

République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique



Université A. Mira de Béjaia
Faculté des Sciences Exactes
Département d'Informatique

MÉMOIRE DE MASTER RECHERCHE

En Informatique

Option : Systèmes d'Information Avancés

Thème

Proposition d'une approche de composition de services IoT avec prise en compte des QoS et de l'énergie

Présenté par :

M. AISSAOUI Aghiles

M. BAGHDADI Chems Eddine

Devant les jurés composés de :

Président :	M.	ZERARGA Lotfi
Examineur :	Mme.	BATTAT Nadia
Encadrante :	Mme.	AIT HACENE Souhila

2023 - 2024

✧ Remerciements ✧

*Quoi que nous avançons et que nos portes s'ouvrent,
que nous atteignons tout ce que nous rêvons, il faut rappeler ceux qui ont été
la cause de notre succès,
ceux qui nous ont soutenus et ont tenu notre main Pour continuer, ceux dont
la générosité nous a encouragés et motivés :*

*Avant tout, nous remercions Dieu, le Tout-Puissant, pour sa guidance et ses
bénédictions tout au long de ce projet. C'est grâce à Sa miséricorde et à Sa
grâce que nous avons trouvé la force et la persévérance nécessaires pour mener
à bien ce travail.*

*Nous adressons nos sincères remerciements à notre encadrant, Mme AIT
HACENE SOUHILA, pour son accueil, ses conseils avisés et sa
disponibilité, ainsi que son soutien indéfectible qui étaient précieux afin de
mener notre travail sur la bonne voie.*

*Nous tenons à exprimer notre profonde gratitude à nos familles, notamment à
nos parents, pour leur amour, leur soutien inconditionnel et leur patience.
Leur sacrifice et leur foi inébranlable en nos capacités ont été essentiels à
notre réussite, nous permettant de poursuivre nos études et de mener à bien ce
projet.*

*Nous souhaitons également exprimer notre gratitude aux membres du jury
pour avoir accepté de présider le jury et d'examiner ce mémoire.*

*Enfin, nous souhaitons remercier toutes les personnes qui, de près ou de loin,
ont contribué à l'accomplissement de ce mémoire.*

Aghiles & Chems Eddine

※ Dédicaces ※

C'est avec fierté que je dédie ce modeste travail à

À celui dont je porte le nom avec fierté, à celui qui a consacré toute sa vie à être meilleur que lui, à celui qui m'a soutenu sans compensation et m'a donné sans limites, mon cher père « Aissaoui Rabah ».

À celle qui m'a soutenue et qui m'a inspirée de continuer mon chemin et qui a facilité mes difficultés par ses prières, à la lumière qui m'a éclairé le chemin, ma chère mère « Z. Ourdia ».

A mes agréables frères « Massinissa », « Fatah », « redouane », et ma chère sœur « thanina ».

À toute ma famille pour leur soutien.

Je tiens également à remercier mes chers amis : Smail, Hani, Nabile, Ahmed, Yacine, Slimane, Akram, Aissam, Ilyes, Syfhax, Mayas.

À mon binôme, « Chems Eddine ». Que je lui souhaite beaucoup de réussite dans sa vie.

À tous ceux qui ont participé à ma réussite.

M. AISSAOUI Aghiles

※ Dédicaces ※

C'est avec fierté que je dédie ce modeste travail à

À celui dont je porte le nom avec fierté, à celui qui a consacré toute sa vie à être meilleur que lui, à celui qui m'a soutenu sans compensation et m'a donné sans limites, mon cher père « Baghdadi Mohamed ».

À celle qui m'a soutenue et qui m'a inspirée de continuer mon chemin et qui a facilité mes difficultés par ses prières, à la lumière qui m'a éclairé le chemin, ma chère mère « A.Ghania ».

*A mon frère « Sid ali », et mes chère sœurs « Wiam »,
« oumaima », « sabah ».*

À toute ma famille pour leur soutien « A. Abedrezak ».

Je tiens également à remercier mes chers amis : Abderrahmane, Zakaria, Bilal, Sid ahmed, nassime, Anouare, Imad, Rahim, Abedrezak, Youcef, Areslane.

À mon binôme, « Aghiles ». Que je lui souhaite beaucoup de réussite dans sa vie.

À tous ceux qui ont participé à ma réussite.

M. BAGHDADI Chems Eddine

Table des matières

Table des matières	i
Liste des figures	iv
Liste des tableaux	vi
Liste des abréviations	vii
Introduction générale	1
1 Concepts généraux	3
1.1 Introduction	3
1.2 Internet des objets	3
1.2.1 L'historique de l'Internet des objets	3
1.2.2 Définition de l'IoT	4
1.2.3 Axes de l'IoT	5
1.3 Qu'est-ce qu'un objet ?	6
1.4 Capteurs et Actionneurs	6
1.4.1 Capteurs	6
1.4.2 Actionneurs	7
1.5 Caractéristiques fondamentales de l'IoT	7
1.6 L'architecture de l'IoT	8
1.7 Les domaines de l'Internet des Objets	10
1.8 Les Avantages et les défis de l'IoT	12
1.9 Middleware (ou intergiciel)	13
1.10 Architecture orienté service (SoA)	14
1.10.1 La notion de service	14
1.10.2 La notion de SoA	15
1.10.3 Acteurs de l'architecture SoA :	16
1.10.4 Caractéristiques de l'architecture SoA	16
1.11 Conclusion	17

2	État de l'art sur la composition de services	18
2.1	Introduction	18
2.2	Composition des services IoT.....	18
2.2.1	Définition de concepts liés à la problématique de composition de services	19
2.2.2	Composition concrète (Comp).....	20
2.2.3	Etapes de la composition des services.....	20
2.2.4	Qualité de service (QoS)	21
2.2.5	Vecteur QoS d'un service candidat « QoS (SC) »	23
2.2.6	Vecteur d'attributs QoS d'une composition	23
2.2.7	Fonction fitness (d'utilité).....	24
2.3	Machine Learning	25
2.3.1	Types de systèmes d'apprentissage automatique.....	25
2.3.1.1	L'apprentissage supervisé.....	25
2.3.1.2	L'apprentissage non supervisé.....	27
2.3.1.3	Apprentissage semi-supervisé	27
2.3.1.4	Apprentissage par renforcement	27
2.4	Les Métaheuristique	27
2.4.1	Définition	27
2.5	Etat de l'art sur les approches de composition de services IoT prenant en compte les QoS et l'énergie.....	30
2.5.1	Approches basées sur des technique déterministes.....	30
2.5.2	Approches basées sur des techniques non déterministes	31
2.5.3	Comparaison entre les approches basées sur des métaheuristiques et les approches déterministes.....	33
2.6	Conclusion.....	36
3	Approche proposée et simulation	37
3.1	Introduction	37
3.2	Motivation.....	37
3.3	K-means.....	38
3.3.1	Définition	38
3.3.2	Le fonctionnement de l'algorithme de K-Means.....	39
3.3.3	Avantages de K-means	39
3.3.4	Inconvénients.....	39
3.4	Algorithme Génétique originale (GA)	40
3.4.1	Définition	40
3.4.2	Principe de fonctionnement	40
3.5	Le fonctionnement de l'approche proposée K-GA	41

3.5.1	Tableau des variables de K-GA	42
3.5.2	Dataset.....	42
3.5.3	Classification et sélection des services candidats via le K-means	44
3.5.4	Application de GA	46
3.6	Scénario d’application de K-GA (Étude de cas)	53
3.7	Simulation	56
3.7.1	Environnement, outils et bibliothèques utilisés	56
3.7.2	Pré-traitement et normalisation des données.....	58
3.7.3	Classification via K-means	60
3.7.4	Recherche d’une composition quasi optimale via K-GA	65
3.8	Conclusion.....	67
Conclusion générale		68
Références		70

Table des figures

1.1	Définition de l'IoT [7]	5
1.2	Axes de l'internet des objets [12].	5
1.3	Architecture en quatre couches de l'IoT [13].	8
1.4	Maison-intelligente [17].....	10
1.5	Révolutions industrielles [18].....	11
1.6	Architecture d'un middleware [28].....	14
1.7	Architecture orientée services [33].....	16
2.1	Représentation de service abstrait [40].....	19
2.2	Étapes duprocessus de composition sensible aux [36].....	20
2.3	Apprentissage supervisé [48].....	26
2.4	Classes des métaheuristiques [51].....	28
2.5	Principe des métaheuristiques à solution unique [61].....	29
2.6	Principe des métaheuristiques à population [61].....	29
3.1	Exemple d'application de k-means [64].....	39
3.2	Schéma général d'un algorithme génétique [69].....	41
3.3	IoTS_Dataset.....	43
3.4	Dataset après l'ajout de l'énergie.....	44
3.5	Population initiale.....	47
3.6	valeurs de fitness initiales.....	48
3.7	Croisement avec un point de crossover [68].....	50
3.8	Mutation aléatoire [78].....	50
3.9	Plan de composition associé au scénario de K-GA.....	54
3.10	Fonction de normalisation.....	58
3.11	Dataset après la normalisation.....	59
3.12	Données normalisées avant classification.....	59
3.13	Classification avec K-means.....	60
3.14	Dataset après la classification.....	61
3.15	Classification des services après l'utilisation de K-means.....	62
3.16	Silhouette score pour déferent nombre de clusters.....	63
3.17	Les valeurs de silhouette Score sans chaque attribut.....	64

3.18	Classification des services après l'amélioration de K-means	65
3.19	Temps d'exécution en fonction de la taille des services	65
3.20	Évaluation des valeurs de fitness de K-GA et GA traditionnel.....	66

Liste des tableaux

2.1	Les fonctions d'agrégation pour un structure séquentielle	23
2.2	Les fonctions d'agrégation des propriétés QoS.....	24
2.3	Tableau récapitulatif des approches étudiées.....	35
3.1	Tableau des variables de k-GA.....	42
3.2	Services candidats disponibles pour chaque service abstrait dans le scénario de composition.....	56
3.3	Valeurs de fitness de K-GA et de GA	66

Liste des abréviations

IoT	L'internet des objets	Internet of Things
QoS	Qualité de service	Quality of Service
GA	Algorithme Génétiques	Genetic Algorithm
SoA	Architecture orientée service	Service Oriented Architecture
RFID	Radio-identification	Radio Frequency Identification
IHM	Interfaces Homme-Machine	Human-Machine Interfaces
BDD	Base De Données	Database
CVC	Chauffage, Ventilation et Climatisation	Heating, Ventilation, and Air Conditioning
IoMT	Internet des Objets Médicaux	Internet of Medical Things
SOAP	Protocole Simple d'Accès aux Objets	Simple Object Access Protocol
REST	Transfert d'État Représentationnel	Representational State Transfer
B2B	Commerce Inter-Entreprises	Business to Business
C2C	Commerce entre Particuliers	Consumer to Consumer
B2C	Commerce Entreprise-Particulier	Business to Consumer
SC	Service Candidat	Candidate Service
AS	Service Abstrait	Abstract Service
SCA	Services Composites Abstraites	Abstract Composite Services
Comp	Composition	Composition

C	Coût	Cost
Rt	Temps de Réponse	Response Time
R	Fiabilité	Reliability
RP	Réputation	Reputation
L	Emplacement	Location
A	Disponibilité	Availability
Sr	Taux de Réussite	Success Rate
S	Sécurité	Security
D	Débit	Throughput
ML	Apprentissage Automatique	Machine Learning
IA	Intelligence Artificielle	Artificial Intelligence
PSO	Optimisation par Essaim Particulaire	Particle Swarm Optimization
LP	Programmation Linéaire	Linear Programming
GWO	Optimisation par la Meute de Loups	Grey Wolf Optimizer
GT- EQCA	Théorie des Jeux - Approche de Composition de Services Sensible à l'Énergie et à la QoS	Group Teaching- Energy and QoS-Aware Services Composition Approach
nbSC	Nombre de Services Candidats	Number of Candidate Services
SCA	Composition Abstraite	Abstract Composition

Introduction générale

L'Internet des objets (IoT) représente une avancée technologique majeure qui transforme radicalement la manière dont les objets interagissent et communiquent entre eux. En permettant l'interconnexion d'objets intelligents pour collecter, échanger et analyser des données en temps réel. Cette interconnexion crée une infrastructure globale où les objets, qu'ils soient physiques ou virtuels, interagissent de manière autonome pour offrir des services avancés et améliorer l'efficacité opérationnelle dans divers domaines, tels que la domotique, la santé, les transports, et bien d'autres.

L'IoT vise à connecter divers objets du quotidien à Internet, dans le but d'offrir des fonctionnalités spécifiques et personnalisées, et de répondre aux besoins complexes des utilisateurs de manière efficace. La plupart des applications IoT ne se limitent pas à un seul service, au contraire, elles impliquent souvent la composition de plusieurs services pour répondre à ces exigences complexes.

La composition de services se réfère au processus de combinaison de deux ou plusieurs services afin de répondre à des exigences complexes qu'un seul service ne peut offrir individuellement. La composition de services permet de définir de nouveaux services à valeur ajoutés pour répondre à des requêtes complexes, tout en satisfaisant les exigences de l'utilisateur en termes de QoS. Sachant qu'en IoT, l'énergie est un critère de qualité important, nous avons considéré la consommation d'énergie d'un service IoT comme critère de QoS lors de la sélection de services durant le processus de composition.

Dans ce mémoire, nous présentons une approche de composition de services IoT, nommée K-means with Genetic Algorithm (K-GA), qui combine l'algorithme de clustering K-means avec l'algorithme génétique GA. Son objectif est de rechercher une solution quasi optimale en choisissant une bonne composition de services IoT répondant aux exigences des utilisateurs en termes de QoS et de consommation d'énergie. Afin de garantir un bon temps de réponse de l'approche proposée, nous avons utilisé K-means pour réduire l'espace de recherche et par la suite nous faisant appel à GA pour la recherche d'une bonne composition satisfaisant les préférences de l'utilisateur.

Ce mémoire, est organisé en trois chapitres, comme suit :

Le premier chapitre présente les concepts fondamentaux de l’IoT, ses principales fonctionnalités, les domaines d’application ainsi que quelques avantages et les défis existant dans ce domaine. Dans le deuxième chapitre, nous définissons le processus de composition de services IoT sensible aux QoS. De plus, un aperçu sur le Machine Learning (ML) et les métaheuristiques est également fournis dans ce chapitre. Enfin, nous avons abordé un état de l’art et un tableau récapitulatif des approches de composition de services étudiés lors de la réalisation de ce mémoire. Le troisième et dernier chapitre sera consacré à la présentation de notre approche, nommée K-means with Genetic Algorithm (K-GA), ainsi qu’à son principe de fonctionnement et son évaluation, en le comparant au GA original sans réduction de l’espace de recherche.

Concepts généraux

1.1 Introduction

L'Internet des objets (IoT) est une technologie qui a révolutionné la façon dont nous interagissons avec les objets qui nous entourent. L'IoT est un ensemble de dispositifs informatiques interdépendants, de machines mécaniques et numériques qui possèdent des identifiants uniques et qui peuvent transférer des données sur un réseau sans avoir besoin d'interactions humaines.

Dans ce chapitre, nous présentons les différents concepts de base liés à la technologie IoT qui serviront à introduire les termes utilisés tout au long du document. Puis nous allons fournir un aperçu sur les architectures orientée service (SoA).

1.2 Internet des objets

1.2.1 L'historique de l'Internet des objets

Le premier « objet » connecté à l'Internet remonte à 1982, il s'agissait d'un distributeur de boissons installé à l'université Carnegie-Mellon de Pittsburgh, en Pennsylvanie, qui indiquait le niveau de remplissage de l'appareil et la température des boissons. Le terme IoT a été utilisé pour la première fois en 1999 par Kevin Ashton, un ingénieur britannique, pour décrire un système où les objets physiques sont connectés à Internet. L'Internet des objets a connu une croissance exponentielle. En effet, entre 2008-2009, le nombre d'objets connectés à Internet a dépassé pour la première fois la population mondiale [1]. Aujourd'hui, à la fin 2024, plus de 207 milliards d'appareils devraient être connectés au réseau mondial d'outils, de jouets, de dispositifs et d'appareils électroménagers qui constituent l'internet des objets (IoT) [2].

1.2.2 Définition de l'IoT

L'Internet des Objets est considéré comme l'un des domaines les plus cruciaux pour le développement futur et connaît une croissance significative grâce à divers efforts [3]. Il existe différentes définitions de l'IoT dans la littérature, dans ce qui suit nous présentons quelques-unes.

Définition 1

L'IoT fait référence à une collection de centaines de millions de divers appareils physiques à travers le monde qui sont actuellement connectés au Web pour collecter et partager des informations. Le nombre d'appareils intelligents et de dispositifs intelligents augmente rapidement. Un appareil peut être identifié de manière unique à l'aide d'étiquettes RFID (Radio Frequency Identification) [4].

Définition 2

L'Internet des Objets peut être mis en œuvre selon trois paradigmes principaux : orienté Internet (middleware), orienté objets (capteurs) et orienté sémantique (connaissance). L'utilité de l'IoT se manifeste uniquement dans des domaines d'application où ces trois paradigmes convergent [5].

Définition 3

L'Internet des objets est une infrastructure mondiale pour la société de l'information, avec des capacités d'auto-configuration qui permet de disposer de services évolués en interconnectant des objets (physiques ou virtuels). Comme le montre la figure ci-dessous, l'IoT utilise des interfaces intelligentes dont l'objectif est de soutenir les humains dans leurs tâches quotidiennes grâce aux technologies de l'information et de la communication interopérables existantes ou en évolution. Leurs aptitudes à observer le monde physique et à fournir des informations pour la prise de décision seront une composante essentielle de l'architecture de l'Internet à venir [6].

Après avoir lu plusieurs définitions de l'IoT, nous pouvons définir l'IoT comme un système intégrant des dispositifs informatiques, des réseaux et des machines, leur permettant de se connecter et de transmettre des données de manière autonome, sans nécessiter d'interaction humaine. Ce système interconnecte des objets physiques ou virtuels à travers divers paradigmes, facilite ainsi la collecte et le partage d'informations pour soutenir les humains dans leurs tâches quotidiennes et prendre des décisions éclairées.

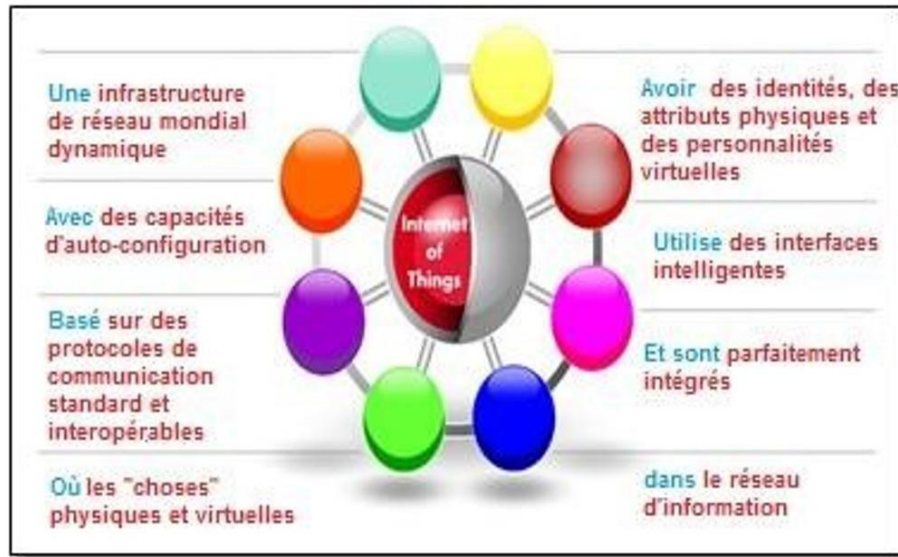


FIGURE 1.1 – Définition de l'IoT [7]

1.2.3 Axes de l'IoT

L'IoT permet aux personnes de se connecter de n'importe où ou à n'importe quel moment. Cependant, la communication dans l'IoT a la dimension toute objets comme dimension supplémentaire. L'objectif principal de l'IoT est de fournir une connectivité pour tout le monde à tout moment et en tout lieu à travers des objets connectés (smartphones, tablettes, capteurs, caméras, etc.) [12].

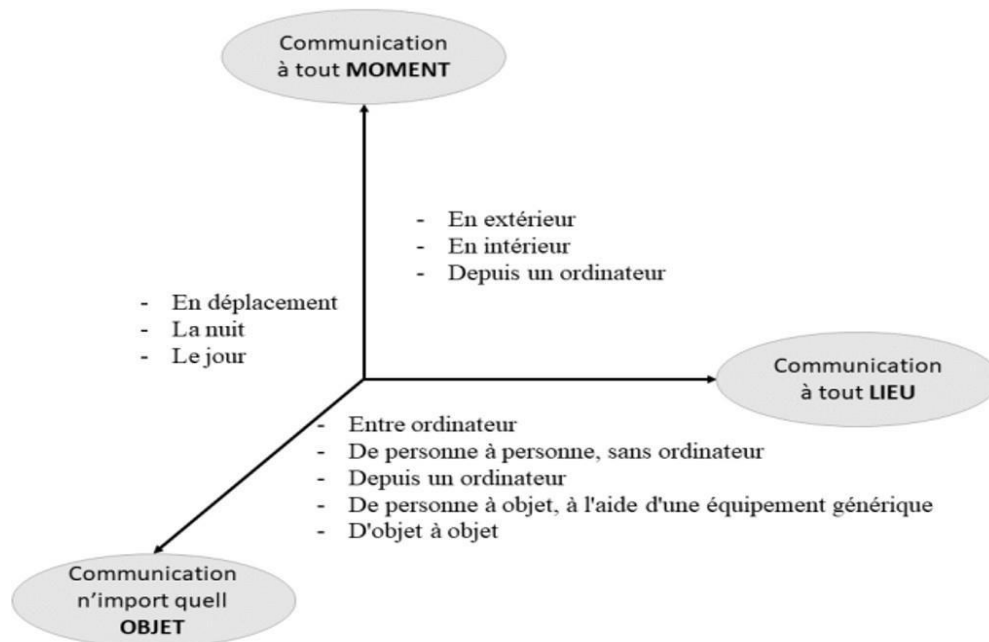


FIGURE 1.2 – Axes de l'internet des objets [12].

1.3 Qu'est-ce qu'un objet ?

Définition 1

Un objet est une entité physique, par exemple, une machine à café, une serrure, un livre, un téléphone mobile, une caméra, etc. Dans le contexte précis de l'IoT, cet objet possède au minimum un identifiant unique attaché à une identité. Exprimant d'une part ses caractéristiques invariantes (catégorie, couleur, poids, etc.) et son état, c'est-à-dire toutes ses propriétés susceptibles de changer au fil du temps (position, niveau de batterie, l'état extérieur, etc.) [5].

Définition 2

Dans le contexte de l'IoT, un objet peut être un dispositif intelligent ou un objet virtuel qui a été équipé de capteurs et d'actionneurs lui permettant de collecter, traiter, communiquer des données, d'interagir avec son environnement ainsi qu'avec d'autres objets, et de fonctionner de manière autonome grâce à la connectivité Internet. Cela permet la création d'environnements et d'applications intelligentes. Ces objets peuvent se connecter à un ordinateur, un smartphone ou une tablette via un réseau sans fil (Wi-Fi, Bluetooth, Wi-Fi longue portée, etc.), ainsi qu'à Internet ou à un réseau local [8] [9].

1.4 Capteurs et Actionneurs

Les capteurs et les actionneurs sont des éléments fondamentaux de l'IoT. Les capteurs fonctionnent comme les yeux et les oreilles des objets IoT, capturant des données sur leur environnement, telles que la température, le mouvement et la position. Les actionneurs, de leur côté, agissent comme les jambes et les mains, exécutant les fonctions assignées aux objets IoT [10]. Ces dispositifs constituent un réseau potentiellement vaste de nœuds interconnectés. Les capteurs IoT doivent relever divers défis de communication, notamment la sécurité, la confidentialité, la mobilité, la portée limitée, la fiabilité, la robustesse et la gestion des ressources limitées (énergie, capacité de stockage et de traitement, bande passante, etc.) [11].

1.4.1 Capteurs

Selon [10] Un capteur est un appareil qui convertit un paramètre physique en une sortie électrique. Un capteur est un type de transducteur. Les capteurs peuvent être divisés en capteurs analogiques et capteurs numériques. Les capteurs analogiques fournissent une sortie sous forme de tensions et de courants. Les capteurs numériques sont excellents pour les systèmes embarqués, car ils rendent le circuit beaucoup plus simple, par exemple, les capteurs

de température, les capteurs d'humidité, les capteurs de pression, les capteurs de fumée, les capteurs de son et de lumière. . . etc. Aussi les capteurs peuvent être classés selon les critères suivants :

- Leurs besoins en alimentation externe.
- Type de signal produit par le capteur.
- Type d'appareil de mesure.

1.4.2 Actionneurs

Un actionneur est un dispositif matériel ou logiciel qui convertit un signal électrique en sortie physique (un mouvement). Un actionneur peut être contrôlé par la tension ou le courant électrique, la pression pneumatique ou hydraulique, ou même la puissance humaine. Dans les systèmes embarqués, les actionneurs sont principalement contrôlés par l'électricité. Lorsque le signal de commande est reçu, l'actionneur convertit l'énergie électrique en mouvement mécanique [10].

1.5 Caractéristiques fondamentales de l'IoT

Les caractéristiques fondamentales de l'IoT sont les suivantes :

- **Interconnectivité** : Dans le contexte de l'IoT, tout peut être interconnecté avec l'infrastructure mondiale d'information et de communication [13].
- **Services liés aux objets** : l'IoT est capable de fournir des services liés aux objets dans le cadre des contraintes des objets, tels que la protection de la vie privée et la cohérence sémantique entre les objets physiques et les objets virtuels qui leur sont associés. Afin de fournir des services liés aux objets dans le cadre des contraintes des objets, les technologies du monde physique et du monde de l'information vont évoluer [13].
- **Hétérogénéité** : les appareils de l'IoT sont hétérogènes, car ils sont basés sur des plateformes matérielles et des réseaux différents. Ils peuvent interagir avec d'autres appareils et d'autres réseaux [13].
- **Communication** : les objets de l'IoT communiquent entre eux et avec des systèmes centraux via des protocoles de communication standardisés. Cela facilite l'échange de données entre les objets, la collecte des informations dans des systèmes centraux, et la transmission de commandes aux actionneurs [14].

- **Économiser l'énergie** : les appareils IoT sont souvent conçus pour minimiser la consommation d'énergie et maximiser l'efficacité énergétique. Par exemple, les capteurs de mouvement comme le Motion Sensor Light intègrent un détecteur de mouvement qui allume la lumière uniquement lorsqu'un mouvement est détecté, évitant ainsi le gaspillage d'énergie électrique. Ces dispositifs contribuent significativement à la récupération et à l'utilisation efficace de l'énergie [10].

1.6 L'architecture de l'IoT

L'architecture de l'IoT est généralement structurée en plusieurs couches qui collaborent pour collecter, transmettre et traiter les données. Cette structure permet d'illustrer la relation entre différentes technologies et de démontrer l'évolutivité, la modularité et la configuration des implémentations IoT dans divers scénarios. Il existe différentes architectures pour l'IoT : l'architecture en 3 couches, l'architecture en 4 couches et l'architecture en 5 couches. La figure suivante présente en détail l'architecture en quatre couches de l'IoT [13].

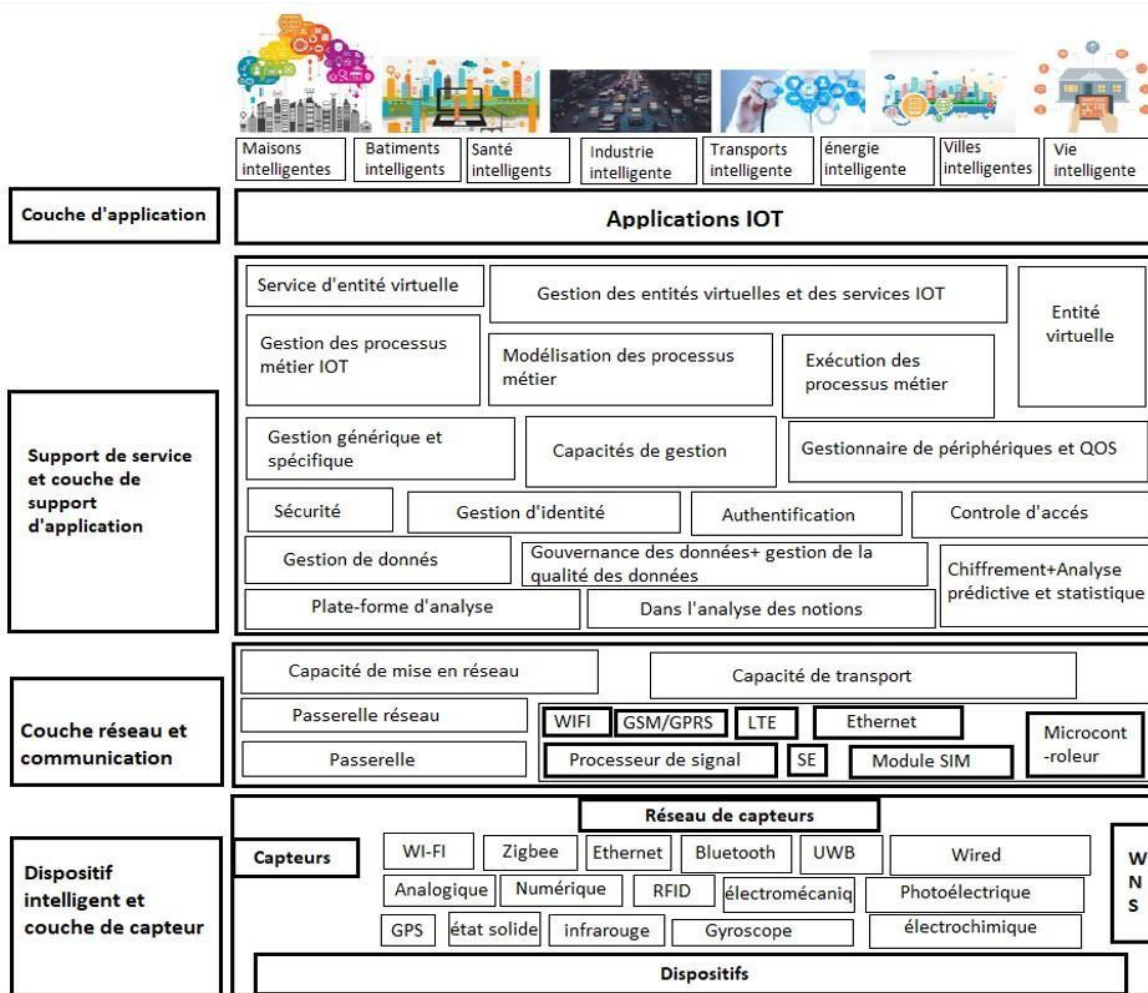


FIGURE 1.3 – Architecture en quatre couches de l'IoT [13].

La fonctionnalité de chaque couche est décrite ci-dessous [13] :

- **Couche application** : l'application de l'IoT couvre les environnements/espaces intelligents dans des domaines tels que : transport, bâtiment, ville, commerce de détail, agriculture, usine, chaîne d'approvisionnement, urgence, soins de santé, interaction avec l'utilisateur, culture et tourisme, environnement et énergie.
- **Support de services et couche de support d'application** : le service de gestion permet le traitement des informations grâce à l'analyse, aux contrôles de sécurité, à la modélisation des processus et à la gestion des appareils.
- **Couche réseau et communication** : les capteurs IoT génèrent un volume massif de données nécessitant une infrastructure réseau robuste et performante. Pour répondre à la diversité des services IoT, plusieurs réseaux avec différentes technologies sont nécessaires pour une communication efficace, pouvant prendre la forme de modèles privés, publics ou hybrides.
- **Dispositif intelligent et couche de capteurs** : la couche la plus basse est constituée d'objets intelligents intégrés à des capteurs. Les capteurs permettent l'interconnexion du monde physique et numérique, ce qui permet de collecter et de traiter des informations en temps réel.

Aussi, pour mettre en place un synthèse IoT il faut inclure les composants suivants [15] :

- **Des capteurs** : Ce sont les éléments qui collectent des données à partir de leur environnement.
- **Un réseau (connectivité)** : Les réseaux constituent l'élément le plus important, fournissant le canal de communication entre les objets connectés. Les objets doivent être installés dans une zone où il y a une couverture qui doit être alimentée en permanence par l'énergie.
- **Les données** : Les données collectées par les objets connectés constituent une source précieuse de valeur et doivent être impérativement stockées, archivées et sauvegardées dans une base de données correctement structurée. Une base de données bien organisée améliore significativement les performances des services opérationnels de l'IoT.
- **Les informations** : Ce sont celles qu'on a déduites après le traitement et l'analyse des données brutes interceptées par les objets. Il est préférable aussi de les enregistrer dans des BDD.
- **Les applications d'exploitation** : les applications d'exploitation sont en principe les interfaces homme-machine (IHM) dans lesquelles nous pouvons visualiser les données sous forme de tableau de bord, les charts, les graphes... etc.

1.7 Les domaines de l'Internet des Objets

Les Maisons Intelligentes

Les maisons intelligentes, ou domotiques, représentent probablement les applications IoT les plus appréciées, en raison de leur capacité à surveiller et contrôler divers aspects de l'habitat, comme le Chauffage, la Ventilation et la Climatisation (CVC). En connectant les équipements électroménagers, il devient possible d'automatiser de nombreuses tâches quotidiennes, telles que l'allumage des lumières, le chauffage, la cuisson et le lavage, réduisant ainsi les consommations d'énergie grâce aux réseaux et compteurs intelligents. De plus, les dispositifs de sécurité à bande, utilisant des capteurs infrarouges, de mouvement, sonores et de vibration, améliorent la sécurité en détectant et dissuadant les intrusions. Les maisons intelligentes offrent également des avantages significatifs pour les personnes âgées, handicapées et les enfants, en augmentant leur sécurité, leur autonomie et en fournissant des environnements interactifs et éducatifs [16].



FIGURE 1.4 – Maison-intelligente [17].

Industrie

La première révolution industrielle du XVIIIe siècle, initiée par la machine à vapeur, a transformé la production industrielle. Au début du XIXe siècle, la deuxième révolution a vu l'énergie électrique propulser la production de masse. La fin du XIXe siècle a marqué la troisième révolution industrielle, ou révolution numérique, avec l'avènement de l'électronique et de l'informatique. L'Industrie 4.0, ou quatrième révolution industrielle, intègre des systèmes cyber-physiques, l'IoT, l'intelligence artificielle et le cloud computing pour créer des usines intelligentes. Ces usines utilisent l'auto-optimisations et l'autoconfiguration pour améliorer

l'efficacité, réduire les coûts et produire des biens et services de meilleure qualité en s'appuyant sur des données en temps réel et des technologies avancées [18].

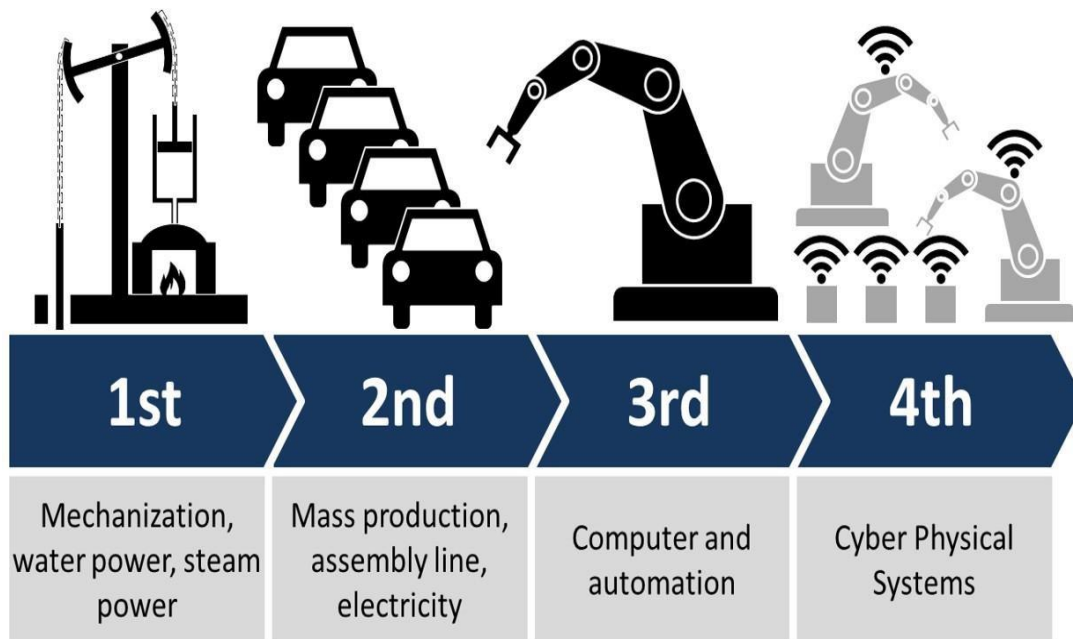


FIGURE 1.5 – Révolutions industrielles [18].

La santé

L'Internet des objets dans le domaine de la santé, aussi appelé L'Internet of Medical Things (IoMT) est une extension spécifique de l'IoT dans le domaine de la santé. Il se réfère à l'utilisation de dispositifs médicaux et d'objets connectés qui collectent, transmettent et analysent des données médicales pour améliorer les soins de santé. Il permet la surveillance à distance (télésanté/télémédecine) de la santé et les systèmes de notification d'urgence. L'IoMT englobe un large éventail d'applications et de technologies dans le domaine de la santé. Par exemple [19] :

- **Dispositifs portables** : les dispositifs portables tels que les montres intelligentes, les bracelets et les capteurs.
- **Implants médicaux connectés** : certains dispositifs médicaux implantables, tels que les stimulateurs cardiaques, les prothèses articulaires et les implants de surveillance pour les diabètes ... etc.

Énergie

L'intégration des capteurs et des actionneurs dans les systèmes énergétiques permet de réduire considérablement la consommation d'énergie des appareils. En implantant des solutions IoT au cœur du secteur de l'énergie, on assiste à une expansion et une prospérité accrue, rendues possibles par les avancées technologiques et méthodologiques. Ces innovations ont permis la réalisation de projets diversifiés, profitant tant aux institutions qu'aux particuliers. Un exemple notable est l'énergie solaire, qui, grâce aux solutions IoT, est transformée en un puissant générateur d'électricité, exploitant pleinement les énergies renouvelables. De plus, l'IoT facilite l'optimisation des ressources énergétiques en temps réel, contribuant à une consommation plus efficace et économique des énergies, tout en soutenant des pratiques durables et respectueuses de l'environnement [20].

Transport

Le transport a également été révolutionné par l'intégration de la stratégie de l'IoT. Actuellement, les développements dans ce domaine ont abouti à la création de divers systèmes qui contrôlent le trafic, assurent la sécurité routière et fournissent des indications précises sur la position, la pression, et d'autres paramètres essentiels [20].

1.8 Les Avantages et les défis de l'IoT

L'IoT présente plusieurs avantages et défis. Dans cette section, nous présentons quelques-uns.

Avantages de l'IoT [21] :

- Les appareils IoT peuvent collecter des données en temps réel, ce qui permet des prises de décision plus rapides et plus précises.
- L'IoT peut aider à réduire les coûts en automatisant des tâches et en optimisant l'utilisation des ressources, par exemple, en utilisant des capteurs IoT pour surveiller la consommation d'énergie dans un bâtiment, les systèmes de gestion intelligents peuvent ajuster automatiquement l'éclairage et la climatisation en fonction des besoins réels, réduisant ainsi les coûts énergétiques.
- L'IoT peut permettre de fournir des services personnalisés et d'améliorer l'expérience utilisateur grâce aux données collectées par les dispositifs connectés.
- L'automatisation et la gestion intelligente des ressources via l'IoT permettent d'optimiser le temps en réalisant les tâches de manière plus efficace et rapide.

Les défis de l'IoT :

- **Sécurité** : La sécurité est un pilier essentiel de l'Internet et elle est considérée essentiellement comme le défi le plus important pour l'IoT. Les appareils IoT sont faciles à pirater en raison de leurs capacités limitées en calcul, stockage et réseau, ce qui les rend plus vulnérables aux attaques [22].

- **Interopérabilité** : L'interopérabilité fait référence à la capacité de différents systèmes, appareils ou composants à fonctionner ensemble de manière transparente et à échanger des données efficacement [23].

- **Problèmes énergétiques au niveau des appareils**

- L'un des principaux défis de l'IoT consiste à interconnecter les objets de manière interopérable tout en prenant compte des contraintes énergétiques, sachant que la communication est la tâche qui consomme le plus d'énergie sur les appareils [24].

- **Gestion des données** : La gestion des données est un aspect crucial de l'internet des objets. Si l'on considère un monde d'objets interconnectés échangeant constamment tous types d'informations, le volume des données générées et les processus impliqués dans le traitement de ces données deviennent critiques [24].

- **Standardisation** : La standardisation dans l'IoT est un défi important. Elle cherche à définir des règles communes pour les dispositifs et protocoles, mais la variété des technologies rend cela complexe. Trouver l'équilibre entre compatibilité et innovation est essentiel et nécessite une collaboration entre les parties prenantes pour le succès à long terme de l'IoT [25].

1.9 Middleware (ou intergiciel)

En d'autres termes, le middleware, ou intergiciel en français, se définit comme une couche logicielle située entre le système d'exploitation et les applications qui permet l'échange et la gestion des données entre des applications et des systèmes d'exploitation non compatibles entre eux. Sont utilisés pour supporter des applications complexes et distribuées.

Définition 1

Tüma [26] définit le middleware comme étant « une couche logicielle située entre le système d'exploitation et les applications de chaque côté d'un réseau informatique distribué, fournissant divers services pour l'intégration d'applications dans des environnements hétérogènes ».

Définition 2

Un Middleware est un logiciel de communication qui permet aux processus exécutés sur une ou plusieurs machines d'interagir à travers un réseau [27].

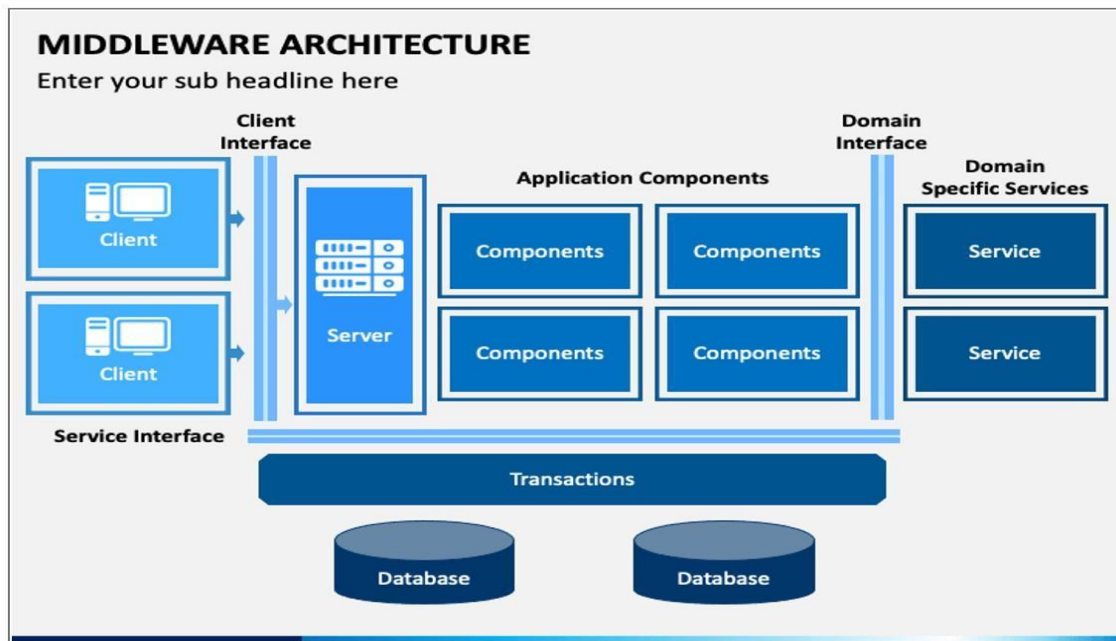


FIGURE 1.6 – Architecture d'un middleware [28].

1.10 Architecture orienté service (SoA)

L'architecture orientée services (SoA) est un paradigme clé pour le développement des systèmes d'information modernes, et elle joue un rôle crucial dans l'IoT. En structurant les applications en services indépendants, SoA améliore la modularité, l'interopérabilité et la réutilisabilité des composants logiciels, ce qui est essentiel pour gérer la complexité et l'hétérogénéité des dispositifs IoT. Cette section explore les notions fondamentales, les définitions clés et les caractéristiques de SoA, afin de démontrer son importance et ses avantages dans les technologies de l'information contemporaines, notamment pour l'intégration et la gestion efficace des systèmes IoT.

1.10.1 La notion de service

Définition 1

Un service est un processus par lequel un fournisseur remplit une mission pour un client, créant de la valeur pour les deux parties prenantes. Englobant toutes les activités de service

perçues comme présentant un intérêt. Un service peut être défini comme l'application de compétences au bénéfice d'une autre personne [29].

Définition 2

Un service est une unité autonome de fonctionnalité logicielle ou d'un ensemble de fonctionnalités, conçue pour réaliser une tâche précise comme récupérer des informations ou exécuter une opération. Il contient les intégrations de code et de données nécessaires pour exécuter une fonction métier distincte et complète [30].

Définition 3

Un service peut être aussi défini comme étant un module qui peut être invoqué, qui est assigné à une fonction spécifique et qui offre une interface bien définie [31].

En conséquence, un service peut être défini comme un ensemble de fonctionnalités ou de capacités fournies à un utilisateur ou à un appareil pour faciliter la communication, l'échange de données ou d'autres interactions. Ainsi, des services peuvent être utilisés tels quels ou bien être composés pour mener à bien des processus complexes et atteindre des objectifs spécifiques de plus haut niveau.

1.10.2 La notion de SoA

L'architecture orientée service est une architecture logicielle qui est définie par un ensemble de principes de conception. Une SoA traditionnelle implique trois acteurs principaux qui interagissent directement Comme illustré dans la figure ci-dessous : un fournisseur de services, un consommateur de services et un registre de services. Cette architecture de conception organise les fonctionnalités sous forme de services bien définis, interopérables et réutilisables dans divers contextes et applications. Les services communiquent entre eux via des interfaces standardisées, souvent basées sur des protocoles comme SOAP (Simple Object Access Protocol) ou REST (Représentationnel State Transfer), garantissant ainsi une bonne intégration entre les systèmes hétérogènes [32].

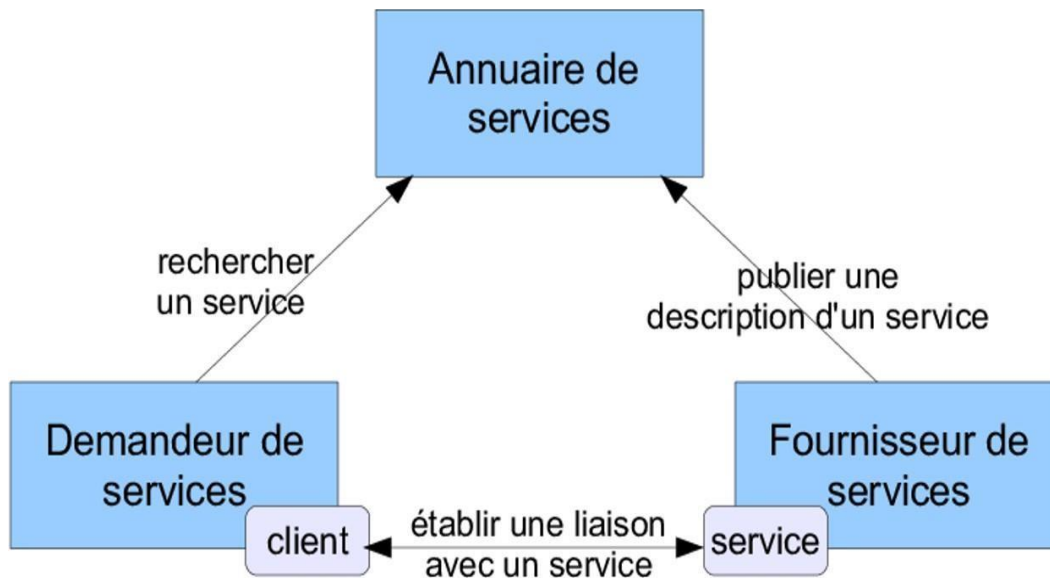


FIGURE 1.7 – Architecture orientée services [33].

1.10.3 Acteurs de l'architecture SoA

L'architecture orientée services comme illustrée dans la figure ci-dessous, est composée de trois acteurs principaux :

- **Fournisseur de services** : il crée des services qu'il met à disposition dans un registre de services. Il est responsable des conditions d'utilisation du service, de sa documentation et de sa maintenance.
- **Demandeur de services** : c'est un client consommateur ou un demandeur (une application, un autre service ou un utilisateur humain) qui fait la demande d'un service bien précis pour répondre à ses besoins.
- **Annuaire de services** : c'est un annuaire central où les fournisseurs publient leurs nouveaux services et où les consommateurs les découvrent et les utilisent.

1.10.4 Caractéristiques de l'architecture SoA

L'architecture orientée service est caractérisée par :

- La réutilisation des services
- La découverte des services disponibles.

- Un couplage faible entre les services : les services sont couplés de manière faible, ce qui signifie qu'ils peuvent communiquer entre eux sans être étroitement liés.
- L'indépendance par rapport aux aspects technologiques : c'est-à-dire les services sont indépendants des plates-formes.
- La mise à l'échelle est rendue possible grâce à la découverte et à l'invocation des nouveaux services lors de l'exécution.
- Interopérabilité : les services peuvent communiquer entre eux via des interfaces bien définies, favorisant l'interopérabilité.

1.11 Conclusion

L'Internet des objets est un domaine en croissance rapide qui offre de nombreuses opportunités mais aussi des défis importants. Ce chapitre a exploré les concepts fondamentaux de l'IoT, en détaillant ses composantes, une architecture d'un système IoT, ainsi que les caractéristiques et les défis associés. La prochaine section portera sur les architectures orientées services. Dans le chapitre suivant, nous aborderons le concept de la composition des services IoT, ainsi qu'un état de l'art sur les approches de la composition des services sensibles aux qualités de service et l'énergie.

État de l'art sur la composition de services

2.1 Introduction

Dans ce chapitre, nous détaillons les étapes du processus de composition de services. Ensuite, nous définissons des concepts liés à la problématique de la composition de services, puis nous étudions certains aspects du Machine Learning (ML), notamment l'algorithme K-means. Étant donné que notre approche propose une solution au problème de composition de services IoT avec prise en compte des QoS et de l'énergie combinant un algorithme K-means et une métaheuristique « algorithme génétique », une partie de ce chapitre est consacrée aux métaheuristiques. Enfin, nous concluons par un état de l'art des travaux dédiés à la problématique de la composition de services avec prise en compte des QoS et de l'énergie.

2.2 Composition des services IoT

Définition

La composition de services se réfère au processus de combinaison de deux ou plusieurs services, qu'ils proviennent de différentes applications ou d'une même application, pour créer un service plus complexe répondant à une exigence spécifique des utilisateurs en matière de fiabilité, d'adaptabilité, de coût, de temps de réponse et de l'efficacité énergétique, etc. Ce processus implique la coordination et l'orchestration des services individuels. La composition de services constitue un moyen puissant de gérer des activités de services complexes dans les domaines traditionnels B2B (Business to Business), C2C (Consumer to Consumer), B2C (business to consumer) [34].

2.2.1 Définition de concepts liés à la problématique de composition de services

1 Service candidat (SC)

Un service candidat dans la composition des services est un service envisagé pour la composition, sélectionné sur la base des caractéristiques fonctionnelles, et peut être un service préexistant qui peut être combiné pour fournir de nouveaux services à valeur ajoutée, qui n'est pas nécessairement incluse dans le plan de composition (SCA).

Un service concret, quant à lui, est un service candidat sélectionné pour effectuer une tâche spécifique dans une composition concrète [35].

2 Service abstrait (AS)

Un service abstrait représente une classe de service candidats, qui possèdent une équivalence fonctionnelle, c'est-à-dire ceux qui ont les mêmes paramètres d'entrée/sortie, mais qui diffèrent en fonction des valeurs de leurs attributs de QoS. Un service abstrait « i » noté AS_i est caractérisé par quatre composantes [36], comme illustré dans la figure 2.1 où :

AS_i^{in} représente les messages d'entrée.

AS_i^{out} représente les messages de sortie.

AS_i^{cat} représente la catégorie de service.

Une liste de services candidats (AS_i^{CS}) appartenant au service abstrait AS_i .

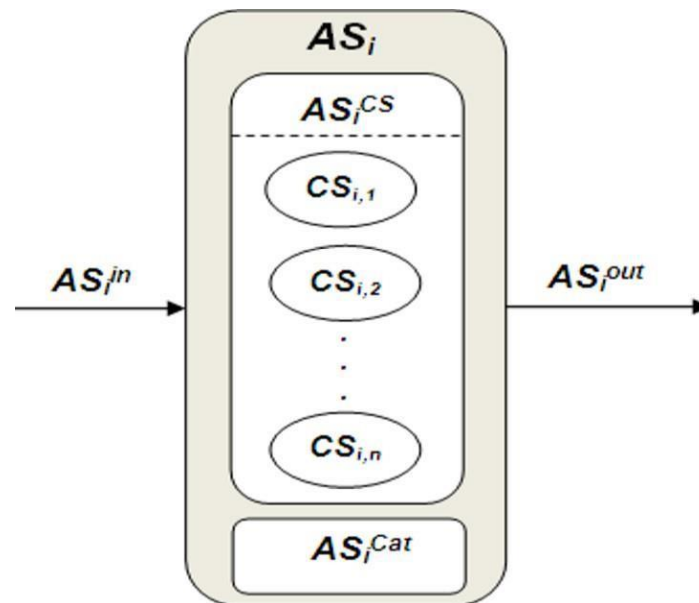


FIGURE 2.1 – Représentation de service abstrait [40].

3 Les services composites abstraits (SCA)

Les services composites abstraits, aussi dit composition abstraite, comprennent un ensemble de n services abstraits, notés $SCA = \langle AS_1, AS_2, \dots, AS_m \rangle$. Ces services sont connectés les uns aux autres à travers l'une des structures de composition de services telles que séquentielle, parallèle, conditionnelle, en boucle, ou une combinaison de celles-ci. La composition abstraite est le résultat de la première étape du processus de composition (élaboration du plan de composition). La définition de l'SCA est basée sur les exigences fonctionnelles de l'utilisateur, en utilisant un ensemble de services abstraits connectés. Un SCA permet de combiner les fonctionnalités offertes par divers services existants, qu'ils soient atomiques ou composites, dans le but de satisfaire les besoins complexes d'un utilisateur en termes de QoS et d'énergie [36].

2.2.2 Composition concrète (Comp)

La composition concrète est définie comme un vecteur qui contient les indices des services candidats inclus dans cette composition, notés $Comp = \{S_1, S_2, \dots, S_n\}$ où S_1, S_2, \dots, S_n sont les indices des services candidats inclus dans la composition. Cette représentation vectorielle nous permet de modéliser et de décrire de manière concise les compositions de services. [36].

2.2.3 Etapes de la composition des services

La réalisation d'une application par composition de services implique plusieurs étapes permettant de passer progressivement d'une spécification abstraite à une composition concrète de services, c'est-à-dire une composition prête à être exécutée [36] [37] :

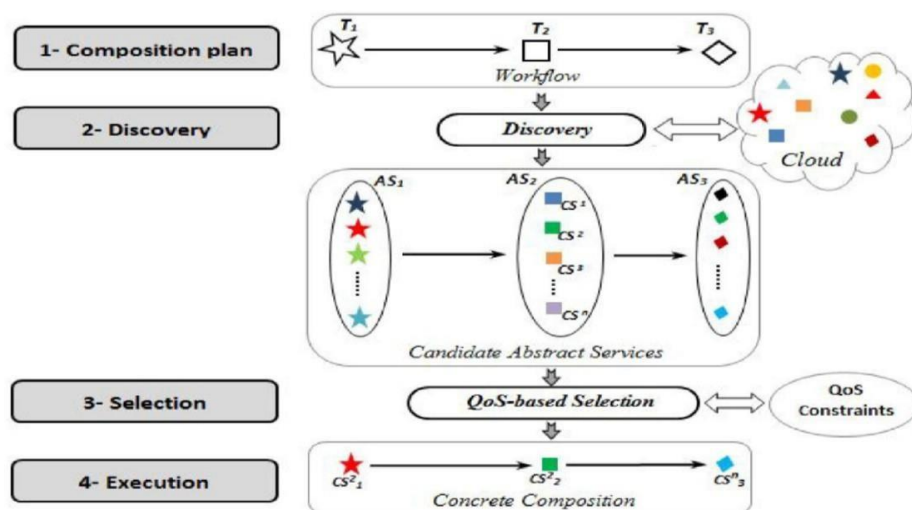


FIGURE 2.2 – Étapes du processus de composition sensible aux [36].

Phase d'élaboration du plan de composition

Cette phase comprend l'identification des exigences fonctionnelles attendues du service résultant de la composition. Ces exigences peuvent être spécifiées sous la forme d'une description abstraite contenant une composition d'un ensemble de services abstraits, chacun représentant une fonctionnalité différente. Ces fonctionnalités peuvent être spécifiées sous forme de requêtes exprimant les besoins des utilisateurs.

Phase de découverte de services

Cette phase consiste à découvrir les services disponibles qui correspondent aux exigences fonctionnelles définies dans la phase d'élaboration. Cela se fait en utilisant un registre existant, pour localiser les services disponibles en fonction de la correspondance fonctionnelle entre les tâches et les descriptions des services. Aussi, la découverte des services permet d'identifier, pour chaque service abstrait de la composition, un ensemble de services candidats ayant des fonctionnalités similaires avec des valeurs de QoS différentes.

Phase de sélection de services

Cette phase consiste à identifier les services candidats les plus adaptés à la composition parmi tous les services candidats obtenus lors de la phase de découverte, en tenant compte des critères de QoS. L'objectif est de sélectionner le service idéal pour chaque tâche abstraite dans le but d'avoir une composition concrète qui satisfait les exigences de l'utilisateur.

Phase d'exécution et contrôle de la composition

La composition concrète instanciée est exécutée par le moteur d'exécution, qui est responsable de l'appel des services de composants. L'exécution d'une composition peut être considérée comme une série d'échanges de messages entre services composants. Cet échange de données s'effectue entre un service composant et le service suivant dans la description du service composite.

Dans ce mémoire, nous nous concentrons sur la problématique de sélection de services, comme mentionné dans la section précédente la sélection de services se fait selon les valeurs de QoS d'un SC. Dans cette section, nous allons définir certaines qualités de service.

2.2.4 Qualité de service (QoS)

Dans l'article [38], les auteurs définissent les qualités de service comme clés qui permettent de sélectionner un service composite parmi plusieurs services candidats qui répondant

aux besoins et préférences de l'utilisateur. Les valeurs des QoS peuvent varier considérablement d'un service à l'autre, ce qui permet de les différencier. Par conséquent, l'évaluation des QoS permet de différencier un service d'un autre. Ainsi, il aide les demandeurs de service à reconnaître le meilleur service IoT pour son application.

Nous allons fournir une définition de certains attributs QoS dans ce qui suit [39] :

Coût (C) : le coût que l'utilisateur doit payer pour l'acquisition du service.

Temps de réponse (Rt) : l'intervalle de temps entre la soumission d'une demande d'utilisateur et la réception de la réponse. Il comprend le temps de traitement de la requête.

Fiabilité (R) : taux de réussite de la transmission de données aux utilisateurs sur une période donnée.

Réputation (Rp) : Elle est considérée comme un indicateur global de l'expérience de l'utilisateur. La réputation du service est définie comme la note moyenne de la multi-évaluation.

Emplacement (L) : C'est la distance entre le service et la destination spécifiée par l'utilisateur.

Disponibilité (A) : C'est un pourcentage de temps, qui indique quand le service est disponible.

Taux de réussite (Sr) : C'est le nombre de réponses/nombre de messages de demande.

Sécurité (S) : Cela désigne le degré de sécurité fourni par un service, notamment à travers des fonctionnalités telles que l'authentification et le cryptage, etc. La sécurité vise à protéger les utilisateurs contre les accès non autorisés et les intrusions illégales.

Débit : Le débit d'un service désigne la quantité de données qui peuvent être transférées d'un point à un autre dans un réseau par unité de temps, généralement mesurée en octets/sec.

En plus des QoS mentionnées précédemment, nous avons aussi pris en considération la consommation d'énergie, car c'est une information cruciale dans le domaine de l'IoT.

Energie : L'énergie représente le niveau de la batterie du dispositif IoT sur lequel un service spécifique est exécuté, ce paramètre vise à évaluer si le service peut être exécuté jusqu'à son achèvement sans épuiser la batterie. La mesure de l'énergie se fait de plusieurs manières, selon les dispositifs et les technologies impliquées, par exemple capteurs d'énergie et mesures indirectes en surveillant la durée de vie de la batterie pour estimer la consommation d'énergie [41].

2.2.5 Vecteur QoS d'un service candidat « QoS (SC) »

Le vecteur QoS d'un service candidat est un vecteur contenant un ensemble de mesures des propriétés non-fonctionnelles qui évaluent la qualité du service proposé. Ce vecteur inclut des paramètres tels que la performance, la fiabilité, la disponibilité, la sécurité, la latence, la bande passante, la sociabilité, et d'autres aspects pertinents en fonction des exigences spécifiques du système ou de l'application. Il est utilisé pour comparer et évaluer différents services candidats afin de déterminer celui qui répond le mieux aux besoins et aux contraintes de l'utilisateur ou du système [42].

2.2.6 Vecteur d'attributs QoS d'une composition

Le vecteur d'attributs de QoS d'une composition est défini comme suit :

$QoS(comp) = q_1, \dots, q_n$, où q est le nombre des attributs de qualité de service et qp est la valeur de l'attribut p_{ieme} ($1 \leq p \leq n$) après agrégation. La fonction d'agrégation des attributs dépend du QoS utilisée (sommation, produit, maximum ou minimum) et de la structure de composition (séquentielle, parallèle, conditionnelle et/ou en boucle). Les deux tableaux suivants résument les différentes fonctions d'agrégation possible pour un plan de composition avec structure séquentielle seulement [42].

Type	Fonction d'agrégation
Addition	$q^* = \sum_{i=1}^n q_i$
Multiplication	$q^* = \prod_{i=1}^n q_i$
Moyenne	$q^* = \frac{1}{n} \sum_{i=1}^n q_i$
Minimum	$q^* = \min(q_1, q_2, \dots, q_n)$
Maximum	$q^* = \max(q_1, q_2, \dots, q_n)$

TABLEAU 2.1 – Les fonctions d'agrégation pour un structure séquentielle [42].

Les fonctions d'agrégation du service composite pour certaines attributs de QoS sont formulées dans le Tableau respectivement.

Type	Fonction d'agrégation
Disponibilité (A)	$A_{cs} = \prod_{i=1 \Delta S_{ij} \in S_i}^n A_{ij}$
Coût (C)	$C_{cs} = \sum_{i=1 \Delta S_{ij} \in S_i}^n C_{ij}$
Fiabilité (R)	$R_{cs} = \prod_{i=1 \Delta S_{ij} \in S_i}^n R_{ij}$
Temps de réponse (Rt)	$Rt_{cs} = \sum_{i=1 \Delta S_{ij} \in S_i}^n Rt_{ij}$
Réputation (Rp)	$Rp_{cs} = \frac{1}{n} \cdot \sum_{i=1 \Delta S_{ij} \in S_i}^n Rp_{ij}$
Sécurité (S)	$S_{cs} = \min(S_{1j}, S_{2j}, \dots, S_{ij})$
Taux de réussite (Sr)	$Sr_{cs} = \prod_{i=1 \Delta S_{ij} \in S_i}^n Sr_{ij}$
Débit (T)	$T_{cs} = \min(T_{1j}, T_{2j}, \dots, T_{ij})$
Emplacement (L)	$L_{cs} = \sum_{i=1 \Delta S_{ij} \in S_i}^n L_{ij}$

TABLEAU 2.2 – Les fonctions d'agrégation des propriétés QoS [42].

2.2.7 Fonction fitness (d'utilité)

La fonction de fitness est essentielle dans le processus de sélection de composition. Dans ce processus, la fonction d'utilité fait référence à un critère ou à une métrique utilisée pour évaluer et comparer différentes compositions de services afin de déterminer laquelle répond le mieux aux besoins spécifiques de l'utilisateur [43].

Chaque service dans un environnement IoT est évalué à l'aide d'une valeur d'utilité $f(s)$ calculée comme suit [43] :

$$f(s) = \sum_{q=1}^k w_q \cdot qos'_q(s) \quad (2.1)$$

Où w_q fait référence à la préférence de l'utilisateur pour le $q^{\text{ème}}$ critère de qualité de service dans un domaine d'application particulier et $qos'_q(s)$ représente la valeur normalisée du $q^{\text{ème}}$ critère de qualité de service calculée comme suit :

$$\left\{ \begin{array}{ll} \frac{qos_q(max) - qos_s}{qos_q(max) - qos_{min}} & \text{Si } q \text{ est } Q^- \\ \frac{qos_q(s) - qos_{min}}{qos_q(max) - qos_{min}} & \text{sinon} \end{array} \right. \quad (2.1)$$

Où $qos_q(Max)$, $qos_q(Min)$ et $qos_q(s)$ désignent respectivement la valeur maximale, la valeur minimale et la valeur courante du $q^{ème}$ critère de qualité de service [43].

La résolution de problème de sélection de service IoT sensible aux QoS a fait l'objet de plusieurs travaux de recherche. Certains travaux proposent d'intégrer les outils de l'IA dans leur solution, et d'autres peuvent choisir de se baser sur des métaheuristiques. Soit combinés à d'autres métaheuristiques ou à des techniques de IA.

Dans la suite de ce chapitre, nous allons consacrer une section pour le machine learning et la seconde aux métaheuristiques.

2.3 Machine Learning

Machine Learning visant à réduire l'espace de la recherche tout en conservant les services les plus performants.

Définition

Le Machine Learning (ML), également appelé apprentissage automatique, est une sous-catégorie de l'intelligence artificielle (IA) qui se concentre sur la création de systèmes qui apprennent ou améliorent leurs performances en fonction des données qu'ils traitent. Permet aux machines d'apprendre par elles-mêmes sans programmation explicite. Le Machine Learning est souvent utilisé pour automatiser et accélérer la prise de décision [44].

2.3.1 Types de systèmes d'apprentissage automatique

Le domaine de l'apprentissage automatique est souvent divisé en sous-domaines selon les types de problèmes abordés. On peut les classer en fonction du degré de supervision nécessaire pendant la phase d'entraînement comme suit [45] :

2.3.1.1 L'apprentissage supervisé

Les experts en apprentissage automatique estiment qu'environ 70% des algorithmes d'apprentissage automatique utilisés actuellement sont supervisés. Ils fonctionnent avec des ensembles de données connus ou étiquetés afin d'indiquer à la machine quelles patterns elle doit rechercher, avec les informations qu'il est censé déterminer. L'apprentissage supervisé a pour but d'identifier une fonction déterministe qui attribue une entrée à

toute sortie et de l'appliquer sur un processus analytique pour la prédiction de futures observations en réduisant le taux d'erreur. L'objectif est d'entraîner le modèle choisi pour qu'il puisse prédire correctement la sortie pour des entrées non étiquetées [46].

L'apprentissage supervisé revient à apprendre à une machine à créer une fonction f qui permet de prédire un ou plusieurs résultats d'intérêt, notés Y , en fonction des données d'entrée X disponibles. L'apprentissage supervisé est généralement utilisé pour la régression ou la classification [47] :

- A. La régression :** Le processus de régression consiste à trouver le modèle ou la fonction permettant de distinguer les données en valeurs réelles continues ou identifier le mouvement de distribution en fonction des données historiques.
- B. La classification :** C'est une tâche consistant à choisir une classe (valeur) parmi toutes celles possibles. La classification implique l'apprentissage d'une fonction objective, notée f , qui attribue un ensemble d'attributs x à l'une des classes préétablies étiquetées y . Le processus de classification consiste à déterminer le modèle ou la fonction qui permet de séparer les données en plusieurs classes catégorielles (valeurs discrètes) [47].

L'objectif principal de la classification est d'assigner une classe à un objet en se basant sur ses caractéristiques descriptives. Toutes les méthodes de classification partagent une caractéristique commune : elles exigent un processus d'apprentissage et d'évaluation de leurs performances. Pour cela, deux ensembles d'échantillons sont nécessaires [47] :

- La base d'apprentissage doit inclure un nombre adéquat d'échantillons x_i pour chaque classe. Cette collection d'échantillons doit être à la fois qualitative, c'est-à-dire représentative de l'application envisagée, et quantitative, avec un nombre significatif d'échantillons tout en tenant compte des probabilités a priori de chaque classe.
- Le choix de la base de test est crucial car elle est utilisée pour évaluer les performances de l'algorithme.

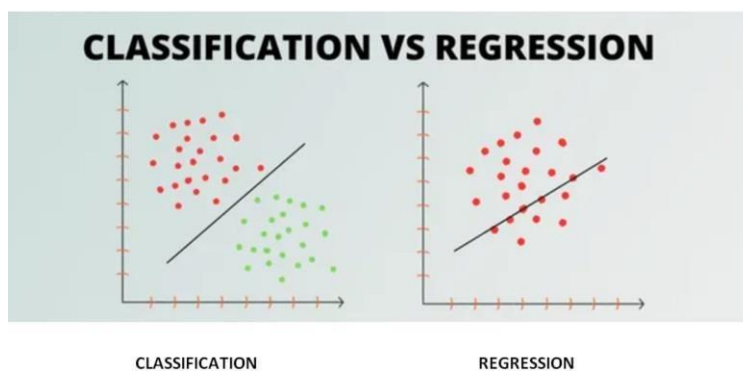


FIGURE 2.3 – Apprentissage supervisé [48].

2.3.1.2 L'apprentissage non supervisé

L'apprentissage non supervisé est un processus itératif d'analyse de données et de création de modèles sans intervention humaine. Dans l'apprentissage non supervisé, les données ne sont pas étiquetées, ce qui signifie qu'il n'y a pas de sortie associée aux données d'entrée. L'objectif du système est d'identifier des caractéristiques communes aux données d'entraînement. Dans ce type d'apprentissage, l'algorithme doit découvrir par lui-même la structure des données. L'objectif de l'apprentissage non supervisé est d'analyser les données d'entrée et de réduire leur dimensionnalité. Cet apprentissage est utilisé pour des tâches telles que le clustering, la détection de patterns, la reconnaissance de formes, etc [49].

2.3.1.3 Apprentissage semi-supervisé

L'apprentissage semi-supervisé se situe entre l'apprentissage supervisé et non supervisé. Il consiste à résoudre des problèmes impliquant des données non étiquetées en utilisant à la fois des données étiquetées et non étiquetées. Cette approche est utile lorsque le nombre de données étiquetées est limité, permettant ainsi de contourner les défis liés au manque de données étiquetées pour l'entraînement d'un algorithme supervisé [47].

2.3.1.4 Apprentissage par renforcement

L'apprentissage par renforcement est une approche de l'intelligence artificielle axée sur l'apprentissage d'un système en interagissant de son interaction avec l'environnement. Dans ce processus, le modèle est entraîné en ajustant ses paramètres à chaque itération en fonction des récompenses et des pénalités reçues pour ses performances dans une tâche donnée. Les récompenses et les pénalités sont cruciales, car elles guident le modèle vers des résultats satisfaisants [47].

2.4 Les Métaheuristique

Au cours de nos recherches, nous avons constaté que les métaheuristiques sont couramment utilisées pour résoudre le problème de la composition de services IoT en prenant en compte les critères de QoS et l'énergie. Une métaheuristique est un algorithme d'optimisation visant à résoudre des problèmes d'optimisation difficiles pour lesquels on ne connaît pas de méthode classique plus efficace [43].

2.4.1 Définition

Une métaheuristique est un processus itératif qui guide intelligemment une heuristique subordonnée pour explorer et exploiter l'espace de recherche dans le but de trouver des so-

lutions proches de l'optimum global d'un problème d'optimisation difficile et complexe. Ces algorithmes stochastiques progressent vers l'optimum global en échantillonnant une fonction objective, combinant différents concepts et stratégies d'apprentissage pour structurer l'information de manière efficace. Ils offrent des solutions approximatives, souvent utilisées lorsque les méthodes exactes deviennent impraticables en raison de la taille ou de la complexité du problème [50].

Les métaheuristiques ont été classées par la figure 2.4 en deux classes selon le type de solution unique ou à population.

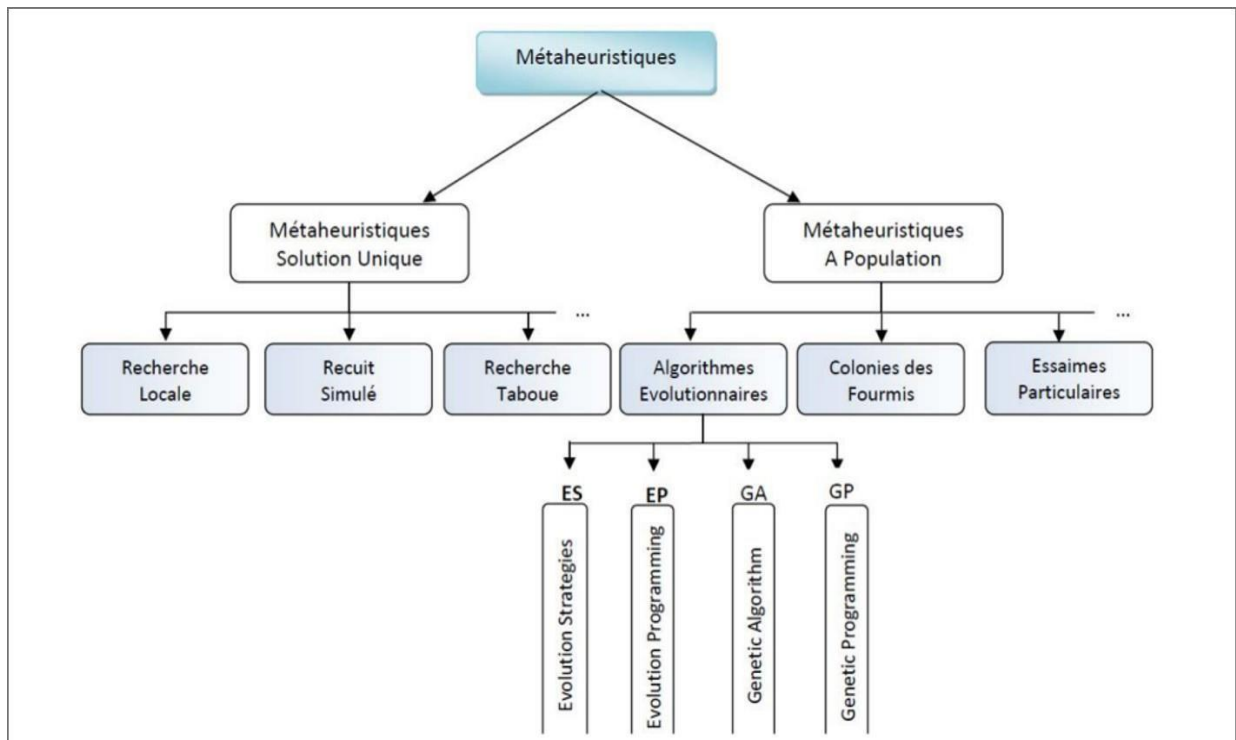


FIGURE 2.4 – Classes des métaheuristiques [51].

Métaheuristiques à base de solution unique :

Les méthodes itératives à solution unique sont toutes basées sur un algorithme de recherche de voisinage qui commence avec une solution initiale, puis l'améliore pas à pas en choisissant une nouvelle solution dans son voisinage [61].

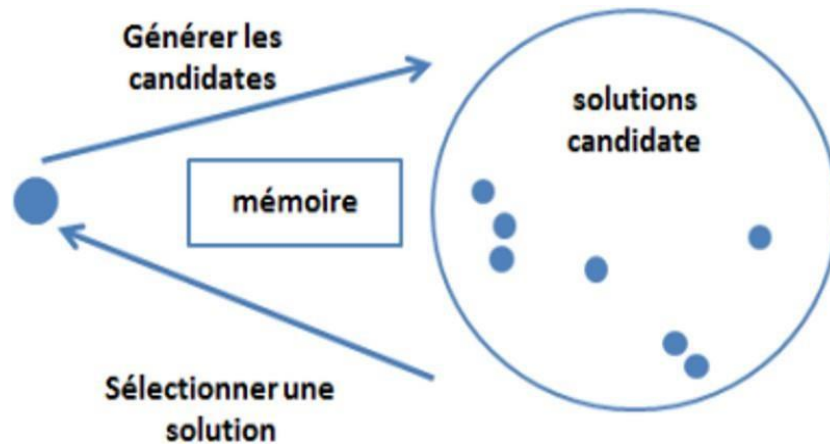


FIGURE 2.5 – Principe des métaheuristiques à solution unique [61].

Métaheuristiques à base de population :

Dans cette classe, les métaheuristiques utilisent la notion de population, elles manipulent toutes un échantillonnage de la fonction objectif, via des processus communs. Autrement dit, les méthodes d'optimisation à population de solutions améliorent, au fur et à mesure des itérations, une population de solutions. L'intérêt de ces méthodes est d'utiliser la population comme facteur de diversité [61].

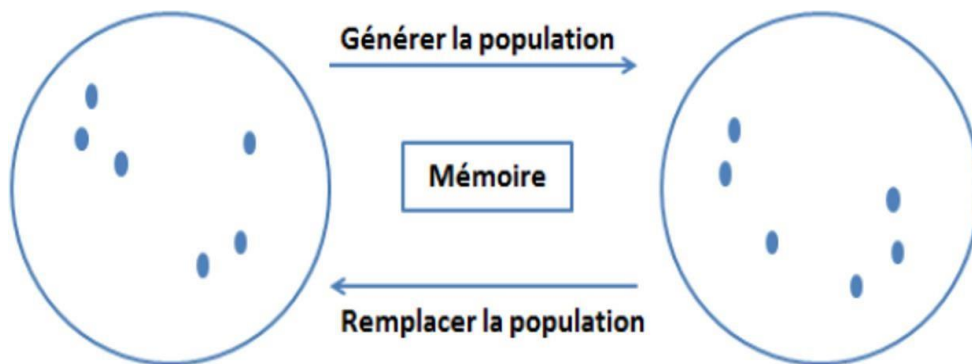


FIGURE 2.6 – Principe des métaheuristiques à population [61].

2.5 Etat de l'art sur les approches de composition de services IoT prenant en compte les QoS et l'énergie

Dans cette partie, nous allons présenter plusieurs approches proposées pour la composition de services dans l'IoT. Pour cela, nous avons étudié une variété d'approches, qu'elles soient basées sur des techniques déterministes ou non déterministes.

2.5.1 Approches basées sur des technique déterministes

Les techniques déterministes sont des techniques qui, compte tenu d'une entrée particulière, génèrent toujours la même sortie, ce qui signifie que, lorsqu'elles sont appliquées à un problème spécifique, elles donneront toujours le même résultat. Ces méthodes suivent une procédure stricte et prédéfinie pour résoudre les problèmes d'optimisation, consistant à chercher la meilleure solution possible dans tout l'espace de recherche, ce qui les rend particulièrement idéales pour les problèmes dont le nombre d'instances est petits [35]. Cette catégorie n'est pas appropriée. Donc, nous utilisons les approches non déterministes basées sur la métaheuristique, parce que l'espace de recherche de notre problème est vaste.

Dans cette étude [53], les auteurs ont développé un cadre de composition de services adaptatif et contextuel pour les villes intelligentes, qui s'appuie sur le modèle de service abstrait wEASEL. Ce modèle représente les services et les tâches des utilisateurs en termes de leur signature, de leur spécification (comprenant les prés conditions, les post conditions et les effets sensibles au contexte), ainsi que de leur conversation (définissant le comportement avec des contraintes de flux de données et de flux de contexte associées).

Dans cette étude [54], les chercheurs proposent une approche basée sur un nouvel algorithme de composition de services IoT multcloud appelé E2C2. L'algorithme vise à créer un plan de composition économe en énergie en recherchant et en intégrant un minimum de services IoT qui répondent aux besoins de l'utilisateur. Il traduit les exigences des utilisateurs en plans de configuration optimaux à l'aide d'une modélisation et d'une analyse formelles. L'approche E2C2 est évaluée par rapport à quatre autres algorithmes de composition de services établis dans différents environnements cloud et montre ses performances supérieures. Les résultats de l'expérimentation ont montré que l'approche E2C2 a une efficacité énergétique et une performance supérieure, mais comme toute approche algorithmique, elle est difficile à mettre en œuvre et la dépendance aux données d'entrée peut parfois poser des défis en termes de collecte et de gestion de ces données.

Les auteurs ont présenté dans [55] une méthode de composition de services rapide, centrée sur l'énergie et sensible aux QoS nommée (FSCA-EQ). Cette approche se base sur un mécanisme d'optimisation hiérarchique pour optimiser à la fois la qualité de service et la

consommation d'énergie. Elle fait intervenir la présélection des services candidats en fonction du niveau de qualité de service, l'utilisation du concept de dominance relative pour sélectionner le meilleur service composite parmi les services candidats et la prise en compte du profil énergétique des services IoT. Les résultats des expérimentations ont montré que l'approche, en prenant en compte la consommation d'énergie dans la composition des services IoT, utilise un mécanisme d'optimisation hiérarchique pour trouver des solutions optimales de Pareto et un concept de dominance relative pour sélectionner le meilleur service composite parmi les services candidats. Cependant, il n'y a pas de prise en compte explicite des changements dynamiques dans les besoins des utilisateurs.

2.5.2 Approches basées sur des techniques non déterministes

Les techniques déterministes sont efficaces pour les problèmes de petite taille, tandis que les algorithmes non déterministes permettent de résoudre des problèmes complexes en temps polynomial. L'espace de recherche de l'optimiseur de requêtes s'élargit, en particulier lorsque les relations sont répliquées. Par conséquent, la stratégie de recherche doit être prise en compte pour faire face à large espace de recherche. Les algorithmes métaheuristiques, souvent non déterministes, surpassent les heuristiques traditionnelles en termes de performance. Alors que les heuristiques, basées sur une approche d'essais et erreurs, fournissent des solutions acceptables en un temps raisonnable, les métaheuristiques optimisent ce processus. Elles équilibrent efficacement les méthodes aléatoires et la recherche locale pour explorer l'espace de solutions de manière plus exhaustive et efficiente. Ces algorithmes peuvent être classés en deux catégories : ceux basés sur la population et ceux basés sur la trajectoire [56].

Dans ce qui suit, nous présentons certains travaux proposant des approches basées sur des métaheuristiques pour la résolution du problème de sélection de service IoT avec prise en compte des contraintes des QoS et l'énergie.

Dans cette recherche [57], les auteurs font une étude comparative d'un nombre d'approches de composition de services IoT sensibles à la QoS. Les méthodes et algorithmes utilisés dans ces approches comprennent des techniques telles que l'algorithme génétique (GA), l'optimisation par essaims de particules (PSO), la programmation linéaire (LP), l'algorithme K-means, l'optimisation lexicographique, l'algorithme de rétrogradation (backtracking) et le classificateur bayésien naïf. Les auteurs évaluent ces approches en fonction de critères tels que la sociabilité, l'optimalité, la prise en compte des paramètres multi-objectifs et les attributs de qualité de service tels que le coût, le temps de réponse, la fiabilité, la localisation géographique et la consommation d'énergie. En fin de compte, l'objectif est de fournir une vue d'ensemble des approches existantes et de déterminer les meilleures pratiques pour la composition de services IoT sensible aux QoS.

Dans cette étude [58], les auteurs proposent une approche visant à optimiser la com-

position des services mobiles dans le cloud en minimisant la consommation d'énergie des fournisseurs de cloud mobiles. Pour ce faire, un mécanisme de composition de services tenant compte de la consommation d'énergie est appliqué, utilisant un algorithme hybride de saut de grenouille (SF) combiné à un et d'un algorithme génétique (GA) (SFGA). Cette approche sélectionne de manière efficace les services atomiques afin d'atteindre une qualité de service optimale tout en minimisant la consommation d'énergie. L'objectif est d'améliorer la faisabilité de la composition des services avec une consommation d'énergie, un temps de réponse et un coût minimum pour les composants mobiles dans le cloud par rapport à certains algorithmes actuels. Les résultats des expériences ont montré que SFGA est excellent pour la minimisation de la consommation d'énergie des fournisseurs de cloud mobiles et l'optimisation de la composition des services mobiles en cloud pour atteindre une qualité de service optimale. Cependant, les limites ou les défis potentiels de l'implémentation de l'algorithme hybride SFGA n'ont pas été explicitement abordés dans l'article.

Dans ce papier [59], les chercheurs ont proposé un modèle de découverte et de sélection de sensible aux QoS dans un environnement IoT. Ce modèle repose sur un algorithme hybride métaheuristique, qui combine grey wolf optimizer (GWO) et un Algorithme Génétique (GA), appelé GWO-GA. L'objectif est de garantir des valeurs de QoS tels que le temps de réponse, la consommation d'énergie et les coûts lors de la découverte et de la sélection des services. Les résultats des expérimentations ont montré que l'algorithme GWO-GA est bon pour réduit le coût, améliore les performances par rapport aux autres algorithmes. L'approche pourrait inclure des limites potentielles, telles que des contraintes spécifiques non abordées dans la méthodologie ou des aspects non couverts dans les expérimentations.

Dans [60], les auteurs proposent une approche de composition de services efficace sur le plan énergétique et sensible à la qualité de service basée sur group teaching optimization n-based energy and QoS-aware services composition approach (GT-EQCA) pour traiter les problèmes d'énergie et de qualité de service dans les environnements de services IoT. Cette approche comporte trois phases : (i) La sélection des services IoT concrets qui ne sont pas en mesure de fournir le niveau de qualité de service requis par l'utilisateur, (ii) la sélection des k premiers services concrets de l'IoT en utilisant le principe de la dominance Pareto et (iii) la sélection des services concrets les plus efficaces en termes de QoS. La recherche du service composite avec une qualité de service sous-optimale (c'est-à-dire la composition qui satisfait aux contraintes globales de QoS et qui a la valeur d'utilité la plus élevée). Les résultats de l'expérimentation prouvent que l'approche GT-EQCA surpasse quatre algorithmes de base en termes de temps de composition, de consommation d'énergie et d'utilité. L'approche permet d'obtenir une composition efficace sur le plan énergétique et prennent en compte les QoS.

2.5.3 Comparaison entre les approches basées sur des métaheuristiques et les approches déterministes

Article : Cette colonne contient la référence de l'article. Cela inclut le titre, les auteurs, l'année de publication, et d'autres informations bibliographiques. Elle permet de situer le contexte de chaque étude, de faciliter la consultation des travaux originaux.

L'approche proposée : Cette colonne décrit la méthodologie ou la stratégie introduite par l'article pour résoudre un problème spécifique lié à la composition de services IoT. Cela peut inclure des algorithmes, des architectures de systèmes, des protocoles de communication ou des cadres théoriques. Cela permet également de comparer différentes méthodologies et solutions pour des problèmes similaires dans le domaine des IoT.

QoS : Cette colonne indique si l'approche prend en compte le (QoS).

Type d'approche : Cette colonne explique si le type de l'approche proposée est déterministe ou non déterministe.

Énergie : Cette colonne précise si la consommation d'énergie est prise en compte par l'approche.

Optimalité : L'optimalité désigne le fait de retourner une solution optimale ou sous optimale. Dans le contexte de l'optimisation, une solution est dite optimale si elle maximise ou minimise la fonction objective en respectant toutes les contraintes du problème (par exemple, minimisation de la latence, maximisation de la disponibilité).

Les avantages et les limites : résume les principaux avantages (aspects positifs ou points forts) et les principales limites (inconvenients ou aspects négatifs) de l'approche proposée dans l'article.

Article	Approche proposée	QoS	Enn eigné	Type d'ap-proche	Optimalité	Avantages/ Li-mites
[Boucetti et al (2022)]. [57].	Etude compa-rative entre les approches suivent : (GA), (PSO), (LP), K-means.	Oui	Oui	Nondéterministe	Quasi-optimale	+ Sensible au QoS et l'énergie.

[Godar et al (2022)]. [58]	SFGA	Oui	Oui	Non déterministe	Optimale	+ Minimisation de la consommation d'énergie. - Les limitations ou les défis potentiels de l'implémentation de l'algorithme hybride SFGA n'ont pas été explicitement abordés dans l'article.
[Ronghan et al (2022)]. [59].	GWO-GA	Oui	Oui	Non déterministe	Optimale	+ Une amélioration des performances par rapport aux autres algorithmes. - Les performances de GWO-GA peuvent être sensibles aux paramètres choisis.
[Hameche et al (2023)]. [60].	(GT-EQCA)	Oui	Oui	Non déterministe	Optimale	+ Obtenir une composition efficace sur le plan énergétique et sensible aux QoS.
[Bakera et al (2017)]. [54]	E2C2	Oui	Oui	Déterministe	Quasi-optimale	+ Réduction du nombre de services examinés. + Efficacité énergétique - Les défis de collecte et de gestion de données

[Urbietta et al (2017)]. [53]	wEASEL.	Oui	Non	Déterministe	Quasi-optimale	+ Permet de découvrir plus d'opportunités de composition - Prise en charge Limitée des exigences complexes des villes intelligentes.
[Zheng-yi et al (2021)]. [55]	FSCA-EQ	Oui	Oui	Déterministe	Optimale	+ Utilisation d'un mécanisme d'optimisation hiérarchique pour trouver des solutions optimales de Pareto. - Adaptabilité Limitée aux changements dynamiques des préférences et attributs.

TABLEAU 2.3 – Tableau récapitulatif des approches étudiées.

Discussion

La composition de services dans le domaine de l'IoT a suscité un intérêt croissant, mettant en lumière la sensibilité à la qualité de service comme critère central. Cette prise en compte des exigences des utilisateurs et de leurs préférences est essentielle pour garantir une expérience utilisateur optimale. Parallèlement, la plupart des articles examinés ont accordé une attention particulière à la consommation d'énergie, un critère crucial dans les environnements IoT.

Les méthodes d'optimisation les plus couramment utilisées, les approches non déterministes et, plus spécifiquement, les algorithmes métaheuristiques, ont été largement adoptées dans les travaux analysés, telles que [57], [58], [59], [60]. Ces métaheuristiques sont conçues pour explorer de manière efficace des espaces de recherche vastes, offrant ainsi un équilibre entre l'exploration et l'exploitation de l'espace de recherche. Leur objectif est de trouver des compositions de services quasi-optimales de qualité en temps raisonnable.

En revanche, certains travaux comme [53], [54], [55] ont privilégié des approches déterministes pour résoudre les problèmes de composition de services. Ces méthodes suivent une

procédure stricte et définie pour examiner de manière approfondie toutes les combinaisons possibles de services. Bien que cette approche puisse garantir une solution optimale. Elle reste inefficace en termes de temps de réponse lors du passage à l'échelle.

2.6 Conclusion

Dans ce chapitre, nous définissons la composition des services et son processus, qui comprend quatre étapes : définition du plan de composition, découverte des services, sélection des services, et exécution de la composition. Dans ce mémoire, nous traitons le problème de sélection de composition en prenant en compte des QoS et l'énergie et la préférence des utilisateurs. Nous avons défini les métaheuristiques en mettant et exploré quelques travaux traitant du problème de la composition des services IoT. Nous concluons ce chapitre par une comparaison et une synthèse sur les travaux étudiés.

Approche proposée et simulation

3.1 Introduction

La composition de services a été largement abordée dans le domaine de l'IoT durant ces dernières années. Elle permet de répondre à des requêtes complexes en combinant les fonctionnalités de plusieurs services individuels. Ce processus favorise la création de services composites réutilisables qui satisfont au mieux les exigences de QoS et d'énergie imposées par l'utilisateur. Dans ce chapitre, nous présentons notre approche nommée K-means with Genetic Algorithm (K-GA) qui combine k-means avec un algorithme génétique pour la résolution du problème de composition de services IoT sensibles aux QoS et à l'énergie. Notre objectif est de trouver une solution quasi optimale (une composition de services) qui répond au mieux à la requête de l'utilisateur. Dans notre approche, k-means est utilisé pour classifier les services IoT selon leurs attributs QoS et GA est utilisé pour sélectionner une solution quasi optimale de bonne qualité en un temps de réponse raisonnable.

3.2 Motivation

Dans le cadre de notre étude sur la composition de services dans l'IoT, notre objectif est de réduire l'espace de recherche et d'optimiser le temps d'exécution. Nous choisissons de classer les services IoT en fonction de leurs évaluations de qualité globale, puis de sélectionner ceux qui obtiennent les meilleures notes. En apprentissage automatique, diverses approches existent pour entraîner les modèles à partir des données, chacune adaptée à des types spécifiques de problèmes et de données, telles que l'apprentissage supervisé, l'apprentissage non supervisé, l'apprentissage semi-supervisé, l'apprentissage par renforcement, etc. Dans notre cas, nous nous appuyons sur l'approche non supervisée. Cette méthode nous permet d'explorer et d'analyser nos données sans avoir besoin d'étiquettes préalables. Après évaluation de plusieurs approches telles que Spectral Clustering, Clustering hiérarchique, K-means, nous avons opté pour l'utilisation de l'algorithme K-means pour notre clustering. Cette approche nous permet de résoudre efficacement le défi de la réduction de l'espace de recherche dans le

contexte de l'IoT. Son choix s'est justifié par sa simplicité de mise en œuvre et son efficacité dans la détection de clusters dans nos données. Il est rapide, évolutif pour des ensembles de données volumineux.

La combinaison de k-means avec l'algorithme génétique lors de la recherche d'une composition quasi optimale nous permet de garantir une excellente balance entre l'exploration et l'exploitation de l'espace de recherche. K-means est utilisé pour regrouper les données en clusters homogènes, facilitant ainsi une exploration structurée et efficace. Ensuite, l'algorithme génétique exploite ces clusters pour rechercher des solutions optimales. Cette combinaison entre k-means et GA permet de tester un grand nombre de compositions et de converger vers une solution de bonne qualité qui répond aux exigences de l'utilisateur en termes de QoS et d'énergie et de temps de réponse. Dans ce qui suit nous détaillons l'approche K-GA.

Le tableau suivant récapitule les résultats d'évaluation des algorithmes en fonction des valeurs Silhouette Score.

Algorithme	Spectral Clustering	Clustering hiérarchique	K-means
Valeur de Silhouette Score	0.07	0.19	0.33

3.3 K-means

Dans cette section, nous allons définir et expliquer le fonctionnement de l'algorithme de K-Means.

3.3.1 Définition

La méthode k-means est une technique largement utilisée dans l'apprentissage non supervisé pour le partitionnement de données non étiquetées en raison de sa facilité d'utilisation. Elle exploite la structure sous-jacente de la distribution des données et définit des règles pour regrouper les données présentant des caractéristiques similaires. Le processus résulte en la partition d'un ensemble de données selon des critères de clustering sans aucune connaissance préalable sur dataset. Dans un scénario de clustering idéal, chaque cluster est constitué d'instances de données similaires, qui sont assez dissemblables des instances dans d'autres clusters comme illustré dans la figure 3.3. Une telle mesure de dissimilarité dépend des données sous-jacentes et de l'objectif de l'algorithme [63].

K-means est souvent combiné avec l'apprentissage profond pour des tâches telles que la segmentation d'images et la reconnaissance d'écriture manuscrite. Bien que l'algorithme k-means fonctionne bien avec des clusters compacts et hyper-sphériques, il présente des limites, notamment en ce qui concerne l'attribution des centroïdes et le nombre de clusters, ainsi que la capacité à gérer différents types de donnée [63].

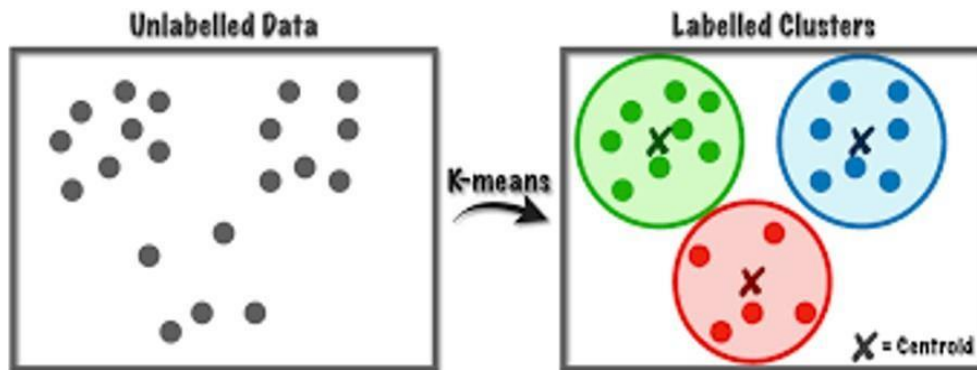


FIGURE 3.1 – Exemple d'application de k-means [64].

3.3.2 Le fonctionnement de l'algorithme de K-Means

Étape 1 : fixer le nombre de classes K .

Étape 2 : Sélectionnez des points K ou des centroïdes aléatoires. (Cela peut être différent de l'ensemble de données d'entrée).

Étape 3 : affecter chaque individu à la classe dont le centre est le plus proche.

Étape 4 : recalculer les centres de classes à chaque affectation ou à la fin d'une itération.

Ainsi, les étapes 3 et 4 se répètent donc jusqu'à ce que l'algorithme converge, c'est-à-dire jusqu'à ce que les objets ne changent plus de classe ou après un nombre fixé d'itérations [65].

3.3.3 Avantages de K-means

- Simple à mettre en œuvre.
- S'applique à des données de grandes tailles.
- Tend à réduire l'erreur quadratique.

3.3.4 Inconvénients

- Nombre de clusters à définir à l'avance.
- Les clusters sont construits par rapport à des objets inexistantes (les milieux)

3.4 Algorithme Génétique originale (GA)

3.4.1 Définition

L'idée des algorithmes génétiques a été proposée pour la première fois par John Holland dans les années 1960 et développée dans son livre "Adaptation in Natural and Artificial Systems" publié en 1975. Holland formalisé les concepts de sélection et d'évolution naturelle en les appliquant à des systèmes artificiels, créant ainsi une méthode pour résoudre des problèmes d'optimisation et de recherche [67].

Les algorithmes génétiques sont une classe d'algorithmes d'optimisation et de recherche inspirés par les principes de la sélection naturelle et de la génétique de Darwin. Ils sont utilisés pour trouver des solutions approximatives à des problèmes complexes en évoluant une population de solutions potentielles au fil des générations. Chaque solution est représentée sous forme de chromosomes, et les opérations de sélection, croisement (crossover), et mutation sont appliquées pour générer de nouvelles solutions [68]. Ils cherchent la solution globale d'une manière aléatoire. Ces algorithmes sont les plus connus et utilisés des algorithmes évolutionnaires. Pour mieux comprendre la similitude entre le processus de sélection naturelle et l'algorithme d'optimisation que nous allons proposer, rappelons tous les plus importants qui sont à la base de l'évolution dans GA [68].

1. L'évolution se produit sur des chromosomes qui représentent chacun des individus dans une population.
2. Le processus de sélection naturelle fait en sorte que les chromosomes les mieux adaptés se reproduisent plus souvent et contribuent davantage aux populations futures.
3. Lors de la reproduction, l'information contenue dans les chromosomes des parents est combinée et mélangée pour produire les chromosomes des enfants (« croisement »).
4. Le résultat du croisement peut à son tour être modifié par des perturbations aléatoires (mutations).

3.4.2 Principe de fonctionnement

GA est défini par un certain nombre d'étapes qui sont : initialisation, évolution, sélection, croisement, mutation, nouvelle génération.

Initialisation : Une population initiale de N chromosomes (individus) est générée de manière aléatoire dans cette étape.

Évaluation : Évaluer chacun des chromosomes de la population à l'aide d'une fonction de fitness qui mesure la qualité de la solution.

Sélection : Reproduire les chromosomes en sélectionnant les parents les plus aptes (ceux avec la meilleure valeur de fitness).

Croisement : Des paires chromosomes sont croisées pour produire une nouvelle génération des chromosomes (enfants), combinant des caractéristiques des parents.

Mutation : Les enfants peuvent subir des mutations avec une certaine probabilité pour introduire de la diversité génétique.

Nouvelle génération : Lorsque N nouveaux individus ont été générés, ils remplacent alors l'ancienne population. Les chromosomes de la nouvelle population sont évalués à leur tour. Et le processus est répété jusqu'à ce qu'un critère d'arrêt soit atteint (comme un nombre fixe de générations ou une solution satisfaisante trouvée).

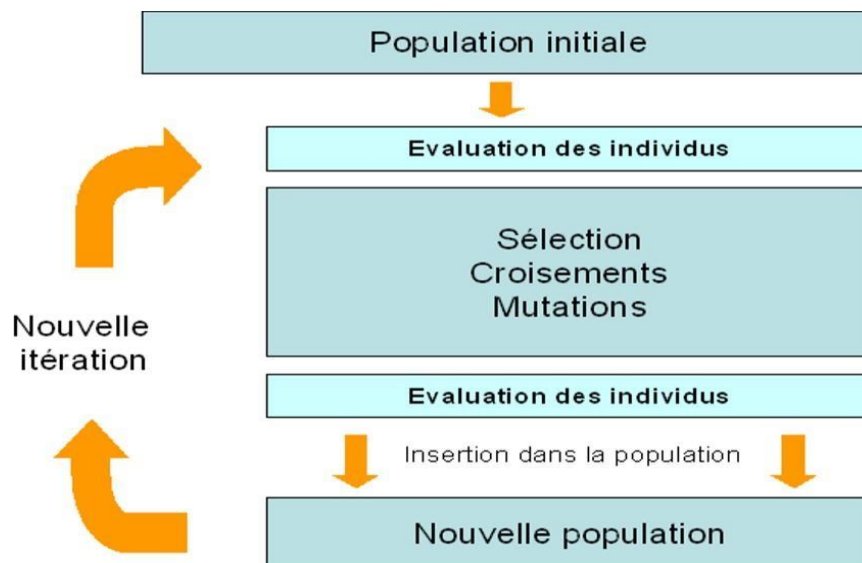


FIGURE 3.2 – Schéma général d'un algorithme génétique [69].

3.5 Le fonctionnement de l'approche proposée K-GA

K-GA est une combinaison d'un modèle de k-means et de la métaheuristique (GA) pour le traitement du problème de composition de service IoT sensible aux QoS et la consommation d'énergie. K-means est utilisé dans K-GA pour classer les services et GA est utilisé pour rechercher une bonne solution quasi optimale qui satisfait les besoins de l'utilisateur en termes de QoS et l'énergie. Dans cette section, nous expliquons la mise en œuvre de notre approche. Nous présentons les deux étapes de K-GA à savoir la classification et la sélection des services IoT.

3.5.1 Tableau des variables de K-GA

Abréviation	Signification
nAS	Le nombre de services abstraits dans la composition
nSC	Nombre de services candidat
I_{max}	Le nombre maximal d'itérations sans amélioration de la fitness
G_{max}	Le nombre maximal de générations
nComp	Nombre de compositions
$P_{selected}$	Le chromosome qui a été sélectionné
ARGmin	L'index de l'élément ayant la valeur minimale

TABLEAU 3.1 – Tableau des variables de k-GA.

3.5.2 Dataset

Pour la validation de notre approche, nous avons utilisé IoTS_Dataset « QoS data IoT services », qui ont été effectuées depuis l'année 2023 [62].

Description de dataset

Ce dataset contient des valeurs de QoS de services IoT généré par un algorithme aléatoire. Les attributs de QoS pris en compte dans IoTS_Dataset sont : temps d'exécution, coût, réputation, fiabilité. Le dataset est devise en sous ensemble à savoir IoTS10X50, IoTS10X100, IoTS20X50, IoTS20X100, IoTS30X50 et IoTS30X100, pour faire variés les CS et les AS.

IoTS10X50 : 10 fichiers de données Excel représentant 10 tâches, chaque tâche étant équivalente à un service IoT abstrait (AS), et chaque tâche comportant 50 services IoT candidats (CS), (c'est-à-dire chaque service IoT abstrait), c'est-à-dire 50 services IoT fonctionnellement identiques ou similaires mais non fonctionnels (QoS).

IoTS10X100 : pour ce cas on retrouve 10 AS et 100 CS.

IoTS20X50 : pour ce cas on retrouve 20 AS et 50 CS.

IoTS20X100 : pour ce cas on retrouve 20 AS et 100 CS.

IoTS30X50 : pour ce cas on retrouve 30 AS et 50 CS.

IoTS30X100 : pour ce cas on retrouve 30 AS et 100 CS.

Pour tester notre approche, nous avons considéré les 9000 services sous la forme d'un seul ensemble. Chaque ligne de ce dataset représente un service candidat et ses quatre mesures de QoS correspondantes comme illustré dans la Figure 3.3.

Dataset avant l'ajout de l'énergie:

	IoTS Number	Execution Time	Service Cost	Credibility	Reliability
0	1	33	73	5	0.17
1	2	60	8	10	0.36
2	3	25	18	8	0.26
3	4	4	35	9	0.45
4	5	25	24	4	0.67
...
8995	8996	17	97	5	0.29
8996	8997	8	94	4	0.94
8997	8998	24	50	4	0.73
8998	8999	40	18	8	0.11
8999	9000	14	89	10	0.87

9000 rows × 5 columns

FIGURE 3.3 – IoTS_Dataset.

Dans notre cas, il est nécessaire d'ajouter une colonne "Énergie" qui définit la consommation d'énergie de chaque service. Cette consommation d'énergie est calculée en fonction des attributs de qualité de service (temps d'exécution, coût, réputation, fiabilité). Cela permet d'inclure une dimension supplémentaire importante pour évaluer l'efficacité énergétique des services IoT dans le processus de classification comme illustré dans la Figure 3.4.

Dataset après l'ajout de l'énergie:

	IoTS Number	Execution Time	Service Cost	Credibility	Reliability	énergie
0	1	33	73	5	0.17	36.117
1	2	60	8	10	0.36	28.436
2	3	25	18	8	0.26	17.026
3	4	4	35	9	0.45	13.945
4	5	25	24	4	0.67	18.067
...
8995	8996	17	97	5	0.29	36.929
8996	8997	8	94	4	0.94	32.294
8997	8998	24	50	4	0.73	25.473
8998	8999	40	18	8	0.11	23.011
8999	9000	14	89	10	0.87	34.387

9000 rows × 6 columns

FIGURE 3.4 – Dataset après l'ajout de l'énergie.

3.5.3 Classification et sélection des services candidats via le K-means

Nous avons utilisé la méthode de classification K-means pour évaluer chaque service IoT en fonction de son vecteur d'attributs QoS pour l'affecter à une catégorie de niveau de QoS. Pour la mise en œuvre de l'algorithme K-means, nous avons utilisé le dataset "IoTS_Dataset". Pour la phase de classification avec K-means, nous avons procédé comme suit :

- **Prétraitement des données :** Dans cette étape, nous ajoutons une colonne "énergie" à notre dataset et supprimons la colonne "ID". Ces opérations préliminaires sont essentielles pour assurer la qualité et la pertinence des résultats de notre analyse.
- **La normalisation des valeurs des QoS de dataset :** Avant de procéder à la classification, Il est nécessaire de normaliser leurs valeurs de QoS de manière à les ramener dans l'intervalle [0,1], pour garantir que chaque attribut contribue de manière égale à la distance calculée par l'algorithme K-means. Les valeurs des attributs positifs doivent être maximisées (réputation, fiabilité), tandis que les valeurs des attributs négatifs doivent être minimisées (coût, temps d'exécution, Énergie). Dans la phase de normalisation, les attributs de QoS positifs et négatifs sont mis à l'échelle selon les formules suivantes [70] :

Positive attributs :

$$MQS (s, q, c) = \frac{MQS (s, q, c) - Q_{min}(q)}{Q_{max}(q) - Q_{min}(q)} \quad (3.1)$$

Negative attributes:

$$MQS(s, q, c) = \frac{Q_{max}(q) - MQS(s, q, c)}{Q_{max}(q) - Q_{min}(q)} \quad (3.2)$$

Où $Q_{max}(q)$, $Q_{min}(q)$ sont les valeurs maximale et minimale du $q^{ème}$ attribut et $MQS(s, q, c)$ est la valeur du $q^{ème}$ attribut pour un service candidat sélectionné s dans la classe c . [70].

- **Détermination du nombre de clusters (K) :** Le nombre de clusters a été déterminé en utilisant la méthode de silhouette pour optimiser le choix de K . Nous avons obtenu $K=4$: bronze, argent, or ou platine.
- **Classification des services :** Les services ont été classés en fonction des clusters formés par K -means. Chaque service a été affecté au cluster dont il est le plus proche en termes de distance euclidienne.
- **Interprétation des clusters :** Les clusters obtenus ont été interprétés pour définir les catégories de niveaux de QoS (bronze, argent, or, platine). Chaque cluster représente une catégorie de services en fonction des QoS.

La sélection des cluster (SC) après l'application de K-means

Une fois la classification effectuée, nous avons choisi de conserver les deux meilleures catégories de services IoT, à savoir "Platine" et "Or". Par la suite, l'espace de recherche a été réduit à un nombre de 4500 services. Après avoir classé et sélectionné les services du dataset "IoTS_Dataset", nous procédons à leur extraction et les répartissons ensuite parmi les services abstraits du plan de composition. Il est important de souligner que chaque service abstrait (AS) recevra le même nombre de services candidats, noté nbSC. Cette répartition équilibrée garantit que chaque service abstrait dispose de services candidats de haute qualité pour optimiser la composition globale.

L'algorithme 1 représente les étapes de K-means utilisé pour la classification des services.

Algorithme 1 : Algorithme de K-means

1. **Entrées** : IoTS_Dataset
2. **Sorties** : IoTS_Dataset normalisé, classé et réparti sur les AS
3. **DEBUT**
 1. Lire le dataset IoTS_Dataset ;
 2. Ajouter la colonne Energy ;
 3. Normalisation des valeurs du IoTS_Dataset ;
 4. Faire appel à la fonction K-means ;
 5. Utiliser K-means pour classer les services du IoTS_Dataset ;
 6. 6. Sélectionner les services qui font partie des deux meilleures catégories "platine" et "or" du IoTS_Dataset ;
 7. Diviser les services sélectionnés à l'aide de la fonction diviser_dataset.

Fin

3.5.4 Application de GA

Dans cette section, nous présentons notre modélisation du problème de composition de services IoT sensibles aux critères de QoS et à l'énergie. Une composition abstraite, ou un plan de composition, notée SCA, est une séquence de n_{AS} services abstraits (AS) et est représentée comme suit : $SCA = \{AS_1, AS_2, \dots, AS_{n_{AS}}\}$. Chaque AS contient un ensemble de n_{SC} services candidats fonctionnellement équivalents : $AS = \{SC_1, SC_2, \dots, SC_{n_{SC}}\}$. Nous représentons une composition par un vecteur de n_{AS} nombre entier comme suit : $Comp = \{X_1, X_2, \dots, X_{n_{AS}}\}$ où $X_1, X_2, \dots, X_{n_{AS}}$ représentent les indices des services candidats appartenant à la composition. Dans la suite de cette section, nous détaillerons les étapes de l'application de GA.

1. Générer aléatoirement les compositions de la population initiale

Dans cette étape, on génère aléatoirement des compositions à travers l'espace de recherche. Les valeurs générées ne doivent pas dépasser le nombre de service candidat dans les AS (c'est-à-dire les x_i de chaque composition ont des valeurs entre 1 et nSC).

Voilà le code et un exemple de population générée :

Créer la population initiale avec des compositions basées sur les AS

```
initial_population =
[create_composition(as1_
reduced, as2_reduced,
as3_reduced, as4_reduced,
as5_reduced) for _ in
range(30)]
print ("
nPopulation initiale :")
for ind in
initial_population :
print(ind)
```

```
Population initiale:
[63, 987, 1835, 2803, 3767]
[111, 916, 1925, 2852, 3691]
[55, 1030, 1865, 2773, 3688]
[107, 971, 1870, 2841, 3706]
[143, 986, 1914, 2784, 3751]
[135, 1037, 1884, 2781, 3795]
[90, 916, 1891, 2752, 3753]
[61, 1036, 1836, 2842, 3793]
[59, 1042, 1968, 2744, 3766]
[78, 1059, 1964, 2750, 3655]
[106, 1041, 1866, 2822, 3670]
[127, 929, 1959, 2762, 3789]
[71, 996, 1882, 2751, 3715]
[110, 1037, 1952, 2755, 3777]
[125, 1004, 1888, 2777, 3744]
[145, 1000, 1848, 2872, 3708]
[105, 970, 1933, 2771, 3697]
[47, 1035, 1866, 2813, 3700]
[97, 1002, 1953, 2807, 3778]
[30, 946, 1951, 2752, 3659]
[69, 1023, 1902, 2747, 3686]
[44, 922, 1943, 2758, 3796]
[1, 914, 1932, 2834, 3682]
```

FIGURE 3.5 – Population initiale.

2. Évaluation de la population initiale

Une fois la population initiale générée, nous calculons la fitness de chaque composition en utilisant la fonction de fitness décrite dans l'équation (2.1) du chapitre 2. Rappelons que la fonction de fitness utilisée est la suivante :

$$f(s) = \sum_{q=1}^k w_q \cdot qos'_q(s)$$

La composition avec la meilleure valeur de fitness est alors enregistrée et sauvegardée jusqu'à la fin de l'exécution de K-GA. Étant donné que notre objectif est de minimiser la valeur de fitness, c'est la composition ayant la valeur de fitness la plus faible qui sera sélectionnée. Voilà le code et un exemple de l'affichage.

Calculer les valeurs de fitness pour chaque composition.

```
fitness_values = [
fitness (ind, df,
weights, scaler)
for ind in initial
_population]
print("
nValeurs de fitness
initiales :")for i,
fit in enumerate
(fitness_values) :
print (f"Composition
{i + 1} : Fitness =
{fit}")
```

```
Valeurs de fitness initiales:
Composition 1: Fitness = 0.630205826340696
Composition 2: Fitness = 0.17921032832288056
Composition 3: Fitness = 0.48072795607222696
Composition 4: Fitness = 0.5416136329285152
Composition 5: Fitness = 0.5661440419291677
Composition 6: Fitness = 0.43792837903643517
Composition 7: Fitness = 0.5293238080124725
Composition 8: Fitness = 0.596989498119922
Composition 9: Fitness = 0.6229627154550765
Composition 10: Fitness = 0.8853861856670047
Composition 11: Fitness = 0.5793443909308571
Composition 12: Fitness = 0.568414143559117
Composition 13: Fitness = 0.6227751796646949
Composition 14: Fitness = 0.4158180317184961
Composition 15: Fitness = 0.7440472840841604
Composition 16: Fitness = 0.3230155105399145
Composition 17: Fitness = 0.7592910752317306
Composition 18: Fitness = 0.41393147212415526
Composition 19: Fitness = 0.6605005777411014
Composition 20: Fitness = 0.5177073942743711
Composition 21: Fitness = 0.7556927024530236
Composition 22: Fitness = 0.5761667340802343
Composition 23: Fitness = 0.42577206813451596
```

FIGURE 3.6 – valeurs de fitness initiales.

3. Appliquer les opérations de GA à la population initiale

Pour chaque composition de la population, on applique les opérations de GA à la population initiale pour mettre à jour les compositions.

Sélection

La sélection consiste à choisir les compositions parentales pour la reproduction. Nous utilisons la sélection par tournoi, où plusieurs sous-ensembles de compositions sont sélectionnés aléatoirement et la meilleure composition de chaque sous-ensemble est choisie comme parent.

La sélection par tournoi se déroule comme suit :

1. Sélectionner aléatoirement de $nComp$ compositions de la population.
2. Choisir la composition avec la meilleure fitness parmi ces $nComp$ compositions.

$$P_{(selected)} = ARGmin_{fitness}(p_i) \quad (3.3)$$

Où

- $i \in nComp$.
- ARGmin : l'index de l'élément ayant la valeur minimale.

Croisement (Crossover)

Le croisement combine deux compositions parentales pour produire une ou plusieurs compositions enfants. Nous utilisons ici un croisement à un point (one-point crossover) Comme le montre la Figure 3.7. Le processus est décrit par l'équation suivante :

$$\begin{aligned} \text{Enfant1} &= (\text{Parent1}[1p], \text{Parent2}[p + 1nAS]) \\ \text{Enfant2} &= (\text{Parent2}[1p], \text{Parent1}[p + 1nAS]) \end{aligned} \quad (3.4)$$

Où

- p est le point de croisement choisi aléatoirement.
- nAS est le nombre de services abstraits dans la composition.

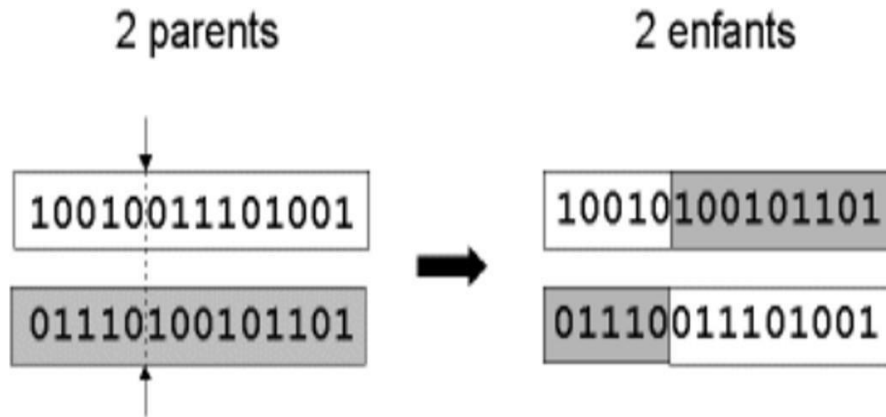


FIGURE 3.7 – Croisement avec un point de crossover [68].

Mutation

La mutation introduit de la diversité en modifiant aléatoirement certaines parties des compositions des enfants. Nous appliquons une mutation de substitution, où un service candidat est remplacé par un autre service candidat choisi aléatoirement. Comme le montre la Figure 3.8. La mutation est définie par :

$$\begin{aligned}
 \text{Enfant}[i] = & \text{Service Candidat}_{\text{alatoire}} \\
 & \begin{cases} \text{new_value}(g_i) & \text{si } r < P_m \\ g_i & \text{sinon} \end{cases} \quad (3.5)
 \end{aligned}$$

Où :

- g_i est le gène après mutation.
- r est un nombre aléatoire uniformément distribué entre 0 et 1.
- P_m est la probabilité de mutation (taux de mutation).
- $\text{new_value}(g_i)$ est une nouvelle valeur aléatoire pour le gène g_i .



FIGURE 3.8 – Mutation aléatoire [78].

Évaluation de la fitness

Après les opérations de croisement et de mutation, la fitness des nouvelles compositions est calculée en utilisant la même fonction d'utilité que précédemment (2.1) :

Sélection de la nouvelle génération

La nouvelle génération est formée en sélectionnant les meilleures compositions parmi les parents et les enfants. Nous utilisons une stratégie d'élitisme pour assurer que les meilleures compositions sont sauvegardées. Cette stratégie peut être formulée comme suit :

- Combiner les parents et les enfants pour former une nouvelle population étendue.
- Trier les compositions en fonction de leur fitness en utilisant fonction (2.1).
- Sélectionner les N meilleures compositions pour former la nouvelle génération, où N est la taille de la population initiale.

Boucle Génétique et critères d'arrêt

Le processus de sélection, croisement, mutation et évaluation de la fitness se répète jusqu'à ce qu'un critère d'arrêt soit atteint. Les critères d'arrêt peuvent inclure :

- **Nombre maximal de générations** : Le processus s'arrête après un nombre prédéterminé de générations, G_{max} .
- **Stagnation de la fitness** : Le processus s'arrête si la meilleure valeur de fitness ne s'améliore pas après un certain nombre d'itérations consécutives, I_{max} .

Equation de la Boucle Génétique avec Critères d'Arrêt

$$\text{Population}(t+1) = \text{Sélection}(\text{Croisement}(\text{Mutation}(\text{Population}(t))))$$

Le processus s'arrête lorsque l'un des critères suivants est atteint :

- $t \geq G_{max}$
- $\Delta \text{Fitness}_{best} = 0$ pour $t_{stagnation} \geq I_{max}$

Où

- t représente l'itération courante.
- G max est le nombre maximal de générations.
- I max est le nombre maximal d'itérations sans amélioration de la fitness.

Retourner la meilleure solution (meilleure composition)

À la fin de l'exécution de K-GA, l'algorithme retourne la meilleure composition parmi toutes celles générées au cours des itérations. Cela est rendu possible car, à la fin de chaque itération, la meilleure composition trouvée est enregistrée dans un ensemble. La meilleure composition finale est déterminée en comparant les valeurs de fitness des compositions sauvegardées. En suivant ces étapes, l'algorithme GA explore l'espace de recherche en générant et en évaluant de nouvelles compositions à chaque itération. Au fil des générations, il tend à converger vers une solution optimale ou proche de l'optimum. L'algorithme 2 présente les étapes suivies par GA.

Algorithme 2 : Etapes de recherche de composition dans K-GA

1. **Entrées** : : nAS , nSC , l_{max} , G_{max} , $nComp$, p_c , p_m ,
 $P_{selected}$, $ARGmin$,
2. **Sorties** : Meilleure composition et fitness,
3. **DEBUT**
 - 1 : Générer une population initiale P aléatoire de N composition ;
 - 2 : Evaluation de chaque composition de la population P avec la fonction de fitness 2.1 ;
 - 3 : Tantque ($t \leq G_{max}$ ou $t_{stagnation} \leq l_{max}$)
 - 4 : **Pour chaque composition faire** :
 - 5 : Sélectionner deux parents $P1$ et $P2$ avec la méthode de sélection de tournoi ;
 - 6 : Appliquer les opérateurs génétiques ;
 - 6.1 : Opérateur de croisement avec une probabilité p_c
 - 6.2 : Opérateur de mutation avec une probabilité p_m .
 - 7 : Ajouter les nouvelles compositions à la nouvelle population ;
 - 8 : Evaluation de ce nouvel ensemble des compositions,
 - 9 : Classement des compositions (nouvelles et anciennes).


```
10 : Evaluation de ce nouvel ensemble des chromosomes
      avec la fonction fitness 2.1,
11 : Classement des compositions (nouvelles et
      anciennes) en fonction de la fitness.
12 : MAJ la population selon la valeur de finesse ;
13 : MAJ la meilleure solution.
14 : Si il y a amélioration incrémenter
      Accept.
15 : Sinon faire Accept=0 et aller à l'étape4.
16 : Fin si
17 : FinPour
18 : Sauvegarder la meilleur fitness et sa
      composition ;
19 : t=t + 1.
20 : Fin tant que
21 : Retourner la meilleure composition et sa
      valeur de fitness.
```

END

3.6 Scénario d'application de K-GA (Étude de cas)

Dans un monde où le vieillissement de la population et les maladies chroniques nécessitent une surveillance continue, les solutions de santé à domicile gagnent en importance. Nous avons choisi comme scénario pour valider notre approche la gestion de la santé à domicile dans une maison intelligente. Cette maison est équipée de divers capteurs de santé et d'appareils connectés pour assurer un suivi régulier de l'état de santé des résidents. L'objectif est de surveiller les signes vitaux et les activités des résidents, de détecter toute anomalie, et d'alerter les professionnels de santé et les proches en cas de besoin. Pour y parvenir, il faudra combiner cinq services IoT comme illustré dans la figure 4.1. Ces services sont essentiels pour garantir le bon fonctionnement de notre système de gestion de la santé à domicile. Chacun d'eux a une fonctionnalité spécifique, allant de la collecte des données de santé à la notification des intervenants en cas de problème. Cette coordination efficace assure la tranquillité

d'esprit des résidents et de leurs familles. Nous présentons dans ce qui suit la description des services abstraits de notre plan de composition ainsi que le rôle de chacun d'entre eux.

Service de Surveillance des Activités (SSA)

Ce service utilise des capteurs de mouvement et des caméras pour surveiller les activités quotidiennes des résidents. Il peut identifier les habitudes normales et détecter les anomalies, comme une chute ou une période d'inactivité prolongée.

Service de Collecte de Données de Santé (SCDS)

Ce service gère les capteurs de santé installés dans la maison (tels que les capteurs de fréquence cardiaque, de pression artérielle, de glycémie, etc.). Il collecte régulièrement les données de santé des résidents et les enregistre pour une analyse ultérieure.

Service d'analyse de données de santé (SADS)

Ce service prend les données collectées par le SCDS et analyse en temps réel les signes vitaux des résidents. Il utilise des algorithmes de machine learning pour détecter toute anomalie ou changement significatif dans les paramètres de santé.

Service d'alerte en cas d'anomalie (SACA)

Lorsque le SADS ou le SSA détecte une anomalie (par exemple, une chute, une irrégularité cardiaque, ou une absence de mouvement), ce service envoie une alerte aux professionnels de santé et aux proches des résidents. L'alerte inclut les données pertinentes pour permettre une évaluation rapide de la situation.

Service de consultation à distance (SCD)

Ce service permet aux résidents de consulter à distance un professionnel de santé en cas de besoin. Il fournit une plateforme de visioconférence sécurisée, intégrée aux autres services, pour faciliter les consultations médicales sans que les résidents aient à quitter leur domicile.

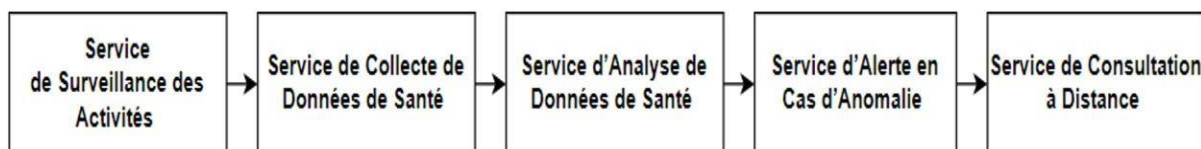


FIGURE 3.9 – Plan de composition associé au scénario de K-GA.

Ces services, lorsqu'ils sont correctement orchestrés, assurent un suivi complet et personnalisé de la santé des résidents, tout en leur permettant de vivre de manière autonome et en toute sécurité.

Rappelons que chaque service abstrait du plan de composition défini dans la figure 3.9 peut être implémenté par plusieurs services IoT candidats possédant des fonctionnalités similaires, mais présentant des valeurs de qualité de service (QoS) différentes. Le tableau 3.2 Présente, pour chaque service abstrait du scénario, trois exemples de services IoT réels pouvant être utilisés. Cela donne lieu à 35 compositions possibles de services IoT. L'objectif de l'approche K-GA proposée est de sélectionner la meilleure combinaison de services répondant aux exigences de l'utilisateur.

Service abstrait	Service candidat	Fournisseur
Service de surveillance des activités	CS^1_1 CS^1_2 CS^1_3	Nest Cam Ring Video Doorbell Amazon Echo Show
Service de collecte de données de santé	CS^2_1 CS^2_2 CS^2_3	Apple HealthKit Fitbit API Withings Health Mate
Service d'analyse de données de santé	CS^3_1 CS^3_2 CS^3_3	IBM Watson Health Google Cloud Healthcare API Microsoft Azure Health Bot
Service d'alerte en cas d'anomalie	CS^4_1 CS^4_2 CS^4_3	Twilio IFTTT Bosch IoT Suite
Service de consultation à distance	CS^5_1 CS^5_2 CS^5_3	Teladoc Health Zoom for Healthcare Amwell

TABLEAU 3.2 – Services candidats disponibles pour chaque service abstrait dans le scénario de composition.

3.7 Simulation

3.7.1 Environnement, outils et bibliothèques utilisés

L'implémentation et l'évaluation de l'approche proposée (K-GA) ont été réalisées sur un ordinateur portable sous Windows 11 Professionnel caractérisé par un Processeur Intel(R) Core (TM) i5-8350U CPU @ 1.70GHz 1.90 GHz équipé d'une RAM de 8,00 Go.

Python

Python est le langage de programmation informatique le plus populaire et le plus utilisé, notamment dans le domaine de la Data Science et du Machine Learning. De plus, Python est un langage multiplateforme qui fonctionne sur divers systèmes d'exploitation, tels que Windows, macOS et Linux, ce qui en fait un choix idéal pour les développeurs travaillant sur différents environnements [71].

Le choix du langage Python s'explique par sa syntaxe simplifiée et intuitive qui facilite tant l'apprentissage pour les débutants que la productivité pour les développeurs aguerris. Sa force réside également dans son vaste écosystème de bibliothèques spécialisées, rendant Python particulièrement efficace pour des domaines tels que l'analyse de données, le machine learning et le développement web. Simple d'utilisation [71].

JupyterLab

C'est le dernier environnement de développement interactif basé sur le Web pour les blocs-notes, le code et les données. Son interface flexible permet aux utilisateurs de configurer et d'organiser des flux de travail en science des données, en informatique scientifique, en journalisme informatique et en apprentissage automatique. Une conception modulaire invite les extensions à étendre et enrichir les fonctionnalités [72].

Jupyter Notebook

Jupyter Notebook est l'application Web originale pour créer et partager des documents informatiques. Il offre une expérience simple, rationalisée et centrée sur les documents [72].

Pandas

La bibliothèque logicielle open-source Pandas est spécifiquement conçue pour la manipulation et l'analyse de données en langage Python. Elle est à la fois performante, flexible et simple d'utilisation. Grâce à Pandas, le langage Python permet enfin de charger, d'aligner, de manipuler ou encore de fusionner des données. Les performances sont particulièrement impressionnantes quand le code source back-end est écrit en C ou en Python [73].

Scikit-Learn

C'est une librairie Python qui donne accès à des versions efficaces d'un grand nombre d'algorithmes courants. Elle offre également une API propre et uniformisée. Par conséquent, un des gros avantages de Scikit-Learn est qu'une fois que vous avez compris l'utilisation et

la syntaxe de base de Scikit-Learn pour un type de modèle, le passage à un nouveau modèle ou algorithme est très simple [74].

Matplotlib

Matplotlib est une bibliothèque Python open source, initialement développée par le neurobiologiste John Hunter en 2002. L'objectif était de visualiser les données. Pour y parvenir, il souhaitait répliquer les fonctionnalités de création graphique de MATLAB avec Python. Il est par exemple possible de créer des tracés, des histogrammes, des diagrammes à barre et tous types de graphiques à l'aide de quelques lignes de code. Il s'agit d'un outil très complet, permettant de générer des visualisations de données très détaillées [75].

NumPy

NumPy est une bibliothèque open source en langage Python, très populaire qui est principalement utilisée pour effectuer des calculs mathématiques et scientifiques sur des tableaux et des matrices. Cet outil permet d'effectuer ces opérations bien plus rapidement et efficacement que les listes Python. Elle offre de nombreuses fonctionnalités et outils qui peuvent être utiles pour les projets de Data Science. Se familiariser avec NumPy est une étape indispensable dans un projet de formation en Data Science [76].

3.7.2 Pré-traitement et normalisation des données

Le pré-traitement des données est une étape déjà expliquée, comme mentionné dans la section [3.5.3]. Le pré-traitement de notre dataset consiste à ajouter une colonne 'énergie' et à normaliser les valeurs pour pouvoir appliquer l'algorithme K-means. Pour cela, nous utilisons la fonction MinMaxScaler. Les résultats de la normalisation sont affichés dans la figure suivante.

```
# Normaliser les valeurs des QoS et de l'énergie entre 0 et 1
scaler = MinMaxScaler()
df[['Execution Time', 'Service Cost', 'Credibility', 'Reliability', 'Energy']]
| scaler.fit_transform(df[['Execution Time', 'Service Cost', 'Credibility', 'Reliability', 'Energy']])
```

FIGURE 3.10 – Fonction de normalisation.

Dataset après la normalisation :

	Execution Time	Service Cost	Credibility	Reliability	Énergie
0	0.542373	0.727273	0.285714	0.076923	0.632038
1	1.000000	0.070707	1.000000	0.285714	0.490382
2	0.406780	0.171717	0.714286	0.175824	0.279955
3	0.050847	0.343434	0.857143	0.384615	0.223134
4	0.406780	0.232323	0.142857	0.626374	0.299153
...
8995	0.271186	0.969697	0.285714	0.208791	0.647013
8996	0.118644	0.939394	0.142857	0.923077	0.561533
8997	0.389831	0.494949	0.142857	0.692308	0.435738
8998	0.661017	0.171717	0.714286	0.010989	0.390333
8999	0.220339	0.888889	1.000000	0.846154	0.600133

9000 rows × 5 columns

FIGURE 3.11 – Dataset après la normalisation.

La Figure 3.12 présente les données normalisées avant classification

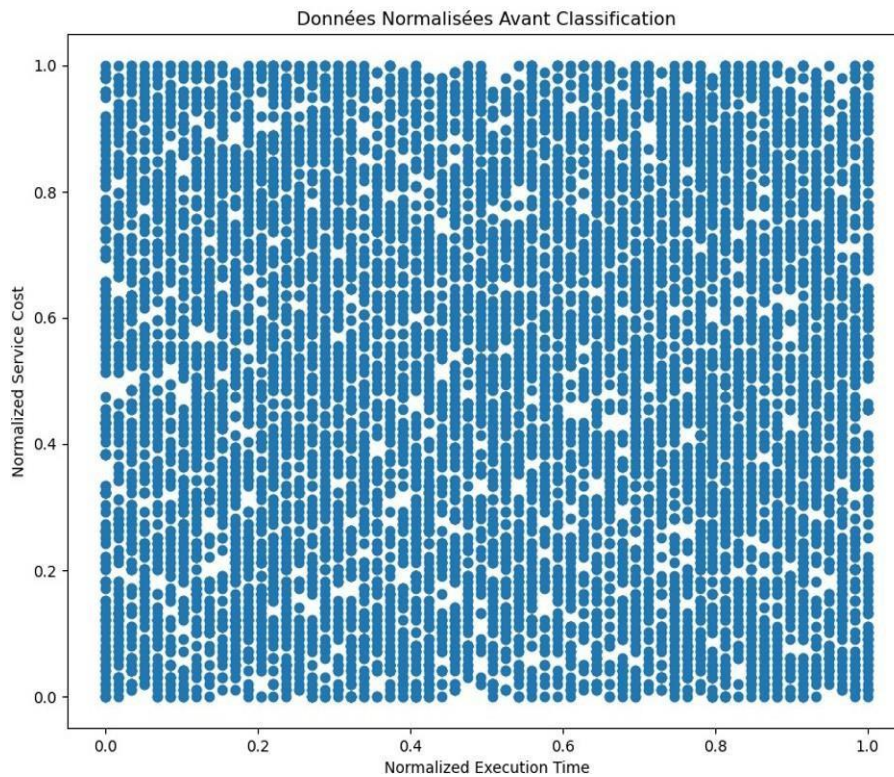


FIGURE 3.12 – Données normalisées avant classification.

3.7.3 Classification via K-means

Dans cette section, nous présentons les résultats de l'application de l'algorithme K-means pour la classification des services IoT.

```
# Utiliser KMeans pour classifier les services en 4 catégories  
kmeans = KMeans(n_clusters=4, random_state=42)  
df['Cluster'] = kmeans.fit_predict(scaled_features)
```

FIGURE 3.13 – Classification avec K-means.

La figure 3.14 présente les valeurs QoS après la classification pour évaluer l'efficacité de notre approche.

1. Platine (Haute qualité)
2. Or
3. Argent
4. Bronze (Basse qualité)

Dataset après la classification :

	IoT5 Number	Execution Time	Service Cost	Credibility	Reliability	Énergie	Category
0	1	33	73	5	0.17	36.117	4
1	2	60	8	10	0.36	28.436	3
2	3	25	18	8	0.26	17.026	3
3	4	4	35	9	0.45	13.945	3
4	5	25	24	4	0.67	18.067	1
...
8995	8996	17	97	5	0.29	36.929	2
8996	8997	8	94	4	0.94	32.294	1
8997	8998	24	50	4	0.73	25.473	1
8998	8999	40	18	8	0.11	23.011	3
8999	9000	14	89	10	0.87	34.387	2

9000 rows × 7 columns

FIGURE 3.14 – Dataset après la classification.

La Figure 3.15 présente la classification des services après l'utilisation de K-means.

- La couleur jaune représente les services de la classe Platine.
- La couleur verte représente les services de la classe Or.
- La couleur Bleu clair représente les services de la classe Argent.
- La couleur Bleu foncé représente les services de la classe Bronze.

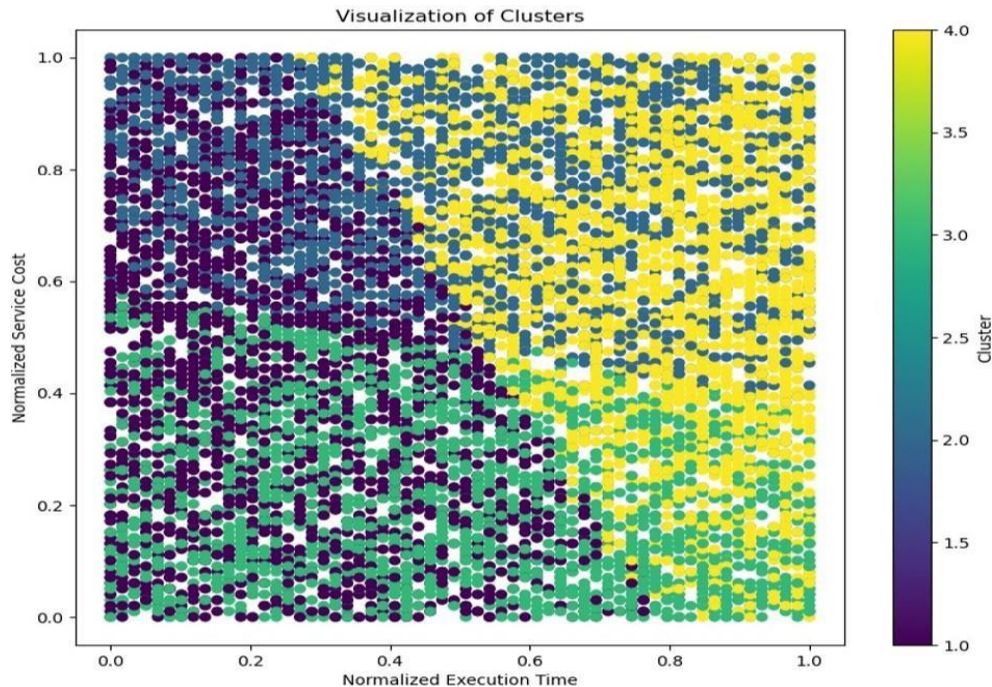


FIGURE 3.15 – Classification des services après l'utilisation de K-means.

Silhouette Score pour différents nombres de clusters

Pour déterminer le nombre optimal de clusters pour notre classification, nous avons évalué le Silhouette Score pour différents nombres de clusters. Le Silhouette Score mesure la qualité de la classification en évaluant le degré d'association de chaque point à son propre cluster par rapport aux autres clusters. Une valeur élevée du Silhouette Score indique que les points sont bien regroupés et distincts des autres clusters.

Nous avons testé plusieurs valeurs pour le nombre de clusters (k) comme illustré dans la Figure 3.16, et observé que le score était particulièrement bon pour $k=4$. Ce résultat indique que quatre clusters permettent de bien séparer les services IoT en groupes distincts, ce qui est idéal pour notre objectif de classification en catégories "bronze", "argent", "or", et "platine". De plus, le choix de quatre clusters s'est révélé être le plus approprié pour notre travail, facilitant l'identification des services de haute qualité tout en tenant compte des critères de QoS et de la consommation d'énergie.

La Figure 3.16 présente la silhouette score pour différent nombre de clusters.

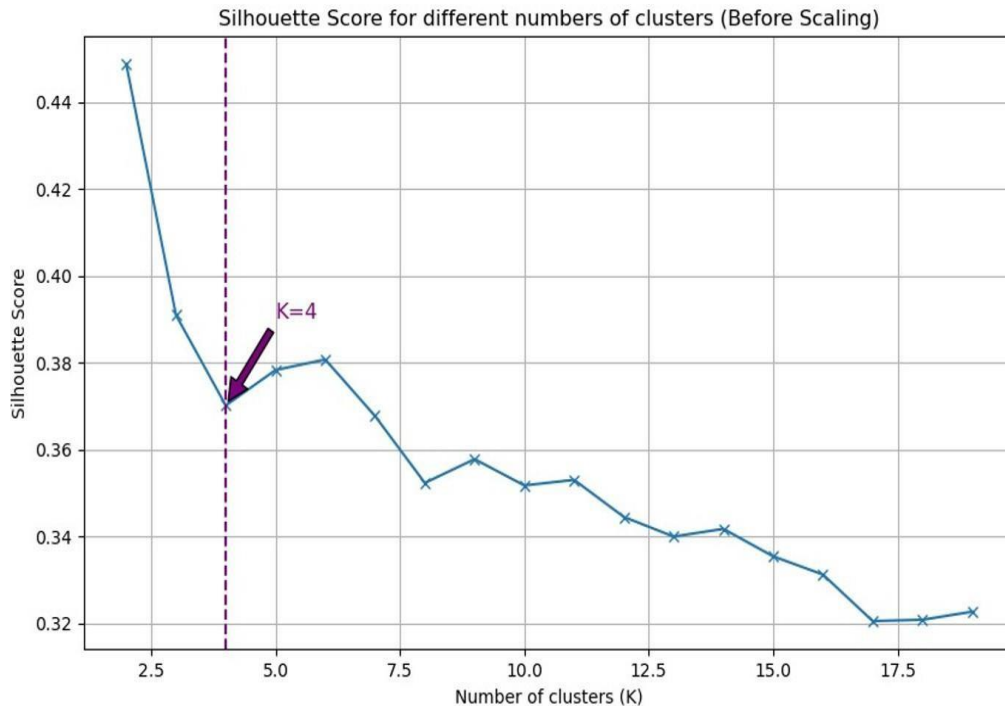


FIGURE 3.16 – Silhouette score pour différents nombres de clusters.

Silhouette Score avec suppression d'un attribut

Dans le cadre de l'amélioration de notre modèle de classification, nous avons évalué l'impact de chaque attribut QoS sur le Silhouette Score. L'objectif était d'identifier si la suppression de certains attributs pouvait améliorer la qualité de la classification.

Pour ce faire, nous avons recalculé le Silhouette Score en excluant un attribut à la fois et en observant l'effet sur la qualité des clusters. Ce processus a permis de déterminer l'importance relative de chaque attribut dans le modèle de classification.

La Figure 3.17 présente les valeurs de silhouette Score obtenue pour chaque cas.

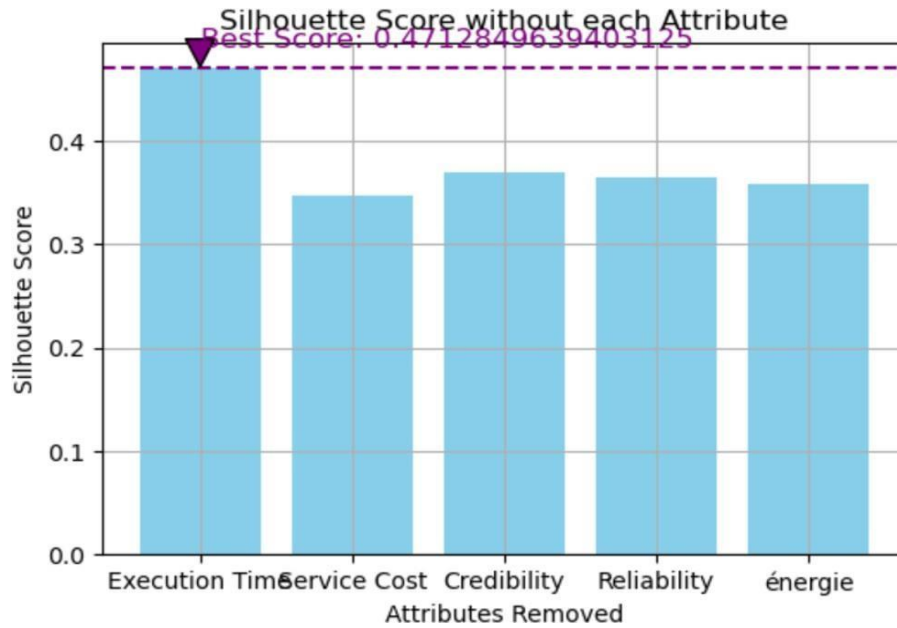


FIGURE 3.17 – Les valeurs de silhouette Score sans chaque attribut.

Selon les résultats obtenus dans les sections 3.16 et 3.17, après avoir apporté ces modifications, en particulier l'exclusion de l'attribut "exécution time", nous avons observé une amélioration notable de la classification. Les clusters formés sont devenus plus distincts et cohérents, comme en témoigne l'augmentation du Silhouette Score. Cette optimisation a permis de mieux distinguer les services de haute qualité des autres, renforçant ainsi la précision de notre modèle de classification.

Les résultats de ces améliorations sont présentés dans le graphe 3.18, qui illustre la répartition finale des services IoT dans les quatre clusters "bronze", "argent", "or" et "platine". Ce graphe démontre clairement l'efficacité de notre approche et la pertinence des ajustements apportés.

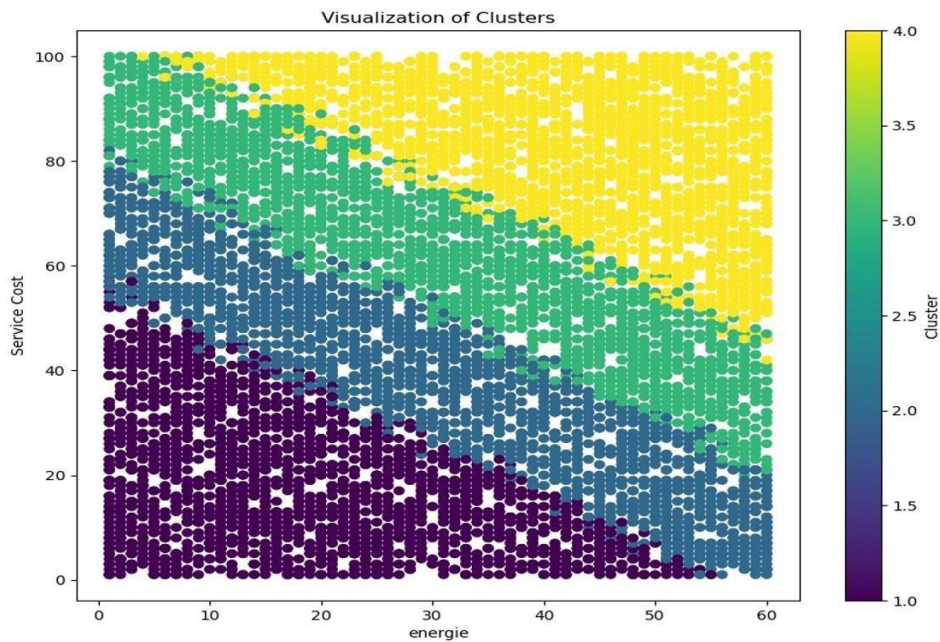


FIGURE 3.18 – Classification des services après l’amélioration de K-means.

3.7.4 Recherche d’une composition quasi optimale via K-GA

Après avoir appliqué la classification par K-means, nous abordons la deuxième étape de notre approche, à savoir l’application du GA. Dans ce qui suit, nous présentons les résultats obtenus par K-GA et les comparons avec ceux obtenus par l’application de GA traditionnel.

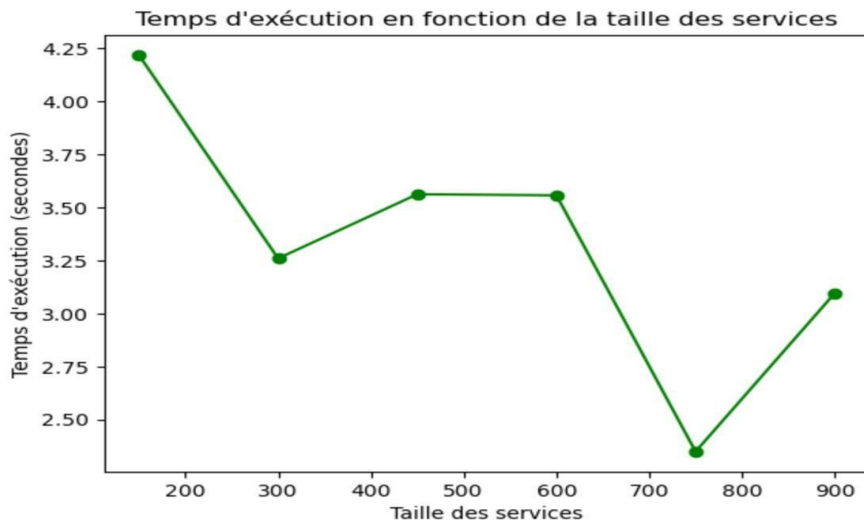


FIGURE 3.19 – Temps d’exécution en fonction de la taille des services.

Le graphe représenté dans la Figure 3.13 montre la relation entre la taille des services

et le temps d'exécution nécessaire pour trouver la meilleure solution dans chaque taille de service.

CS	150	300	450	600	750	900
K-GA	0.006	0.003	0.003	0.0001	0.0001	0.0001
GA	0.015	0.004	0.004	0.004	0.002	0.002

TABLEAU 3.3 – Valeurs de fitness de K-GA et de GA.

Le Tableau 3.2 récapitule les valeurs de fitness obtenues par K-GA en comparaison avec celles obtenues par GA traditionnel. Les résultats montrent une amélioration significative de la fitness, confirmant que la combinaison de K-means et de GA permet d'obtenir des compositions de services IoT plus optimales en termes de QoS et de consommation énergétique.

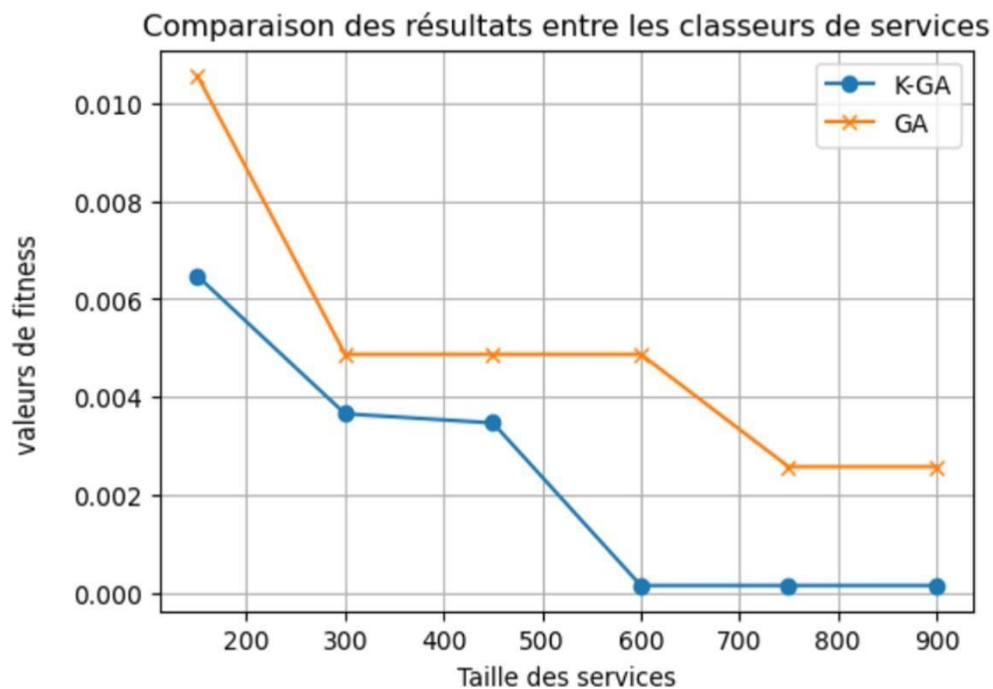


FIGURE 3.20 – Évaluation des valeurs de fitness de K-GA et GA traditionnel.

Le graphique présenté dans la Figure 3.20 illustre cette amélioration, montrant que K-GA offre une meilleure performance en termes de fitness par rapport aux méthodes GA individuelles. Cela démontre l'efficacité de notre approche combinée, qui tire parti des points forts de chaque algorithme pour améliorer les résultats globaux. Cette synergie entre K-means et GA permet de surmonter les limites de chaque méthode lorsqu'elles sont utilisées seules, garantissant ainsi des compositions de services IoT plus efficaces et performantes.

3.8 Conclusion

Dans ce chapitre, nous avons présenté notre approche K-GA, l'environnement de développement utilisé, notamment les différents outils et bibliothèques. Nous avons également présenté le Scénario d'application de K-GA. Ensuite, nous avons évalué les performances de notre approche K-GA en la comparant au GA sans réduction de l'espace de recherche. Les résultats expérimentaux montrent que notre approche obtient des résultats satisfaisants en termes de valeurs de fitness QoS.

Conclusion générale et perspectives

Conclusion générale

Dans ce mémoire, nous avons étudié le domaine de l'IoT et son impact croissant sur la technologie et la société moderne. L'IoT relie divers dispositifs via Internet permettant une interaction fluide entre le monde physique et l'espace d'information. Notre recherche s'est focalisée sur la composition de services IoT, un processus essentiel pour répondre aux demandes complexes des utilisateurs en combinant plusieurs services individuels. Nous avons d'abord établi un cadre théorique introduisant les concepts fondamentaux de l'IoT et nous avons étudié par la suite certaines approches existantes pour la composition de services IoT avec prise en compte des QoS et de l'énergie.

Dans ce travail, nous avons proposé une approche qui combine K-means avec GA que nous avons nommée K-GA. K-GA est proposé pour la résolution du problème de sélection de composition des services avec prise en compte des QoS et la consommation d'énergie. L'algorithme k-means permet de classer les services candidats présents dans le dataset en quatre catégories : platine, or, argent et bronze. Les services présentant les meilleures valeurs de QoS sont classés dans la catégorie platine, suivis de ceux appartenant à la catégorie or. Ensuite, nous réduisons l'espace de recherche en ne gardant que les services de la catégorie or et platine. Cette réduction de l'espace de recherche a permis d'optimiser le temps d'exécution de K-GA tout en maintenant une bonne qualité de la solution retournée.

Nous avons choisi comme scénario pour valider notre approche la gestion de la santé à domicile dans une maison intelligente. Cette maison est équipée de divers capteurs de santé et d'appareils connectés pour assurer un suivi régulier de l'état de santé des résidents. Les résultats d'évaluation de K-GA, en comparaison avec le GA original sans l'utilisation de k-means, démontrent que notre approche propose des meilleurs résultats en termes de valeurs de fitness.

Perspectives

Dans le cadre de l'évolution de notre travail, plusieurs perspectives peuvent être envisagées :

- La création d'un dataset spécifique et réelle pour les services IoT pour faciliter les futures recherches et tests.
- Explorer l'utilisation de nouvelles métaheuristiques et techniques d'apprentissage profond pour améliorer la qualité de la classification des services IoT.
- Implémenter l'approche proposée dans le monde réel.

Références

- [1] *L'histoire de l'IoT.* <https://www.siemens.com/fr/fr/entreprise/stories/innovation-technologies/jumeau-numerique/histoire-iot.html>. Consulté le 03/05/2024.
- [2] *Forbes.fr, Internet des objets (IOT) et appareils intelligents.* <https://www.forbes.fr/technologie/internet-des-objets-ido-et-appareils-intelligents-les-tendances-pour-2024/>. Consulté le 02/05/2024.
- [3] *Moreno M. V., Zamora M. A., Skarmeta A. F: (2015), An IoT based framework for user-centric smart building services, International Journal of Web and Grid Services. (pp. 78-101).*
- [4] *N. Kashyap, A. C. Kumari et R. Chhikara : (2020). Service Composition in IoT using Genetic algorithm and Particle swarm optimization, (PP.56-64).* <https://doi.org/10.1515/comp-2020-0011>.
- [5] *L. Atzori, A. Iera et G. Morabito: (2010). "The Internet of Things : A survey.,". Comput. Networks, vol. 54, (pp. 2787–2805).*
- [6] *Analyse et perspectives de l'Internet des Objets. Horizons 2013- 2020 CITCEura RFID.*
- [7] *Dr. Ovidiu Vermesan, Dr. Peter Friess: (2014). Internet of Things – From Research and Innovation to Market Deployment. RIVER PUBLISHERS SERIES IN COMMUNICATIONS.*
- [8] *M. Roopa, S. Pattar, R. Buyya, K. Venugopal, S. Iyengar et L. Patnaik : (2019). Internet social des objets (SIoT) : fondements, domaines d'intervention, examen systématique et orientations futures. Calculer. Commun. Vol.139. (pp. 32-57).* <https://doi.org/10.1016/J.COMCOM.2019.03.009>.
- [9] *S. Pattar, R. Buyya, K. Venugopal, S. Iyengar et L. Patnaik : (2018). Recherche des ressources IoT : principes fondamentaux, exigences, examen complet et orientations futures. Enquêtes et didacticiels sur les communications IEEE, vol.20, (pp. 2101-2132).* <https://doi.org/10.1109/COMST.2018.2825231>.

- [10] Z. O. A. I. Meftah: (2021). *Une approche cloud computing basée IoT pour le smart House. (Doctoral dissertation, Université de mohamed kheider biskra).*
- [11] F. Lemoine (2019). *Internet des Objets centré service autocontrôlé. (Doctoral Dissertation, Conservatoire national des arts et métiers-CNAM).*
- [12] Saleh, Imad : (2018) *Internet des Objets (IOT) : Concepts, Enjeux, Défis et Perspectives.* openscience.fr.
- [13] K. K. Patel et S. M. Patel, PG Scholar Assistant Professor2: (May, 2016). *Internet of Things-IOT: Definition, Characteristics, Architecture, Enabling Technologies, Application and Future Challenges. International Journal of Engineering Science and Computing. (pp.6122-6163).*
- [14] *Le boom silencieux de l'IoT : quand les objets connectés vont envahir notre vie quotidienne.* <https://www.latribune.fr/opinions/tribunes>. Page consulté le 05/05/2024.
- [15] *Qu'est-ce que l'IoT et pourquoi mener une stratégie d'IoT.* <https://www.digora.com/fr/blog/quest-ce-que-liot-et-pourquoi-mener-une-strategie-diot> . page consulté le 03/05/2024.
- [16] S. Yamamoto, S. Matsumoto et M. Nakamura: (2012) *Using Cloud Technologies for Large-Scale House Data in Smart City. 2012 IEEE 4th International Conference on Cloud Computing Technology and Science. (pp. 141- 148). IEEE.*
- [17] *Maison Intelligente.* <https://www.objetconnecte.com/maison-intelligente-179-millions-2024/> . Page consulté le 04/05/2024.
- [18] J. Wan, H. Cai et K. Zhou : (2015). *Industrie 4.0: enabling technologies. In Proceedings of the 2015 international conference on intelligent computing and internet of things. (pp. 135-140).*
- [19] R. Chabou : (2023). *Système de surveillance intelligent basé sur l'IoT pour la prédiction des maladies cardiaques. (Thèse de doctorat, Université de Larbi Tébessi).*
- [20] C. Makhlouf: (2020). *Conception et Réalisation d'un Système d'IoT pour l'Economie de l'Energie dans le Cadre d'une Cité Intelligente (Smart City). (Thèse de doctorat, Université Larbi Tébessi).*
- [21] A. Musaddiq, Y. B. Zikria, O. Hahm, H. Yu, A. K. Bashir et S. W. Kim: (2018). *A Survey on Resource Management in IoT Operating Systems. IEEE, (pp. 8459-8482).*

- [22] M. Khan et K. Salah : (2017). *Sécurité IoT : examen, solutions blockchain et défis ouverts*. *Futur Gén. Calculer. Système*, vol.82, (pp. 395-411). <https://doi.org/10.1016/J.FUTURE.2017.11.022>.
- [23] *Le boom silencieux de l'IoT : quand les objets connectés vont envahir notre vie quotidienne*. <https://www.latribune.fr/opinions/tribunes>. Page consulté le 05/05/2024.
- [24] K. K. Patel et S. M. Patel : (2016), *International Journal of Engineering Science and Computing*, "Internet des objets-IOT : Définition, caractéristiques, architecture, technologies habilitantes, applications et défis futurs".
- [25] Y. Karouani et H. Ziyati : *Big data dans l'architecture de l'IoT pour les villes intelligentes*.
- [26] P. Tûma : (2006). *Intergiciel. Conférence internationale ACS/IEEE 2006 sur les services omniprésents* (pp.181-182). https://doi.org/10.1007/978-1-4302-1937-8_17.
- [27] J. M. Alliot : (2003). "Qu'est-ce que le 'Middleware,'" (pp. 1-17), [Online]. Disponible. <http://www.recherche.enac.fr/~alliot/middle.pdf>.
- [28] Bubble, Sketch: "Architecture Middleware", [Online]. Disponible : <https://images.app.goo.gl/QYa5rd7c8wHnLrMi8>.
- [29] T. Lau, H. Wang et C. Chuang : (2011). *Une définition du service comme base pour le développement de la science des services. Conférence internationale conjointe 2011 sur les sciences des services*, (pp.49-53). <https://doi.org/10.1109/IJCSS.2011.18>.
- [30] *L'architecture orientée services (SOA)*. <https://www.redhat.com/fr/topics/cloud-native-apps/what-is-service-oriented-architecture>. Page consultée le 03/05/2024.
- [31] *Architecture orientée services (SOA) et cloud computing*. <https://www.service-architecture.com/articles/cloud-computing/service-oriented-architecture-soa-and-cloud-computing.html>. Page consultée le 03/05/2024.
- [32] V. Issarny, G. Bouloukakis, N. Georgantas et B. Billet : (2016). *Revisiter l'architecture orientée services pour l'IoT : une perspective middleware*. Dans : Sheng, Q., Stroulia, E., Tata, S., Bhiri, S. (éd.) *Informatique orientée services. ICSOC 2016. Notes de cours en informatique*, vol 9936. Springer, Cham. https://doi.org/10.1007/978-3-319-46295-0_1.

- [33] Oracle : “*Fusion Middleware Concepts Guide,*” p 4, 2015, [Online]. Disponible : http://docs.oracle.com/cd/E21764_01/core.1111/e10103/intro.htm#ASCON109.
- [34] J. Zhou, J. Rieki et M. Ylianttila : (2009). *Modélisation de la composition du service et exploration de ses caractéristiques. Congrès sur les services 2009 - I, (pp.446-451).* <https://doi.org/10.1109/SERVICES-I.2009.10>.
- [35] S. Sun et J. Zhao : (2012). *A decomposition-based approach for service composition with global QoS guarantees. Inf. Sci. Vol.199, (pp. 138-153).* <https://doi.org/10.1016/j.ins.2012.02.061>.
- [36] H. Ouassila, R. Boucetti et S. M. Hemam : (2022). *f King Saud University - Computer and Information Sciences.*
- [37] Alrifai, M., Risse T. et W NejdI: (2012). *A hybrid approach for efficient web service composition with end-to-end QoS constraints. ACM Transaction on the Web(TWEB). vol. 6 no. 2, (pp. 7-31).*
- [38] H. Ouassila, R. Boucetti et S. M. Hemam : (2022), *QOS-Aware IOT Services Composition : A Survey. (pp. 477-488).*
- [39] M. Haghi Kashani, A. M. Rahmani, and N. Jafari Navimipour: (2020). “*Quality of service-aware approaches in fog computing,*” *Int. J. Commun. Syst., vol. 33, (pp. 1–34).* <https://doi:10.1002/dac.4340>.
- [40] Z. Houhamdi et B. Athamena: (2020), “*Identity identification and management in the internet of things*”. *Int. Arab J. Inf. Technol., vol. 17, no. (pp. 645–654),doi : 10.34028/ia-jit/17/4A/9.*
- [41] R. Buyya, A. V. Dastjerdi: (2016). *Internet of Things: Principles and Paradigms. Elsevier.*
- [42] I. Aoudia : (2022). *Composition adaptative de services pour l’Internet des objets. (Thèse de doctorat, Université de Mohamed Kheider Biskra).*
- [43] I. Charfi : (2013). *Détection automatique de chutes de personnes basée sur des descripteurs spatio-temporels. (Thèse de doctorat, Université de Bourgogne).*
- [44] <https://www.dunod.com/sciences-techniques/machine-learning-avec-scikit-learn-mise-en>.
Page consultée le 09/05/2024.
- [45] L. De Matteis, S. Janny, S. Nathan, W. Shu Quartier : (2022). *Introduction à l’apprentissage automatique. Culture Sciences de l’Ingénieur.*

- [46] *Hands-On*, Géron Aurélien.: (2019). *Machine Learning with Scikit-Learn and Tensor- Flow: Concepts, Tools, and Techniques to Build Intelligent Systems*. O'Reilly Media.
- [47] CharfiOctober, Imen : (2013). *Détection automatique de chutes de personnes basée sur des descripteurs spatio-tempo* Theses. Université de Bourgogne.
- [48] <https://ai.plainenglish.io/a-comparision-of-classification-and-regression/-in-machine-learning-5501879f498?gi=e13064c59434>. Page consultée le 10/05/2024.
- [49] A. Géron: (2019). *Hands-On Machine Learning with Scikit-Learn and Tensor- Flow: Concepts, Tools, and Techniques to Build Intelligent Systems*. O'Reilly Media.
- [50] *Métaheuristique*. wikipédia. [En ligne].
[https://fr.wikipedia.org/wiki/MÃtaheuristique](https://fr.wikipedia.org/wiki/M%C3%A9taheuristique). Page consultée le 09/05/2024.
- [51] Dr, LEMOUARI : (2014). *Introduction aux Métaheuristiques*. Université de Jijel.
- [52] E. Azhir, N. Jafari Navimipour, M. Hosseinzadeh, A. Sharifi, A. Darwesh : (2019). *Deterministic and non-deterministic query optimization techniques in the cloud computing*. *Concurrency Computat Pract Exper*, e5240. <https://doi.org/10.1002/cpe.5240>.
- [53] A. Urbietta, A. Gonz alez-Beltran, S. Ben Mokhtar M. Anwar Hossain L. Capra: (2016). *Adaptive and context-aware service composition for IoT-based smart cities*, *Future Generation Computer Systems*. <http://dx.doi.org/10.1016/j.future.2016.12.038>.
- [54] T. Baker, M. Asim, H. Tawfik, B. Aldawsari, R. Buyya: *An energy-aware service composition algorithm for multiple cloud-based IoT applications*. (2017), *Revue Journal of Network and Computer Applications*, Vol 89, (pp.96–108).
- [55] Z. Chai, M. Du, G. Song : (2021). *A fast energy-centered and QoS-aware service composition approach for Internet of Things*. *Applied Soft Computing Journal*, Vol 100.
- [56] Azhir, E., Navimipour N. J. Hosseinzadeh M. Sharifi A. Darwesh A: (February 2019). *Deterministic and non-deterministic query optimization techniques in the cloud computing*. *Journal of Cloud Computing*. <https://doi.org/10.1002/cpe.5240>.
- [57] R. Boucetti, S. M. Hemam, O. Hioual: (2022). *QoS-Aware IoT Services Composition: A Survey*. (pp. 477-488).
- [58] G. J. Ibrahim, T. A. Rashid, M. O. Akinsolu: (2022). *An energy efficient service composition mechanism using a hybrid meta-heuristic algorithm in a mobile cloud environment*. *Journal of Parallel and Distributed Computing*, Vol.143, (pp. 77-87).
- [59] QoS-R. Wang, J. Lu: (2022). *QoS-Aware Service Discovery and Selection Management for Cloud-Edge Computing Using a Hybrid Meta-Heuristic Algorithm in IoT*. *Wireless Personal Communications*, Vol 126, (pp. 2269-2282). (PP. 2269-2282).

- [60] S. H. Tari, M. E. Khanouche, A. Abdelkamel: (2023). *A Group Teaching Optimization-Based Approach for Energy and QoS-Aware Internet of Things Services Composition*.
- [61] Hassbgli902. *Techniques de résolution pour les problèmes d'optimisation combinatoire*. [PDF].
- [62] *L'ensemble de données IoTS_Dataset*. <https://zenodo.org/records/10440967> . Consulté le 14/05/2023.
- [63] M. Ahmed, R. Seraj, S. M. S. Islam : (2020). *The k-means Algorithm: A Comprehensive Survey and Performance Evaluation*. *Electronics*, vol 1295. <https://doi.org/10.3390/electronics9081295>.
- [64] *K-Means: Clustering Algorithm*. <https://towardsdatascience.com/k-means-a-complete-introduction-1702af9cd8c>. Page consultée le 05/06/2024.
- [65] B. Sanka, N : (2021). *Étude comparative et choix optimal du nombre de classes en classification et réseaux de neurones. : Application en science des données (Mémoire de maîtrise)*. Université du Québec à Trois-Rivières.
- [66] J. H. Holland : (1975). *Adaptation in Natural and Artificial Systems*. University of Michigan Press. DOI : 10.5555/528634.
- [67] M. Mitchell : (1996). *An Introduction to Genetic Algorithms*. MIT Press. DOI : 10.7551/mitpress/3927.001.0001.
- [68] C. Fleurent, J. A. Ferland : (1996). *Algorithmes génétiques hybrides pour l'optimisation combinatoire*. *Revue française d'automatique, d'informatique et de recherche opérationnelle. Recherche opérationnelle*, (pp. 373-398).
- [69] *Schéma général d'un algorithme génétique*. <https://khayyam.developpez.com/articles/algo/genetic/> . Page consultée le 07/06/2024.
- [70] C. Sadouki, A. Tari : (2017). *Elephant Herding Optimization for Service Selection in QoS-Aware Web Service Composition*. *International Journal of Computer and Information Engineering*.

- [71] Python. <https://datascientest.com/python-tout-savoir> . Page consulté le 04/06/2024.
- [72] / home, Project jupyter. <https://jupyter.org/>. Page consulté le 04/06/2024.
- [73] Pandas. <https://datascientest.com/pandas-python-data-science> . Page consulté le 04/06/2024.
- [74] Learn scikit. <https://datascientest.com/tout-savoir-sur-scikit-learn>. Page consulté le 04/06/2024.
- [75] Matplotlib. <https://datascientest.com/matplotlib-tout-savoir> . Page consulté le 04/06/2024.
- [76] NumPy. <https://datascientest.com/numpy> . Page consulté le 04/06/2024.
- [77] Données, Qu'est-ce que le prétraitement des. <https://www.astera.com/fr/type/blog/data-preprocessing/> . Page consulté le 05/06/2024.
- [78] Souquet, A, Radet F. G : (2004). Algorithmes génétiques. TE de fin d'année. Tutorat de M. Philippe Audebaud.

Résumé

L'Internet des objets (IoT) est une technologie révolutionnaire qui transforme la manière dont les objets interagissent et communiquent entre eux. Il s'agit d'un système intégrant des dispositifs informatiques, des réseaux et des machines, leur permettant de se connecter et de transmettre des données de manière autonome, sans nécessiter d'interaction humaine. Ce système interconnecte d'objets physiques et virtuels, facilitant ainsi la collecte et le partage d'informations pour soutenir les tâches humaines et les décisions éclairées. La composition de services IoT est cruciale car les utilisateurs ont des besoins variés et complexe qui ne peuvent être satisfait par un seul service IoT. C'est pourquoi, nous proposons une approche de composition de services IoT prenant en compte les exigences de qualité de service (QoS). Cette approche, nommée K-GA, combine l'algorithme de clustering K-means avec un algorithme génétique. K-means est utilisé pour classer les services tandis que GA est utilisé pour rechercher une solution quasi optimale qui satisfait les besoins de l'utilisateur en termes de QoS et de consommation d'énergie. Notre approche a fourni de bons résultats en comparaison avec GA sans réduction de l'espace de recherche.

Mot clés : IoT, composition de services, l'algorithme de clustering K-means, GA, métaheuristique, QoS.

Abstract

The Internet of Things (IoT) is a revolutionary technology that is transforming the way objects interact and communicate with each other. It is a system integrating computing devices, networks, and machines, enabling them to connect and transmit data autonomously, without the need for human interaction. This system interconnects physical and virtual objects, facilitating the gathering and sharing of information to support human tasks and informed decisions. The composition of IoT services is crucial, as users have varied and complex needs that cannot be met by a single IoT service.

This is why we are proposing an approach to IoT service composition that takes quality of service (QoS) requirements into account. This approach, named K-GA, combines the K-means clustering algorithm with a genetic algorithm. K-means is used to rank services, and the genetic algorithm is used to search for a near-optimal solution that satisfies user requirements in terms of QoS and energy consumption. Our approach performed well compared to GA without search space reduction.

Keywords: IoT, service composition, the K-means clustering algorithm, GA, metaheuristics, QoS.

