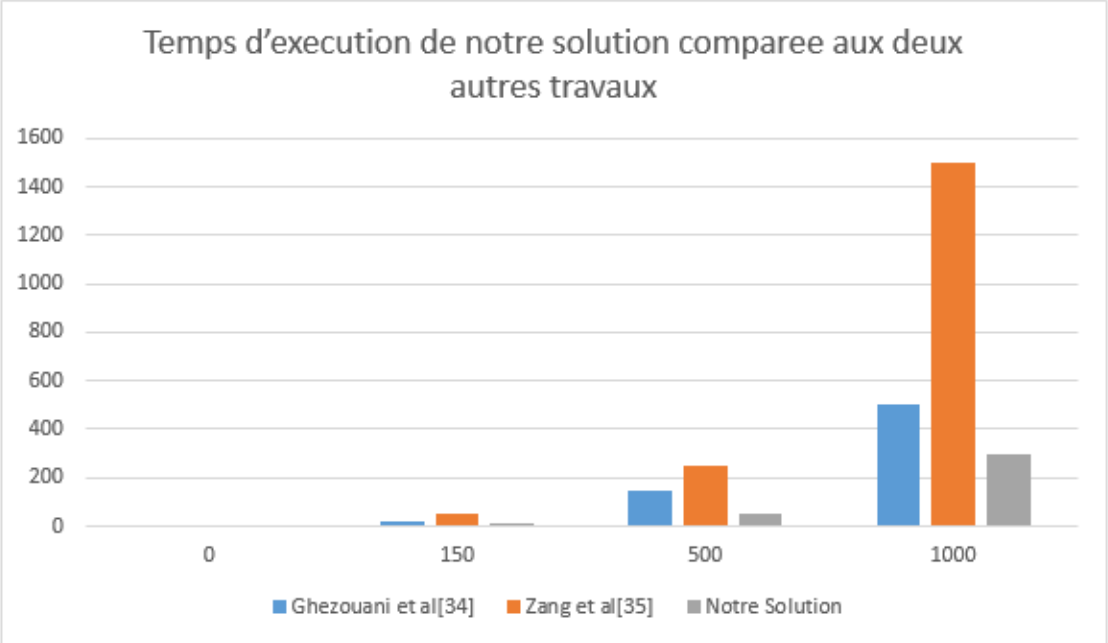

FIGURE 5.19 – Histogramme du temps d'execution de notre solution comparée aux deux autres (Ghezouani et Merizig en fonction du nombre de service)

On peut d'aperçevoir que l'algorithme de la séléction locale / globale que nous avons implémenté apporte des résultats convainquants.

Après les expérimentations et les tests, nous pouvons conclure que notre solution est capable de donner des meilleurs résultats qui satisfait les demandes des clients, les expérimentations montre les meilleures performances pour cette solution.
À la fin nous allons récapituler ces résultats concernant notre solution dans le tableau de l'étude comparative présenté dans le deuxième chapitre.
C1 : Approche proposé dans un environnement
mono-cloud.
C2 : Approche proposé dans un environnement multi-cloud.
C3 : Solution qui prend en compte les valeurs de qualité de service QOS.
C4 : Solution à base de sélection locale.
C5 : Solution à base de sélection globale.
C6 : Solution qui prend en compte l'aspect commercial (SLA).
C7 : Solution qui prend en compte l'aspect sémantique (ontologie).
C8 : Solution méta-heuristique.

Approches	C1	C2	C3	C4	C5	C6	C7	C8
Notre solution	✓	✓	✓	✓	✓	✓	✓	✓

TABLE 5.1 – Récapitulatif de l'approche proposée

5.6 Conclusion

Tout au long de ce dernier chapitre, nous avons démontré l'efficacité de l'approche proposée en montrant les différents outils de développement et les résultats obtenus. En premier lieu, nous avons introduit quelques définitions des outils utilisés.
Ensuite, nous avons illustré les interfaces principales de l'application par des captures écran ainsi que le processus de matching avec l'ontologie. Nous avons terminé l'implémentation de notre application en respectant la conception élaborée. Enfin, une comparaison de l'approche proposée avec les approches décrites dans l'état de l'art est présentée.

Conclusion générale

Le Cloud Computing offre une infinité de services avec une infinité de fonctionnalités avec des valeurs de qualités de service diverses. Il est très rare de trouver deux services ayant la même fonctionnalité et mêmes valeurs de qualités de services QoS. D'autre part avec la croissance et l'évolution du Cloud, les requêtes des clients deviennent de plus en plus complexes et nécessitent pas un pas deux mais plusieurs services.

Etant donné que les requetes des clients sont plus en plus complexes et qu'il n'existe pas un seul service pour adéquat, on se trouve dans l'obligation d'avoir une combinaison des services pour les satisfaire. Donc pour ce faire nous avons besoin d'une composition de service, c'est là où réside toute l'importance de la composition face à ce problème. Il peut s'avérer très compliqué étant donné qu'il faut prendre en compte les valeurs de qualité de service QoS..

Dans ce contexte, nous avons proposé une solution pour ce problème dans le Cloud à base d'une sélection locale et une sélection globale, qui prend en compte les valeurs de qualités de service QoS et la localisation géographique. Avant d'entamer la réalisation de cette solution, nous avons défini des généralités sur le contexte général, qui est le Cloud Computing, et nous avons étudié le problème de la sélection et composition dans le Cloud Computing, avec la comparaison des différents travaux connexes dans ce contexte. Ça nous a permis de modéliser et définir notre solution, puis de présenter et discuter notre solution après une série des tests et expérimentations.

Notre solution présente deux scénarios : Le premier scenario est basé sur la sélection globale à base d'algorithme génétique ,le deuxième scenario utilise la sélection locale .Nous avons utilisé Java pour tester et simuler notre solution avec deux autres solutions pour pouvoir la valider . Les expérimentations montrent que notre solution présente une performance meilleure que les deux travaux avec lesquels nous l'avons comparé.

Ce travail ouvre des perspectives pour la satisfaction des requêtes complexes des utilisateurs qui ne peuvent pas être réalisées grâce à un seul service. Ajouter plusieurs domaines et fonctionnalités à notre ontologie, introduire d'autres valeurs de qualité de service dans le processus de sélection de services dans le but d'améliorer les résultats.

Bibliographie

[1] Amal Maryoosh. Improving data storage security in cloud computing using elliptic curve cryptography. *IOSR Journal of Computer Engineering (IOSR-JCE)*, 17 :48–53, 07 2015.

[2] BAMBA Khadidjatou Iman. Les fondamentaux du cloud computing, September 2014.

[3] George Pallis. Cloud Computing : The New Frontier of Internet Computing. *IEEE Internet Computing*, 14(5) :70–73, September 2010. ISSN 1089-7801. doi : 10.1109/MIC.2010.113. URL http://ieeexplore.ieee.org/document/5562494/.

[4] Fang Liu, Jin Tong, Jian Mao, Robert Bohn, John Messina, Lee Badger, Dawn Leaf, et al. Nist cloud computing reference architecture. *NIST special publication*, 500(2011) :1–28, 2011.

[5] Arron Fu. 7 différents types de structures de calcul en nuage que vous devriez savoir, March 2017. URL https://www.uniprint.net/fr/7-types-cloud-computing-structures/.

[6] Amin Jula, Elankovan Sundararajan, and Zalinda Othman. Cloud computing service composition : A systematic literature review. *Expert Systems with Applications*, 41(8) : 3809–3824, June 2014. ISSN 09574174. doi : 10.1016/j.eswa.2013.12.017.

[7] Ken Rubenstein. Cloud computing in life sciences r&d. *Cambridge : Healthtech Institute*, 2010.

[8] B. Kezia Rani, B. Padmaja Rani, and A. Vinaya Babu. Cloud Computing and Inter-Clouds – Types, Topologies and Research Issues. *Procedia Computer Science*, 50 :24–29, 2015. ISSN 18770509. doi : 10.1016/j.procs.2015.04.006. URL https://linkinghub.elsevier.com/retrieve/pii/S1877050915005074.

[9] +Bastien L. XaaS : définition, avantages et exemples du Tout en tant que Service, September 2017. URL https://www.lebigdata.fr/xaas-definition-avantages-exemples. Publication Title : LeBigData.fr.

[10] Peter Mell, Tim Grance, et al. The nist definition of cloud computing. 2011.

[11] Vic (J.R.) Winkler. Introduction à l'informatique en nuage et à la sécurité. *Pearson Education*, page 8, 2011.

[12] Hwaiyu Geng. *Data Center Handbook*. John Wiley & Sons, 2014.

[13] Qi Zhang, Lu Cheng, and Raouf Boutaba. Cloud computing : state-of-the-art and research challenges. *Journal of Internet Services and Applications*, 1(1) :7–18, May 2010. ISSN 1867-4828, 1869-0238. doi : 10.1007/s13174-010-0007-6.

[14] Tim Abels, Puneet Dhawan, and Balasubramanian Chandrasekaran. An overview of xen virtualization. *Dell Power Solutions*, 8 :109–111, 2005.

[15] Rajkumar Buyya, James Broberg, and Andrzej M. Goscinski. *Cloud Computing : Principles and Paradigms*. John Wiley & Sons, December 2010. ISBN 978-1-118-00220-9.

[16] G Cordall. Service level agreements. *City University London*, pages 1–5, 2014.

[17] Mohammad Aazam and Eui Nam Huh. Inter-cloud Media Storage and Media Cloud Architecture for Inter-cloud Communication. In *2014 IEEE 7th International Conference on Cloud Computing*, pages 982–985, Anchorage, AK, USA, June 2014. IEEE. ISBN 978-1-4799-5063-8 978-1-4799-5062-1. doi : 10.1109/CLOUD.2014.151.

[18] Yashpalsinh Jadeja and Kirit Modi. Cloud computing - concepts, architecture and challenges. In *2012 International Conference on Computing, Electronics and Electrical Technologies (ICCEET)*, pages 877–880, Nagercoil, Tamil Nadu, India, March 2012. IEEE. ISBN 978-1-4673-0212-8 978-1-4673-0211-1 978-1-4673-0210-4. doi : 10.1109/ICCEET.2012.6203873. URL `http://ieeexplore.ieee.org/document/6203873/`.

[19] Vincent Kherbache, Mohamed Moussalih, Yannick Kuhn, and Allan Lefort. Administration de systèmes, réseaux et applications à base de logiciels libres, 2009.

[20] Hajar Omrana. *Vers une Composition Dynamique des Services Web : une approche de Composabilité Offline*. Theses, UNIVERSITÉ MOHAMMED V AGDAL – ECOLE MOHAMMADIA D'INGENIEURS, January 2014. URL `https://tel.archives-ouvertes.fr/tel-01134037`.

[21] Mr BARKAT Abdelbasset. Composition de service web dans le cloud computing. Masters Thesis, Université Mohamed KHIDER - BISKRA, 2018.

[22] Amin Jula, Elankovan Sundararajan, and Zalinda Othman. Cloud computing service composition : A systematic literature review. *Expert Systems with Applications*, 41(8) : 3809–3824, June 2014. ISSN 09574174. doi : 10.1016/j.eswa.2013.12.017. URL `https://linkinghub.elsevier.com/retrieve/pii/S0957417413009925`.

[23] Qiping She, Xiaochao Wei, Guihua Nie, and Donglin Chen. QoS-aware cloud service composition : A systematic mapping study from the perspective of computational intelligence. *Expert Systems with Applications*, 138 :112804, December 2019. ISSN 09574174. doi : 10.1016/j.eswa.2019.07.021. URL https://linkinghub.elsevier.com/retrieve/pii/S0957417419305007.

[24] Minghai Yuan, Zhuo Zhou, Xianxian Cai, Chao Sun, and Wenbin Gu. Service composition model and method in cloud manufacturing. *Robotics and Computer-Integrated Manufacturing*, 61 :101840, February 2020. ISSN 07365845. doi : 10.1016/j.rcim.2019.101840. URL https://linkinghub.elsevier.com/retrieve/pii/S0736584519301024.

[25] Ali Younes, Mohamed Essaaidi, and Ahmed El Moussaoui. SFL Algorithm for QoS-based Cloud Service Composition. *International Journal of Computer Applications*, 97(17) :42–49, July 2014. ISSN 09758887. doi : 10.5120/17103-7700. URL http://research.ijcaonline.org/volume97/number17/pxc3897700.pdf.

[26] Ying Huo, Yi Zhuang, Jingjing Gu, Siru Ni, and Yu Xue. Discrete gbest-guided artificial bee colony algorithm for cloud service composition. *Applied Intelligence*, 42(4) :661–678, June 2015. ISSN 0924-669X, 1573-7497. doi : 10.1007/s10489-014-0617-y. URL http://link.springer.com/10.1007/s10489-014-0617-y.

[27] Rashda Khanam, Rakesh Ranjan Kumar, and Binita Kumari. A novel approach for cloud service composition ensuring global QoS constraints optimization. In *2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pages 1695–1701, Bangalore, September 2018. IEEE. ISBN 978-1-5386-5314-2. doi : 10.1109/ICACCI.2018.8554787. URL https://ieeexplore.ieee.org/document/8554787/.

[28] Besoins fonctionnels & Besoins non fonctionnels - Savoir+, December 2019. URL https://savoir.plus/besoins-fonctionnels-non-fonctionnels/.

[29] Merizig Abdelhak. Approche de composition de services web dans le Cloud Computing basée sur la coopération des agents. Master's thesis, Mohamed Khider – BISKRA, 2017.

[30] Karim Aoun Seghir and Abdelkader Mehraz. Composition de services de Cloud de type SaaS à base de sélection globale. Master's thesis, November 2020.

[31] Ben Hamed Ali. Optimisation multiobjectif et problèmes d'optimisation mono-objectifs. February 2014. Section : Informatique et Télécommunications.

[32] Algorithmes génétiques, September 2022. URL http://igm.univ-mlv.fr/~dr/XPOSE2013/tleroux_genetic_algorithm/fonctionnement.html.

[33] Francisco José MOO Mena. Modélisation des architectures logicielles dynamiques : application à la gestion de la qualité de service des applications à base de services Web. page 219.

[34] Heba Kurdi, Abeer Al-Anazi, Carlene Campbell, and Auhood Al Faries. A combinatorial optimization algorithm for multiple cloud service composition. *Computers & Electrical Engineering*, 42 :107–113, February 2015. ISSN 00457906. doi : 10.1016/j.compeleceng.2014.11.002. URL https://linkinghub.elsevier.com/retrieve/pii/S0045790614002717.

[35] Souad Ghazouani, Haithem Mezni, and Yahya Slimani. Bringing semantics to multicloud service compositions. *Software : Practice and Experience*, 50(4) :447–469, April 2020. ISSN 0038-0644, 1097-024X. doi : 10.1002/spe.2789. URL https://onlinelibrary.wiley.com/doi/10.1002/spe.2789.

[36] Ali Bentaleb and Ahmed Ettalbi. Toward Cloud SaaS for web service composition optimization based on genetic algortihm. In *2016 2nd International Conference on Cloud Computing Technologies and Applications (CloudTech)*, pages 147–152, Marrakech, Morocco, May 2016. IEEE. ISBN 978-1-4673-8894-8. doi : 10.1109/CloudTech.2016.7847692. URL http://ieeexplore.ieee.org/document/7847692/.

[37] Miao Zhang, Li Liu, and Songtao Liu. Genetic Algorithm Based QoS-aware Service Composition in Multi-cloud. In *2015 IEEE Conference on Collaboration and Internet Computing (CIC)*, pages 113–118, Hangzhou, China, October 2015. IEEE. ISBN 978-1-5090-0089-0. doi : 10.1109/CIC.2015.23. URL http://ieeexplore.ieee.org/document/7423072/.

[38] Guobing Zou, Yang Xiang, Ruoyun Huang, and You Xu. Ai planning and combinatorial optimization for web service composition in cloud computing. pages 28–35, 04 2010. ISBN 9789810858643. doi : 10.5176/978-981-08-5837-7_166.

[39] Extreme Programming : A Gentle Introduction., sep 2022. URL http://www.extremeprogramming.org/.

[40] Télécharger Java Runtime Environment 64-bit 8.0.3010.9 pour Windows - Filehippo.com. URL https://filehippo.com/fr/download_jre-64/.

[41] Télécharger XAMPP (gratuit) - Clubic, sep 2022. URL https://www.clubic.com/telecharger-fiche70674-xampp.html.

[42] Développons en Java - JDBC (Java DataBase Connectivity), sep 2022. URL https://www.jmdoudoux.fr/java/dej/chap-jdbc.htm.

[43] Diego Calvanese, Benjamin Cogrel, Elem Güzel Kalaycı, Sarah Komla-Ebri, Roman Kontchakov, Davide Lanti, Martín Rezk, Mariano Muro, and Guohui Xiao. Obda with the ontop framework. 01 2015.