Faculté des sciences

Département d’informatique

**Mémoire de projet de fin d’études**

Pour l’obtention du titre

**Master spécialisé en big data et cloud computing**

**Filiere:** Master Big Data & cloud computing

**Sujet:**

Comment l’apprentissage automatique (ou les modèles d’apprentissage profond) et le réseau neuronal aident au routage dans le réseau VRP

**Réalisé par:** Doha SASSA **Encadré par:** Pr. Mme Hala KHANKHO

**Coordinateur** : Pr. JAAFAR ABOUCHABAKA

**Membre de jury : Pr**. Jaafar ABOUCHABAKA

**Pr**. Najat RAFALIA

Année universitaire : 2022-2023

***Dédicace***

***À mon père***

Disparu très tôt, qui m’a toujours encouragé et motivé durant mes premières années d’études.

J'espère que, du monde qui est sien maintenant, il soit content d’apprendre ce que sa fille a pu

Accomplir. Puisse Dieu, le tout puissant, l'avoir en sa sainte miséricorde !

***À ma mère***

Aucune dédicace ne saurait exprimer mon respect, mon amour et ma considération pour les

Sacrifices que tu as consenti pour mon instruction et mon bien être. Puisse Dieu, le très haut,

T’accorder santé, bonheur et longue vie et faire en sorte que jamais je ne te déçoive !

***À mes Frere***

Pour vos encouragements, votre écoute

***À mes amis***

Pour vos encouragements, votre écoute et en souvenir des beaux moments que nous avons passé

Ensemble.

***Je dédie ce travail...***

***Remerciement***



Au terme de ce projet de fin d’études, il m’est très agréable de remercier toutes les personnes qui ont contribué, de près ou de loin, à la réalisation de ce modeste travail et je leur demande de bien vouloir m’excuser de ne pas pouvoir les citer toutes et qu’ils sachent que je leur dois ma sincère gratitude. Mes sincères remerciements vont au professeur Hala Khankhour, mon encadrante, qui n’a ménagé aucun effort pour m’apporter assistance durant toutes les étapes de mon travail. Ses remarques, ses qualités d’enseignant et sa rigueur m’ont été d’un apport considérable. Ainsi, il m’a offert la latitude de travailler très librement sur ce projet et m’a accordé, tout au long de la période de ce travail, le temps nécessaire pour me renseigner, me conseiller, m’orienter et m’acquitter les réponses nécessaires à mes interrogations multiples. Avec elle, j’ai pu goûter aux plaisirs de la mise en pratique de mes connaissances théoriques. Si j’ai pu réaliser ce travail dans les meilleures conditions, c’est grâce à sa disponibilité malgré sa grossesse. Que Dieu soit à ses rangs.

Je tiens également à exprimer ma gratitude envers les membres du jury, notamment le Professeur Najat Rafalia, notre instructeur en méthodologie de recherche. Elle a toujours été une source essentielle d'informations pour la rédaction d'articles et de rapports de projets de doctorat ou de master. Grâce à ses précieux conseils, nous avons découvert un sujet d'actualité passionnant : "les systèmes multi-agents". C'est ce choix qui a guidé la rédaction de notre mémoire. De plus, un grand merci au Professeur Jaafar Abouchabaka pour le temps qu'il a consacré à la lecture et à la correction de ce projet de fin d'étude.



**Résumé**

La conception globale du développement durable comprend un mécanisme pour atteindre l’objectif économique par lequel tous les besoins humains fondamentaux sont satisfaits par l’utilisation de la technologie tout en prenant soin de notre environnement naturel. Pour réduire la pollution atmosphérique et le trafic, nous devons commencer par optimiser les processus de la chaîne d’approvisionnement qui se produisent sur notre globe chaque jour et nuit. Cet objectif sert également tous les citoyens en réduisant les embouteillages quotidiens.

Ce projet se concentre sur la résolution d’une variante du problème de routage des véhicules (VRP) dans laquelle une fenêtre de temps est associée à chaque service client et certains services nécessitent des visites simultanées à différents véhicules pour être effectués. Le problème est donc appelé VRP avec des fenêtres de temps et des contraintes de synchronisation. Nous présentons un algorithme de recuit simulé (SA) qui intègre plusieurs techniques de recherche locales pour résoudre ce problème. Les expériences sur des exemples de littérature montrent que notre AS est rapide et surpasse les approches existantes. À notre connaissance, c’est la première fois que la recherche locale dédiée. Nous présentons un algorithme CNN (réseau de neurones conventionnel) pour détecter les modèles de congestion du trafic, les groupements de clients, les zones à faible demande, ces informations peuvent être utilisées pour générer des itinéraires et des temps de déplacement et des allocations de ressources. Ce modèle permet de classer les types de véhicules selon leurs contraintes de capacité. Nous proposons une architecture multi-agents pour garantir l’autonomie de notre programme, la flexibilité et l’adaptabilité, le parallélisme et l’efficacité. Pour réduire le temps d’exécution du programme en parallélisant les opérations.

**Abstract**

The global conception of sustainable development includes a mechanism for achieving the economic goal by which all fundamental human needs are met by the use of technology with simultaneously taking care of our natural environment. To reduce air pollution and traffic we need to begin with optimizing the supply chain processes that occur on our globe every day and night. This purpose also serves every citizen by reducing everyday traffic congestions.

This project focuses on solving a variation of the vehicle routing problem (VRP) in which a time window is associated with each customer service and some services require simultaneous visits to different vehicles to be accomplished. The problem is therefore called VRP with time windows and synchronization constraints .We present a simulated annealing (SA) algorithm that integrates several local search techniques to address this problem. Experiments on literature examples show that our SA is fast and outperforms existing approaches. To our knowledge, this is the first time that dedicated local search. We present a CNN (conventional neural network) algorithm to detect traffic congestion patterns, customer groupings, low demand areas, this information can be used to generate routes and travel times and resource allocations. This model can be used to classify vehicle types according to their capacity constraints. We propose a multi-agent architecture to guarantee the autonomy of our program, flexibility and adaptability, parallelism and efficiency. To reduce program execution time by parallelizing operations.

**Keywords:** vehicle routing problem, machine learning, deep learning, multi-agent system, CNN, JAVA, JADE agent.

**Liste des abréviations**

|  |  |
| --- | --- |
| VRP | Vehicle Routinng Problem |
| TSP | Traveling System Problem |
| QOT | Quality of traveling |
| its | Intelligent transprtation system |
| SOC-VRP | Service oriented cooperative |
| CNN | Convolutional neural network |
| SA | Simulated anealing |
| JADE | Java agent development framwork |
| SMA | System multi agent |
| GPU | Graphics processing Unit |
| CUDA | Compute Unified Device Architecture |

**Liste des figures**

[**Figure 1 :** Service-oriented. Cooperative VRP in C-ITS. 12](#_Toc153112862)

[**Figure 2 :** TSP vs VRP 14](#_Toc153112863)

[**Figure 3 :** TSP pose la question suivante : Étant donné une liste de villes et les distances entre chaque paire de villes, quel est le chemin le plus court possible qu’un vendeur peut prendre pour visiter chaque ville et retourner à la ville d’origine? 15](#_Toc153112864)

[**Figure 4 :** Diagramme illustrant comment un réseau de neurones traite les données. 18](#_Toc153112865)

[**Figure 5 :** Architecture CNN simple 18](#_Toc153112866)

[**Figure 6 :** Tableau illustrant les différences entre un réseau de neurones convolutifs et un réseau de neurones récurrents. 19](#_Toc153112867)

[**Figure 7 :** dimension d’image (width & height) 19](#_Toc153112868)

[**Figure 8 :** dimension d’image (width & height) 20](#_Toc153112869)

[**Figure 9 : pooling layer** 21](#_Toc153112870)

[**Figure 10 :** couche de sortie 21](#_Toc153112871)

[**Figure 11 :** L’algorithme de la fonction d’acceptation 24](#_Toc153112872)

[**Figure 12 :** l’optimiseur de l’algorithme SA 25](#_Toc153112873)

[**Figure 13 :** le diagramme de flux de l’algorithme recuit simulé 25](#_Toc153112874)

[**Figure 14 :** montrant la solution candidate, son énergie et la température à chaque étape 26](#_Toc153112875)

[**Figure 15 :** Architecture d’agent 27](#_Toc153112876)

[**Figure 16 :** Objet «versus» Agent 28](#_Toc153112877)

[**Figure 17 :** Différence entre un agent et objet 29](#_Toc153112878)

[**Figure 18 :** Réactivité 30](#_Toc153112879)

[**Figure 19 :** Proactivité 31](#_Toc153112880)

[**Figure 20 :** Système des activités de coopération 37](#_Toc153112881)

[**Figure 21 :** Système de communication d’après Claude Shannon 37](#_Toc153112882)

[**Figure 22 :** tableau de situation des sockets dans le modèle OSI 40](#_Toc153112883)

[**Figure 23 :** flux typique d'événements 41](#_Toc153112884)

[**Figure 24 :** API socket 42](#_Toc153112885)

[**Figure 25 :** diagramme de classe du système 46](#_Toc153112886)

[**Figure 26 :** diagramme de séquence du système. 48](#_Toc153112887)

[**Figure 27 :** donne une présentation synthétisée du diagramme de cas d’utilisation de notre système multi agent.. 50](#_Toc153112888)

[**Figure 28 :** python vs R 56](#_Toc153112889)

[**Figure 29 :** moteur OCR 57](#_Toc153112890)

[**Figure 30 :** moteur Tesseract de google 57](#_Toc153112891)

[**Figure 31 :** Processus de tesseract 58](#_Toc153112892)

[**Figure 32 :** les applications de geopy 62](#_Toc153112893)

[**Figure 33 :** explication de platforme jade 65](#_Toc153112894)

[**Figure 34 :** Architecture d’une plateforme JADE 65](#_Toc153112895)

[**Figure 35 :** dataframe des donnes des véhicules 67](#_Toc153112896)

[**Figure 36 :** histogrammedistribution de voitures 68](#_Toc153112897)

[**Figure 37 :** diagramme circulaore de la distribution des vehicules 68](#_Toc153112898)

[**Figure 38 :** capture d’écran du maroc. 70](#_Toc153112899)

[**Figure 39 :** Congestion de trafic de chaque ville 71](#_Toc153112900)

[**Figure 40 :** Résultat du système 72](#_Toc153112901)

**Table des matières**

[**Introduction général** 10](#_Toc153112551)

[**L’état de l’art** 12](#_Toc153112552)

[**Introduction** 12](#_Toc153112553)

[**1.** **C’est quoi le VRP ?** 12](#_Toc153112554)

[1.2 Problème de voyageur de commerce (TSP) 13](#_Toc153112555)

[1.3 Pourquoi est-ce un tel défi de résoudre le VRP ? 14](#_Toc153112556)

[**2 IA, l'apprentissage par machine et le Deep Learning** 15](#_Toc153112557)

[Introduction 15](#_Toc153112558)

[2.1 Introduction au réseau de neurones à convolution 16](#_Toc153112559)

[2.2 Convolution Neural Network 17](#_Toc153112560)

[2.2.1 Architecture CNN 17](#_Toc153112561)

[2.2.2 Fonctionnement des couches convolutives 18](#_Toc153112562)

[2.2.3 Couches utilisées pour construire ConvNets 19](#_Toc153112563)

[2.2.4 Avantages des réseaux de neurones convolutifs (CNN) : 21](#_Toc153112570)

[2.2.5 Inconvénients des réseaux de neurones convolutifs (CNN) : 21](#_Toc153112578)

[**3** **Algorithme recuit simulé (SA)** 21](#_Toc153112579)

[Introduction 21](#_Toc153112580)

[3.1 C’est quoi l’algorithme recuit simulé ? 21](#_Toc153112581)

[3.2 Définir le problème 22](#_Toc153112582)

[3.3 Définir la fonction de perturbation 22](#_Toc153112583)

[3.4 Critère d'acceptation 22](#_Toc153112584)

[3.5 Calendrier des températures 23](#_Toc153112585)

[3.6 Exécuter l’algorithme SA 23](#_Toc153112586)

[3.7 Exemple d’application 25](#_Toc153112587)

[Conclusion 25](#_Toc153112588)

[**4** **Système multi agent** 25](#_Toc153112589)

[Introduction 25](#_Toc153112590)

[4.1 Agent 26](#_Toc153112591)

[4.2 Systèmes Multi-agent 27](#_Toc153112592)

[4.3 Des Objets aux Agents 27](#_Toc153112593)

[4.4 Propriétés d’un agent intelligent 28](#_Toc153112594)

[4.5 Propriétés des systèmes multi-agents 31](#_Toc153112595)

[4.6 Etapes de réalisation d’un SMA 37](#_Toc153112596)

[4.7 Plates-formes de développement 37](#_Toc153112597)

[Conclusion 38](#_Toc153112598)

[**5.** **Sockets** 38](#_Toc153112599)

[Introduction 38](#_Toc153112600)

[5.1 Position des sockets dans le modèle OSI 39](#_Toc153112604)

[5.2 Fonctionnement des sockets 39](#_Toc153112605)

[Conclusion 41](#_Toc153112606)

[**Expérimental** 42](#_Toc153112607)

[**1** **Conception** 42](#_Toc153112608)

[Introduction 42](#_Toc153112609)

[1.1 Diagramme de classe 42](#_Toc153112610)

[1.2 Diagramme de séquence 45](#_Toc153112611)

[1.2.2 Avantages de diagramme de séquences 45](#_Toc153112612)

[1.2.3 Cas d’utilisation de diagramme de séquence 46](#_Toc153112613)

[1.2.4 Diagramme de cas d’utilisation 48](#_Toc153112617)

[Conclusion 50](#_Toc153112618)

[**2** **Environnement de travail** 50](#_Toc153112619)

[**2.1 Python** 50](#_Toc153112620)

[2.1.1 Les origines de langages de programmation python 50](#_Toc153112621)

[2.1.2 Les avantages et inconvénients de Python 51](#_Toc153112624)

[2.1.3 Que peut-on faire avec Python ? 53](#_Toc153112625)

[2.1.4 Différents bibliothèques que nous avons utilisées dans notre projet 55](#_Toc153112626)

[2.1.4.1 PytesserAct 55](#_Toc153112634)

[2.1.4.1.1 C’est quoi Tesseract 55](#_Toc153112641)

[2.1.4.1.2 Étapes du processus d’OCR de Tesseract 57](#_Toc153112649)

[2.1.4.1.3 Amélioration du traitement des images en combinant OpenCV et Tesseract 57](#_Toc153112658)

[2.1.4.1.4 Comment fonctionne (Py)Tesseract? 58](#_Toc153112668)

[2.1.4.1.5 Limite de Tesseract 58](#_Toc153112679)

[2.1.4.2 OpenCV 59](#_Toc153112686)

[2.1.4.2.1 Les applications d’OpenCV 59](#_Toc153112694)

[2.1.4.2.2 Les fonctionnalites d’openCV 60](#_Toc153112703)

[2.1.4.2.3 Image processings 60](#_Toc153112713)

[2.1.4.3 Geopy 60](#_Toc153112721)

[**2.2 JAVA** 61](#_Toc153112722)

[2.1.1 Applications de java 61](#_Toc153112723)

[2.1.2 Plateformes de java 62](#_Toc153112726)

[2.1.3 Avantage de JAVA 62](#_Toc153112730)

[2.1.4 JADE ( java agent development framwork) 62](#_Toc153112735)

[2.1.4.1 Caractéristiques 63](#_Toc153112741)

[2.1.4.2 Platforme de JADE 63](#_Toc153112742)

[2.1.4.3 Présentation de l’architecture de JADE 64](#_Toc153112743)

[Conclusion 65](#_Toc153112744)

[**3** **Implémentation et résultats** 65](#_Toc153112745)

[Introduction 65](#_Toc153112746)

[3.1 Framework et environnement de développement 65](#_Toc153112748)

[3.2 Donnes utilisés 65](#_Toc153112749)

[3.2.1 Donnes des véhicules : 65](#_Toc153112750)

[3.2.2 Données des villes 68](#_Toc153112756)

[3.3 Résultats du système 69](#_Toc153112757)

[3.4 Comparaison 85](#_Toc153112758)

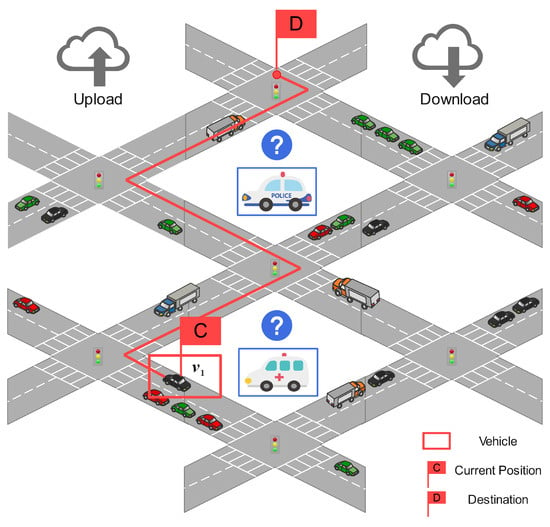
[Conclusion 85](#_Toc153112760)

[**Conclusion général et perspectives** 87](#_Toc153112761)

# **Introduction général**

Grâce aux nouvelles technologies de l'information, le système de transport est devenu plus intelligent, intégrant des capacités puissantes telles que la perception approfondie de l'environnement de circulation, la conduite autonome, la communication V2X et la coordination. Avec l'évolution et le déploiement rapides de ces technologies, il passe d'un système de transport intelligent (ITS) à une nouvelle étape connue sous le nom d’ITS coopératif (C-ITS). Dans le contexte de cette tendance émergente, les défis dominants liés aux embouteillages, aux véhicules d'urgence bloqués et au gaspillage d'énergie dans les systèmes de transport conventionnels sont progressivement résolus avec des solutions de plus en plus intelligentes. Cela a suscité une attention considérable et suscité un intérêt considérable dans les domaines universitaire et industriel.

Comme chacun le sait, le problème d’acheminement des véhicules (VRP) est l’un de ces problèmes clés qui exerce un impact profond sur les systèmes de transport. Fondamentalement, de tels problèmes de planification d'itinéraire pour un seul véhicule peuvent toujours être assimilés à un problème spécial de voyageur de commerce (TSP) avec des contraintes de priorisation, couvrant la distance la plus courte, les autoroutes et les émissions minimales, etc. Cependant, ce problème deviendra plus compliqué lors de la planification des itinéraires. Pour plusieurs véhicules dans un réseau de circulation, en particulier si ces véhicules ont des attributs de service différents, par exemple, secours d'urgence, transports en commun et usage privé, comme le montre la figure 1. Dans ce contexte particulier, la résolution de ce nouveau VRP nécessite la prise en compte non seulement des contraintes imposées aux véhicules individuels, mais aussi des interdépendances découlant de la planification d'itinéraires multi véhicules et des besoins de conduite distinctifs associés aux différents types de véhicules de service. Il est évident que les fonctionnalités coordonnées et orientées services d’un tel nouveau VRP posent inévitablement de nouveaux défis aux approches de solutions conventionnelles. Heureusement, les progrès continus des technologies de l’information et de l’intelligence artificielle (IA) permettent de résoudre de plus en plus efficacement ce problème et en ont récemment fait un point chaud de la recherche.



**Figure 1 :** Service-oriented. Cooperative VRP in C-ITS.

En règle générale, le VRP est considéré comme une variation du problème des vendeurs de commerce, et les solutions pour le VRP sont presque entièrement dérivées de solutions pour le TSP. Par conséquent, plusieurs algorithmes statiques classiques, couvrant les algorithmes de Dijsktra, Floyd et Bellman–Ford, ont été largement adoptés en raison de leur bonne compréhensibilité. Bien que ces méthodes soient simples et efficaces, les lacunes de ces algorithmes deviennent assez évidentes, y compris les processus informatiques complexes et les résultats optimaux locaux. Pour améliorer l’efficacité des déplacements, au cours des dernières décennies, plusieurs mécanismes heuristiques et dynamiques de planification des itinéraires ont été proposés, explorant rapidement les arbres aléatoires, l’optimisation des colonies de fourmis et les algorithmes évolutifs. En outre, diverses méthodes d’IA ont également été introduites pour résoudre ce problème, y compris les réseaux de neurones, les arbres de décision flous, l’apprentissage par renforcement et l’apprentissage par renforcement profond.

Comme mentionné ci-dessus, les véhicules avec des propriétés de service différentes nécessiteront toujours différentes qualités de déplacement (QoT), ce qui est un facteur clé qui doit être pris en compte dans les systèmes de transport réels. Concrètement, les véhicules d’urgence, qui ont le QoT le plus élevé, devraient subir des retards de déplacement minimes dans la mesure du possible, tandis que les navettes et les véhicules privés nécessitent respectivement un QoT moyen et le QoT le plus bas. Dans de telles situations axées sur le service, nous avons introduit le terme ITS coopératif axé sur le service (SoC-ITS), et le VRP devient un VRP coopératif axé sur le service (SoC-VRP). D’après la documentation disponible, le EVRP a attiré l’attention au cours des dernières décennies. Pour planifier une voie d’urgence.

Dans ce contexte les travaux menés dans ce mémoire portent sur la présentation d'un algorithme de recuit simulé (SA) qui intègre plusieurs techniques de recherche locales pour résoudre ce problème. Les expériences sur des exemples de littérature montrent que notre (SA) est rapide et surpasse les approches existantes. À notre connaissance, c’est la première fois que la recherche locale dédiée. Nous présentons un algorithme CNN (réseau de neurones conventionnel) pour détecter les modèles de congestion du trafic, les groupements de clients, les zones à faible demande, ces informations peuvent être utilisées pour générer des itinéraires et des temps de déplacement et des allocations de ressources. Ce modèle permet de classer les types de véhicules en fonction de leurs contraintes de capacité. Nous proposons une architecture multi-agents pour garantir l’autonomie de notre programme, la flexibilité et l’adaptabilité, le parallélisme et l’efficacité. Pour réduire le temps d’exécution du programme en parallélisant les opérations, nous utiliserons CUDA Framework. Un GPU peut grandement améliorer les performances en effectuant toutes ces opérations simultanément.

# **L’état de l’art**

# **Introduction**

Ce premier chapitre situe le travail sur comment l’apprentissage automatique (ou les modèles d’apprentissage profond) et le réseau neuronal aident au routage dans le réseau VRP. L’objectif de ce mémoire est la résolution d’une variante du problème de routage du véhicule (VRP) dans laquelle une fenêtre de temps est associée à chaque service client et certains services nécessitent des visites simultanées à différents véhicules pour être effectués. Le problème est donc appelé VRP avec des fenêtres de temps et des contraintes de synchronisation.

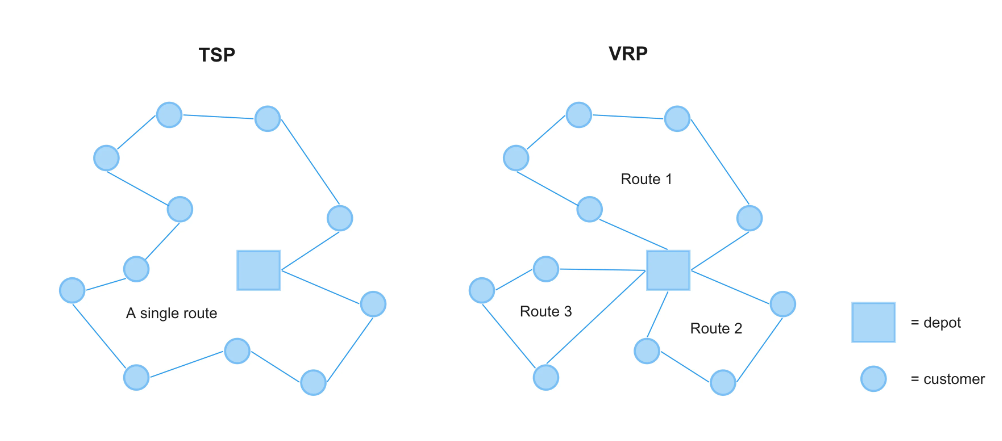
# **C’est quoi le VRP ?**

La plupart des gens ont une compréhension de base du problème de routage du véhicule. Essentiellement, il s’agit de déterminer les itinéraires optimaux d’un dépôt à un ensemble de clients géographiquement dispersés afin de répondre aux exigences connues des clients. L’objectif principal du VRP est de minimiser les coûts de distribution, mais la complexité de la résolution de ce problème est souvent sous-estimée.

Le VRP n’est pas un phénomène nouveau puisqu’il remonte au milieu du XXe siècle. Il est apparu pour la première fois dans un article écrit par George Dantzig et John Ramser en 1959, dans lequel ce problème algorithmique a été appliqué pour la première fois dans le contexte de la livraison d’essence. À l’époque, les mathématiciens appliquaient des algorithmes d’optimisation d’itinéraires calculés manuellement pour trouver les itinéraires les plus optimaux pour un ensemble de véhicules. De nos jours, la technologie fournit un soutien considérable dans ce domaine sous la forme d’un logiciel d’optimisation des itinéraires. Grâce à des algorithmes à grande vitesse, les prestataires logistiques sont en mesure d’optimiser leur VRP en quelques minutes.

## Problème de voyageur de commerce (TSP)

Un exemple classique et bien connu d’un VRP est le problème des Voyageur de commerce (TSP). TSP se préoccupe de trouver le chemin le plus court qu’un vendeur ou un véhicule peut prendre, compte tenu d’une liste de destinations spécifiques qui ne peuvent être visitées qu’une seule fois. Remarquez que lorsqu’on parle de TSP, il n’y a qu’un seul véhicule dans l’équation. Dès que plusieurs véhicules entrent en jeu, cela devient un VRP. Ceci est illustré ci-dessous.



**Figure 2 :** TSP vs VRP

**Figure 2 :** TSP vs VRP



**Figure 3 :** TSP pose la question suivante : Étant donné une liste de villes et les distances entre chaque paire de villes, quel est le chemin le plus court possible qu’un vendeur peut prendre pour visiter chaque ville et retourner à la ville d’origine?

## Pourquoi est-ce un tel défi de résoudre le VRP ?

**Capacité:**  
Chaque véhicule a une capacité de transport, ce qui signifie qu’il ne peut transporter qu’un certain nombre d’articles à la fois, sans dépasser le volume et le poids seuil. L’objectif est de créer un itinéraire qui permet à un véhicule de ramasser / livrer la quantité maximale au coût le plus bas dans la capacité disponible. Le niveau de complexité peut varier considérablement d’une entreprise à l’autre. Alors qu’un fournisseur de logistique générale pouvait transporter des marchandises allant de la nourriture à l’électronique et au mobilier, un détaillant de mode ne transporte généralement que des vêtements. Pour le fournisseur logistique, le même camion peut avoir des spécifications de capacité différentes selon le type de marchandise transportée. Le détaillant de mode, d’autre part, peut n’avoir qu’une seule catégorie, ce qui rend les spécifications de capacité beaucoup plus simplifiées.  
  
**Fenêtres temporelles:**  
Les fenêtres de temps indiquent quand chaque visite peut être effectuée et ajoutent donc une autre couche de complexité au puzzle. Chaque client et dépôt ont souvent leurs propres heures d’ouverture spécifiques qui varient tout au long de la semaine, qui doivent toutes être prises en compte lors de la résolution d’un VRP réel.  
  
**Ramassage et livraison**:  
Un VRP de ramassage et de livraison survient lorsqu’un véhicule doit ramasser des marchandises ou des personnes à divers endroits et les déposer à plusieurs points de destination. Le principal défi consiste à combiner de manière optimale le transport de toutes les marchandises (ou personnes) tout en minimisant la longueur totale de l’itinéraire. Dans la plupart des cas, les ramassages et les livraisons se font de façon séquentielle, ce qui signifie qu’il n’y a pas de poste de facteurs en cause.

**Contraintes de ressources:**

Les ressources sont limitées, mais très nécessaires pour la livraison des marchandises. Votre problème de routage de véhicule devra donc toujours prendre en compte le nombre de conducteurs / véhicules disponibles et les heures de travail, afin de parvenir à une solution réaliste.

**Conclusion :**

En conclusion, le VRP est un problème algorithmique extrêmement complexe qui a fait l’objet de nombreuses études depuis plus de 60 ans. Bien que ce ne soit pas un phénomène nouveau, trouver une solution optimale reste un défi. Les réseaux de distribution d’aujourd’hui sont très complexes et les prix du carburant montent en flèche, ce qui rend essentiel pour les entreprises de rechercher des moyens plus efficaces de résoudre le VRP. Le logiciel d’optimisation des itinéraires offre aux fournisseurs logistiques d’aujourd’hui la possibilité d’utiliser leurs ressources au maximum, ce qui leur permet d’économiser du temps, de l’argent et des émissions inutiles.

# **2 IA, l'apprentissage par machine et le Deep Learning**

## Introduction

L'intelligence artificielle (IA) est un domaine de l'informatique qui se concentre sur le développement de systèmes capables d'exécuter des tâches nécessitant généralement l'intelligence humaine. Parmi les techniques d'IA, on retrouve l'apprentissage automatique, également connu sous le nom de machine learning, qui consiste à permettre aux ordinateurs d'apprendre à partir de données sans être explicitement programmés pour chaque tâche.

Le Deep Learning, une branche de l'apprentissage automatique, repose sur l'utilisation de réseaux de neurones artificiels pour apprendre et effectuer des tâches complexes. Ces réseaux de neurones sont structurés en couches successives, permettant ainsi une représentation hiérarchique des données et la capture de motifs ou de caractéristiques abstraites.

L'évolution rapide de ces domaines a entraîné des avancées significatives dans divers secteurs tels que la reconnaissance d'images, la traduction automatique, la prise de décision autonome, et même la résolution de problèmes complexes tels que l'optimisation des itinéraires de véhicules. L'IA, le machine learning et le Deep Learning continuent de susciter un intérêt croissant en raison de leur potentiel à transformer de nombreux aspects de notre vie quotidienne et de divers domaines industriels.

Avant de vous plonger dans le fonctionnement des CNN, il est important de comprendre comment ces algorithmes de deep learning se rapportent au domaine plus large de l'IA et les distinctions entre les termes clés couramment utilisés dans ce domaine.

* **L'intelligence artificielle** **:** le domaine de l'informatique axé sur les programmes informatiques intelligents capables de sentir, de raisonner, d'agir et de s'adapter.
* **L'apprentissage par machine** **:** un sous-ensemble de l'IA dans lequel les algorithmes peuvent améliorer en performance au fil du temps lorsqu'ils sont mis en présence de plus de données.
* **Le réseau neuronal** **:** une série d'algorithmes utilisés comme processus dans l'apprentissage par machine qui peut reconnaître les signaux et des relations dans de grandes quantités de données. Les réseaux neuronaux utilisent une structure logique inspirée du cerveau humain et constituent la base des algorithmes pour le deep learning.
* **Le Deep learning :**un sous-domaine de l'apprentissage par machine dans lequel les réseaux neuronaux multicouches apprennent à partir de grandes quantités de données.

## Introduction au réseau de neurones à convolution

Un réseau de neurones convolutionnel (CNN) est un type d’architecture de réseau de neurones à apprentissage profond couramment utilisé en vision par ordinateur. La vision par ordinateur est un domaine de l’intelligence artificielle qui permet à un ordinateur de comprendre et d’interpréter les données visuelles ou d’image.

En ce qui concerne le Machine Learning, les réseaux de neurones artificiels fonctionnent très bien. Les réseaux

dans cette couche est égal au nombre total de caractéristiques dans nos données (nombre de pixels dans le cas d’une neuronaux sont utilisés dans divers ensembles de données comme les images, l’audio et le texte. Différents types de réseaux de neurones sont utilisés à différentes fins, par exemple pour prédire la séquence de mots que nous utilisons Réseaux de neurones récurrents plus précisément un LSTM, de même pour la classification des images, nous utilisons des réseaux de neurones de convolution. Dans ce blog, nous allons construire un bloc de base pour CNN.

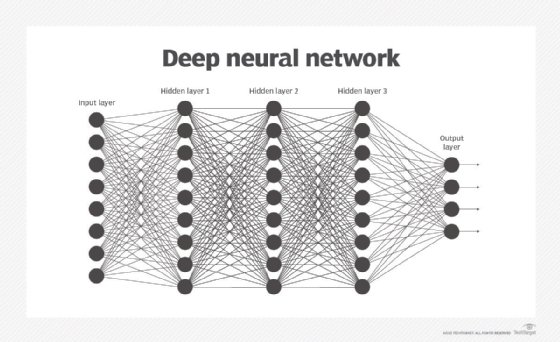
Dans un réseau neuronal ordinaire, il existe trois types de couches :

**1-Couches d’entrée :** C’est la couche dans laquelle nous donnons une entrée à notre modèle. Le nombre de neurones image).

**2-Couche cachée:** L’entrée de la couche d’entrée est ensuite introduite dans la couche cachée. Il peut y avoir de nombreuses couches cachées en fonction de notre modèle et de la taille des données. Chaque couche cachée peut avoir des nombres différents de neurones qui sont généralement plus grands que le nombre de caractéristiques. La sortie de chaque couche est calculée par multiplication matricielle de la sortie de la couche précédente avec des poids pouvant être définis pour cette couche, puis par l’ajout de biais exploitables suivis d’une fonction d’activation qui rend le réseau non linéaire.

**3-Couche de sortie:** La sortie de la couche cachée est ensuite introduite dans une fonction logistique comme sigmoid ou softmax qui convertit la sortie de chaque classe dans le score de probabilité de chaque classe.

Les données sont introduites dans le modèle et la sortie de chaque couche est obtenue à partir de l’étape ci-dessus est appelée **feedforward**, nous calculons ensuite l’erreur en utilisant une fonction d’erreur, certaines fonctions d’erreur courantes sont **cross-entropy**, erreur de perte carrée, etc. La fonction d’erreur mesure la performance du réseau. Après cela, nous nous propageons dans le modèle en calculant les dérivés. Cette étape est appelée **Backpropagation** qui est essentiellement utilisée pour minimiser la perte.

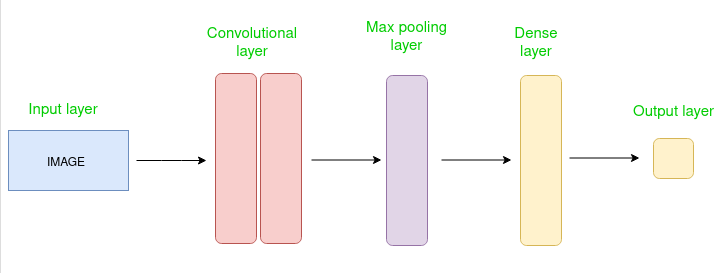


**Figure 4 :** Diagramme illustrant comment un réseau de neurones traite les données.

## 2.2 Convolution Neural Network

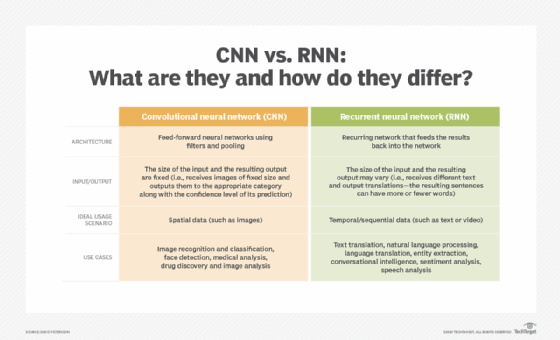
Le réseau de neurones convolutionnel (CNN) est la version étendue des réseaux de neurones artificiels (ANN) qui est principalement utilisée pour extraire la caractéristique de l’ensemble de données matricielles de type grille. Par exemple, des jeux de données visuels comme des images ou des vidéos où les modèles de données jouent un rôle important.

### **Architecture CNN**



**Figure 5 :** Architecture CNN simple

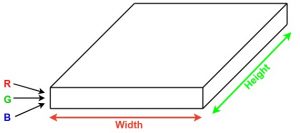
Le calque convolutionnel applique des filtres à l’image d’entrée pour extraire des fonctionnalités, le calque de regroupement échantillonne l’image pour réduire le calcul, et le calque entièrement connecté effectue la prédiction finale. Le réseau apprend les filtres optimaux par rétropropagation et descente de gradient.



**Figure 6 :** Tableau illustrant les différences entre un réseau de neurones convolutifs et un réseau de neurones récurrents.

### **Fonctionnement des couches convolutives**

Les réseaux de neurones à convolution ou convNets sont des réseaux de neurones qui partagent leurs paramètres. Imaginez que vous avez une image. Il peut être représenté comme un cuboïde ayant sa longueur, sa largeur (dimension de l’image) et sa hauteur (c’est-à-dire le canal car les images ont généralement des canaux rouges, verts et bleus).



**Figure 7 :** dimension d’image (width & height)

Maintenant, imaginez prendre un petit patch de cette image et exécuter un petit réseau de neurones, appelé un filtre ou un noyau, avec, disons, des sorties K et les représenter verticalement. Maintenant, faites glisser ce réseau de neurones sur l’ensemble de l’image, par conséquent, nous obtiendrons une autre image avec des largeurs, des hauteurs et des profondeurs différentes. Au lieu des canaux R, G et B, nous avons maintenant plus de canaux, mais moins de largeur et de hauteur. Cette opération s’appelle Convolution. Si la taille du patch est la même que celle de l’image, ce sera un réseau de neurones régulier. En raison de ce petit patch, nous avons moins de poids.



**Figure 8 :** dimension d’image (width & height)

Maintenant, parlons un peu de mathématiques qui sont impliqué dans tout le processus de convolution.

* Les couches de convolution se composent d’un ensemble de filtres (ou de noyaux) pouvant être mémorisés ayant de petites largeurs et hauteurs et la même profondeur que celle du volume d’entrée (3 si la couche d’entrée est entrée image).
* Par exemple, si nous devons exécuter la convolution sur une image avec des dimensions 34x34x3. La taille possible des filtres peut être axax3, où « a » peut être n’importe quoi comme 3, 5 ou 7 mais plus petit par rapport à la dimension de l’image.
* Au cours de la passe avant, nous faisons glisser chaque filtre sur l’ensemble du volume d’entrée étape par étape où chaque étape est appelée foulée (qui peut avoir une valeur de 2, 3 ou même 4 pour les images de grande dimension) et calculons le produit de points entre les poids du noyau et le patch du volume d’entrée.
* En glissant nos filtres, nous obtiendrons une sortie 2-D pour chaque filtre et nous les empilerons en conséquence, nous obtiendrons un volume de sortie ayant une profondeur égale au nombre de filtres. Le réseau apprendra tous les filtres.

### **Couches utilisées pour construire ConvNets**

Une architecture complète de réseaux de neurones à convolution est également connue sous le nom de covnets. Un covnets est une séquence de couches, et chaque couche transforme un volume à un autre grâce à une fonction différenciable.

Types de couches : jeux de données

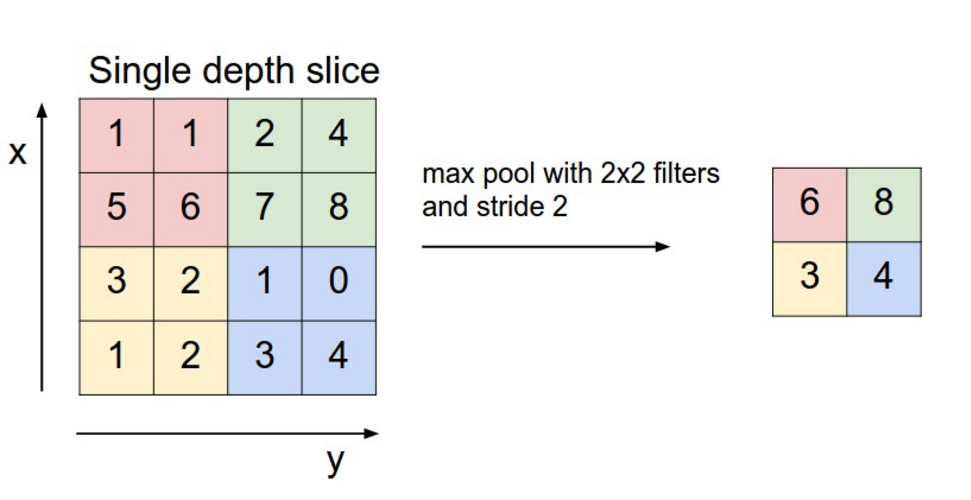
Prenons un exemple en exécutant un covnets sur une image de dimension 32 x 32 x 3.

**Couches d’entrée :** C’est la couche dans laquelle nous donnons l’entrée à notre modèle. Dans CNN, généralement, l’entrée sera une image ou une séquence d’images. Ce calque contient l’entrée brute de l’image avec la largeur 32, la hauteur 32 et la profondeur 3.

**Couches convolutionnelles** : Il s’agit de la couche utilisée pour extraire la fonctionnalité du jeu de données d’entrée. Il applique aux images d’entrée un ensemble de filtres lisibles appelés noyaux. Les filtres / noyaux sont des matrices plus petites, généralement de forme 2 2, 3 3 ou 5 5. Il glisse sur les données d’image d’entrée et calcule le produit de points entre le poids du noyau et le patch d’image d’entrée correspondant. La sortie de cette couche est référée aux cartes d’entités publicitaires. Supposons que nous utilisions un total de 12 filtres pour cette couche, nous obtiendrons un volume de sortie de dimension 32 x 32 x 12.

**Couche d’activation :** En ajoutant une fonction d’activation à la sortie de la couche précédente, les couches d’activation ajoutent une non-linéarité au réseau. il appliquera une fonction d’activation par élément à la sortie de la couche de convolution. Certaines fonctions d’activation courantes sont RELU : max (0, x), **Tanh, Leaky** **RELU,** etc. Le volume reste inchangé, donc le volume de sortie aura des dimensions 32 x 32 x 12.

**Couche de mise en commun (Pooling layer) :** Cette couche est périodiquement insérée dans les **covnets** et sa fonction principale est de réduire la taille du volume, ce qui rend le calcul rapide réduit la mémoire et empêche également le sur-ajustement. Deux types courants de couches de mise en commun sont la mise en commun maximale et la mise en commun moyenne. Si nous utilisons une piscine max avec 2 x 2 filtres et stride 2, le volume résultant sera de dimension 16x16x12.

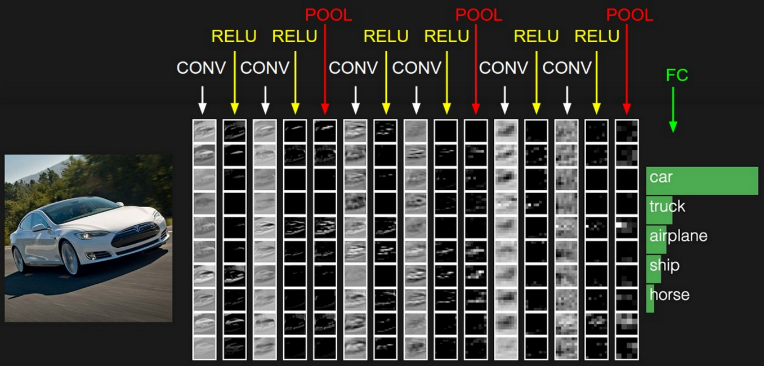


**Figure 9 : pooling layer**

**Source image:** *cs231n.stanford.edu*

**Flattening:** The resulting feature maps are flattened into a one-dimensional vector after the convolution and pooling layers so they can be passed into a completely linked layer for categorization or regression.

**Fully Connected Layers:** It takes the input from the previous layer and computes the final classification or regression task.



**Figure 10 :** couche de sortie

**Couche de sortie :** La sortie des couches entièrement connectées est ensuite introduite dans une fonction logistique pour les tâches de classification comme sigmoïde ou softmax qui convertit la sortie de chaque classe dans le score de probabilité de chaque classe.



### **Avantages des réseaux de neurones convolutifs (CNN) :**

1. Bon pour détecter les motifs et les fonctionnalités dans les images, les vidéos et les signaux audio.

2. Robuste à l’invariance de translation, de rotation et d’échelle.

3. Formation de bout en bout, pas besoin d’extraction manuelle des fonctionnalités.

4. Peut gérer de grandes quantités de données et atteindre une grande précision.



### **Inconvénients des réseaux de neurones convolutifs (CNN) :**

1. Computationally cher à former et nécessite beaucoup de mémoire.
2. Peut être sujette au sur-ajustement si pas assez de données ou une régularisation appropriée est utilisée.
3. Nécessite de grandes quantités de données étiquetées.
4. L’interprétabilité est limitée, il est difficile de comprendre ce que le réseau a appris.

**Conclusion**

En conclusionLe Deep Learning et les CNN continueront à faire partie des outils d'IA les plus puissants pour les développeurs et les entreprises à l'avenir. Les entreprises seront toujours amenées à trouver des solutions nouvelles et innovantes pour les aider à relever leurs uniques défis. Nombreux sont celles qui se tourneront vers des technologies reposant sur le deep learning et les CNN, comme la vision par ordinateur, l'IA, la réalité augmentée et la réalité virtuelle, pour trouver des solutions. À mesure que l'IA progresse, Intel s'engage à faire en sorte que les développeurs, les scientifiques des données, les chercheurs et les ingénieurs des données puissent préparer, construire, déployer et mettre à l'échelle leurs solutions d'IA de la manière la plus transparente possible.

# **Algorithme recuit simulé (SA)**

# **Introduction**

Trouver une solution optimale à certains problèmes d’optimisation peut être une tâche incroyablement difficile, souvent pratiquement impossible. En effet, lorsqu’un problème devient suffisamment important, nous devons rechercher un grand nombre de solutions possibles pour trouver la solution optimale. Même avec la puissance de calcul moderne, il y a souvent encore trop de solutions possibles à envisager. Dans ce cas, parce que nous ne pouvons pas raisonnablement nous attendre à trouver le meilleur dans un délai raisonnable, nous devons nous contenter de quelque chose qui est assez proche.

## C’est quoi l’algorithme recuit simulé ?

SA est une technique d’optimisation méta heuristique introduite par **Kirkpatrick** et **al**. en 1983 pour résoudre le problème des vendeurs itinérants (TSP).

L’algorithme SA est basé sur le procédé de recuit utilisé en métallurgie, où un métal est chauffé à haute température rapidement puis refroidi progressivement. À des températures élevées, les atomes se déplacent rapidement, et lorsque la température est réduite, leur énergie cinétique diminue également. À la fin du processus de recuit, les atomes tombent dans un état plus ordonné et le matériau est plus ductile et plus facile à travailler.

De même, en SA, un processus de recherche commence par un état de haute énergie (une solution initiale) et abaisse progressivement la température (un paramètre de contrôle) jusqu’à atteindre un état d’énergie minimale (la solution optimale).

La SA a été appliquée avec succès à un large éventail de problèmes d’optimisation, tels que le TSP, le pliage de protéines, le partitionnement de graphes et la planification d’atelier. Le principal avantage de la SA est sa capacité à s’échapper des minima locaux et à converger vers un minimum global. SA est également relativement facile à mettre en œuvre et ne nécessite pas de connaissance a priori de l’espace de recherche.

Le processus de recuit simulé commence par une solution initiale, puis améliore itérativement la solution actuelle en la perturbant aléatoirement et en acceptant la perturbation avec une certaine probabilité. La probabilité d’accepter une solution pire est initialement élevée et diminue progressivement à mesure que le nombre d’itérations augmente.

L’algorithme SA est assez simple, et il peut être implémenté directement comme décrit ci-dessous.

## Définir le problème

Premièrement, il faut définir le problème pour l’optimiser, c’est-à-dire définir la fonction énergétique, c’est-à-dire la fonction pour minimiser ou maximiser. Par exemple, si nous voulons minimiser une fonction à valeur réelle de deux variables, par exemple, f(x,y) = x 2 + y 2, l’énergie correspond à la fonction f(x,y) elle-même. Dans le cas du PTS, l’énergie liée à une séquence de villes est représentée par la longueur totale du trajet.

Une fois la fonction énergétique définie, nous devons définir la valeur de température initiale et la solution candidate initiale. Ce dernier peut être généré aléatoirement ou en utilisant une autre méthode heuristique. Ensuite, nous calculons l’énergie de la solution candidate initiale.

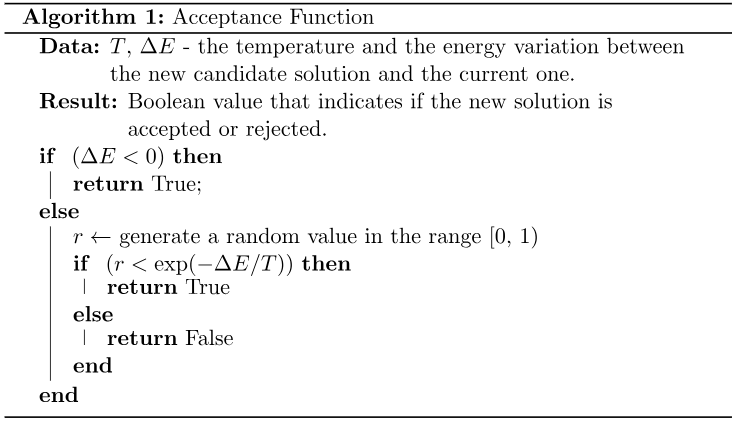
## Définir la fonction de perturbation

Une fonction de perturbation est définie pour générer de nouvelles solutions candidates. Cette fonction devrait générer des solutions proches de la solution actuelle mais pas trop similaires. Par exemple, si nous voulons minimiser une fonction f(x,y), nous pouvons perturber aléatoirement la solution actuelle en ajoutant une valeur aléatoire entre -0,1 et 0,1 à x et y. Dans le cas du TSP, une nouvelle solution candidate peut être générée en échangeant deux villes dans l’ordre de déplacement de la solution actuelle.

## Critère d'acceptation

Le critère d’acceptation détermine si une nouvelle solution est acceptée ou rejetée. L’acceptation dépend de la différence d’énergie entre la nouvelle solution et la solution actuelle, ainsi que de la température actuelle. Le critère d’acceptation classique de SA vient de la mécanique statistique, et il est basé sur la distribution de probabilité de Boltzmann. Un système en équilibre thermique à température T peut être trouvé dans un état à énergie E avec une probabilité proportionnelle à exp (-E / k T) (1) commencer {équation\*} operatorname {Prob}(E) sim exp (-E / k T) fin{équation\*} où k est la constante de Boltzmann. Par conséquent, à basse température, il y a une petite chance que le système soit dans un état de haute énergie. Cela joue un rôle crucial dans l’AS car une augmentation de l’énergie permet d’échapper aux minima locaux et de trouver le minimum global.

Basé sur la distribution de Boltzmann, l’algorithme suivant définit le critère d’acceptation d’une variation d’énergie Delta E à température T.



**Figure 11 :** L’algorithme de la fonction d’acceptation

Une solution candidate avec moins d’énergie est toujours acceptée. À l’inverse, une solution candidate avec une énergie plus élevée est acceptée aléatoirement avec une probabilité exp (- Delta E / T) (pour notre part, nous pouvons définir k = 1). Ce dernier cas peut être mis en œuvre en comparant la probabilité avec une valeur aléatoire générée dans la plage [0, 1).

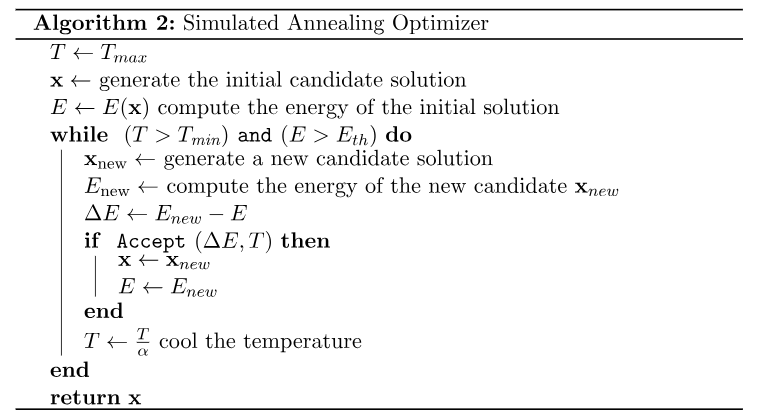
## Calendrier des températures

Le programme de température détermine la façon dont la température du système change au fil du temps. Au début, la température est élevée afin que l’algorithme puisse explorer un large éventail de solutions, même si elles sont pires que la solution actuelle. À mesure que les itérations augmentent, la température diminue progressivement, de sorte que l’algorithme devient plus sélectif et accepte de meilleures solutions avec une probabilité plus élevée. Un ordonnancement simple peut être obtenu en divisant la température actuelle par un facteur alpha, qui est inférieur à 1.

## 3.6 Exécuter l’algorithme SA

Enfin, exécutez l’algorithme en appliquant itérativement la fonction de perturbation et le critère d’acceptation à la solution actuelle. L’algorithme se termine lorsque la température a refroidi à un certain niveau T\_ {min} ou lorsque l’énergie de la solution actuelle est inférieure à un seuil fixe E\_ {th}.

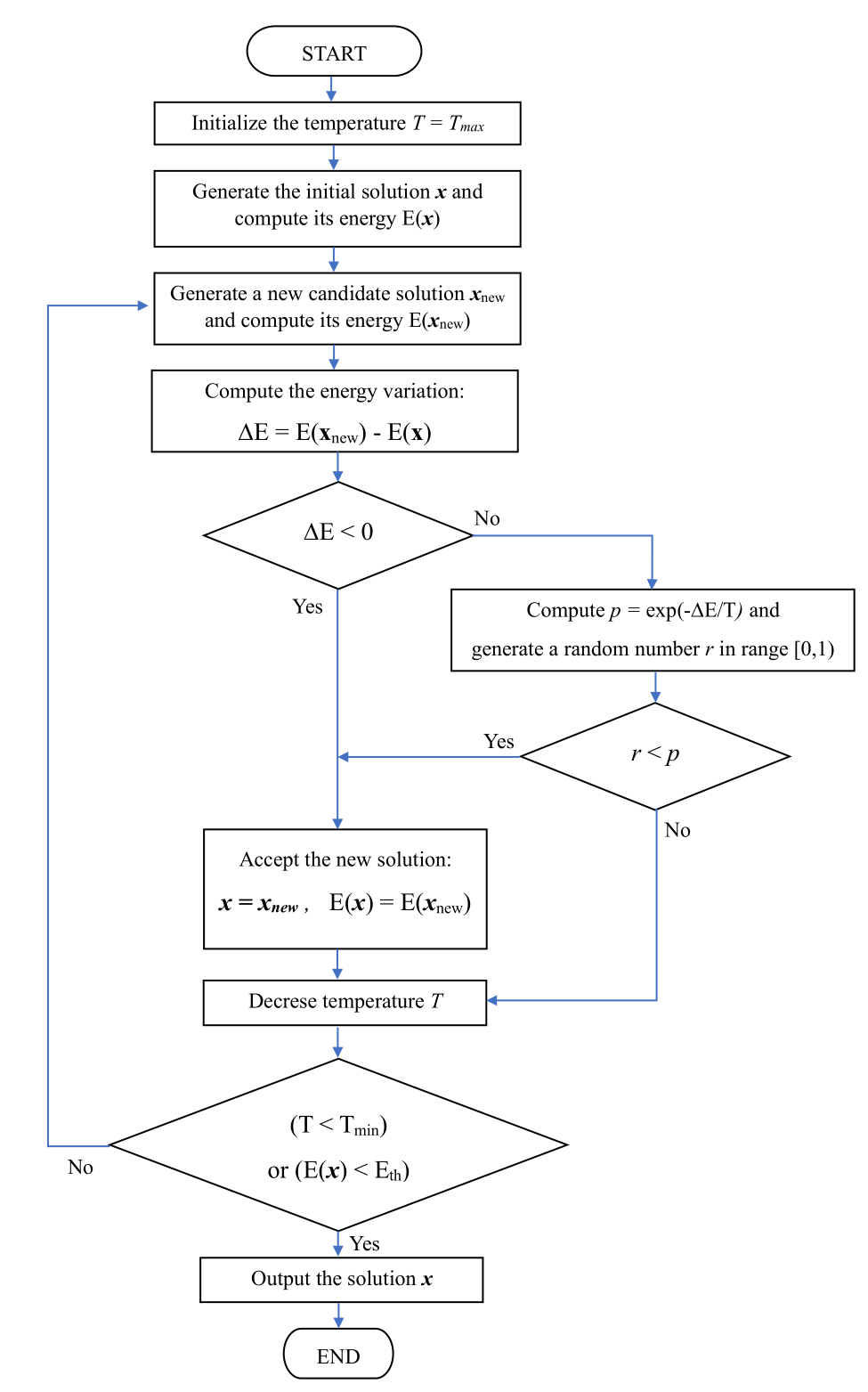
Voici le pseudocode de SA :



**Figure 12 :** l’optimiseur de l’algorithme SA

**Le diagramme de flux de SA**

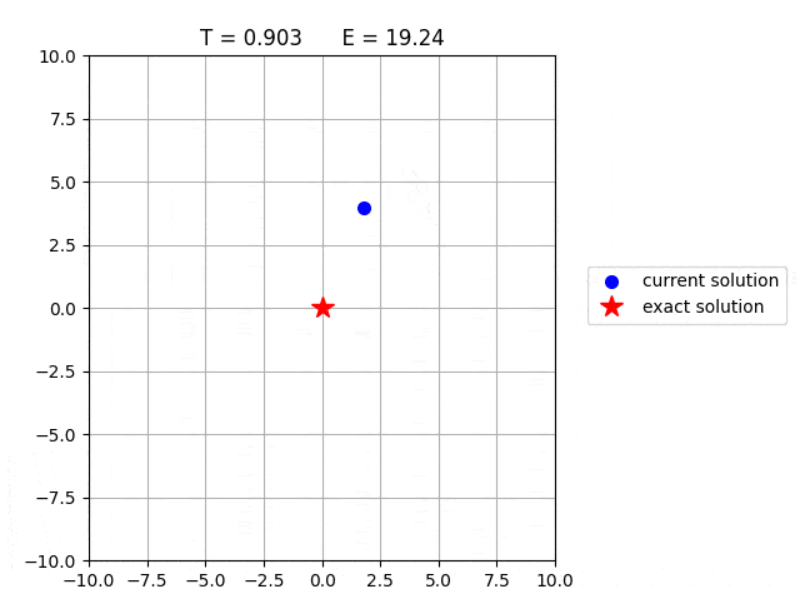
Ici, nous fournissons un organigramme détaillé représentant toutes les étapes de l’AS :



**Figure 13 :** le diagramme de flux de l’algorithme recuit simulé

## 3.7 Exemple d’application

Pour mieux comprendre l’algorithme, nous utilisons SA pour illustrer la minimisation de la fonction f(x,y) = x 2 + y 2. Nous avons utilisé comme espace de recherche une grille de taille 101 fois 101 placée dans la zone carrée définie par (x,y) en [-10, 10] fois [-10, 10]. Nous fixons la vitesse de refroidissement alpha=0,84 et la solution initiale (x,y)=(4,4). À chaque étape, une nouvelle solution est générée en déplaçant aléatoirement la solution actuelle par pm 0,2 dans x et y direction.



**Figure 14 :** montrant la solution candidate, son énergie et la température à chaque étape

Nous pouvons observer que des solutions pires sont fréquemment acceptées lorsque la température est élevée. Inversement, lorsque la température est basse (par exemple, T<1), l’algorithme est plus sélectif et de meilleures solutions sont acceptées avec une probabilité plus élevée.

# **Conclusion**

Dans cette partie, nous avons donné un aperçu de l’algorithme SA. Nous avons illustré la procédure d’optimisation et fourni un exemple pratique de son application.

# **Système multi agent**

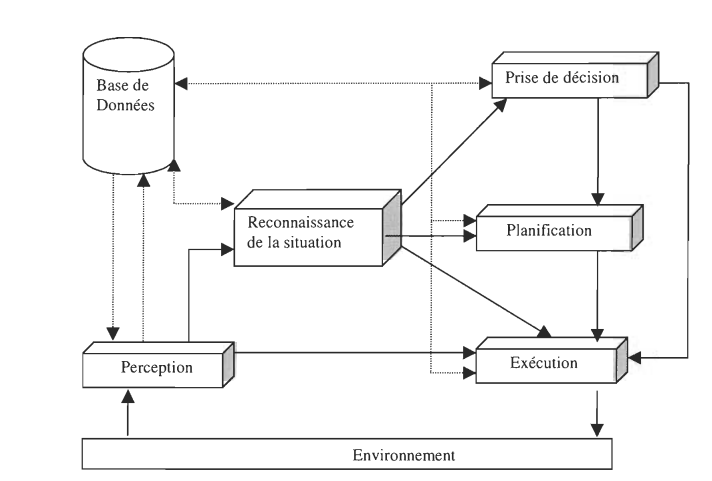
# **Introduction**

Le thème des systèmes multi agents (SMAs), s’il n’est pas récent, est actuellement un champ de recherche très actif. Cette discipline est à la connexion de plusieurs domaines en particulier de l’intelligence artificielle, des systèmes informatique distribués et du génie logiciel. C’est une discipline qui s’intéresse aux comportements collectifs produits par les interactions de plusieurs entités autonomes et flexibles appelées agents, que ces interactions tournent autour de la coopération, de la concurrence ou de la coexistence entre ces agents. Ce chapitre introduit, tout d’abord, les notions d’agents et de systèmes multi agents (SMAs), et détaille par la suite les différentes questions que soulèvent la problématique des SMAs, en particulier : les interactions et la coopération, la coordination, la planification et la communication. À la fin, un aperçu des différentes applications des SMAs est donné.

## 4.1 Agent

Un agent est un système informatique encapsulé situé dans un environnement dans lequel il est capable d'effectuer une action flexible et autonome, compatible aux objectifs de la conception. Les agents sont: des entités clairement identifiables de résolution de problèmes avec des bornes et des interfaces bien définies; situés dans un environnement particulier; ils reçoivent des entrées liées aux états de cet environnement par des capteurs et agissent sur cet environnement par des émetteurs; destinés à atteindre un objectif spécifique; autonomes et responsables de leur comportement; capables d'adopter un comportement flexible pour résoudre des problèmes selon les objectifs de la conception; ils sont réactifs (capables de s'adapter aux changements d'état de leur environnement) et proactifs (capables d'adopter un nouvel objectif); capables dans un univers multi-agents, de communiquer, coopérer, se coordonner, négocier les uns avec les autres.

La figure 1.1 donne, de façon générale, l'architecture interne d'un agent.

****

**Figure 15 :** Architecture d’agent

Lorsqu'un agent perçoit une situation dans l'environnement, il essaie de la reconnaître. Si la situation lui est familière, il peut enclencher un processus de planification afin de résoudre le problème. il peut aussi reconnaître la situation en terme d'action et donc, passe à l'exécution de la tâche (Reconnaissance- Exécution). Lorsque l'agent perçoit des situations qu'il connaît très bien, il peut faire intervenir son comportement réactif en passant directement à l'action (Perception-Exécution). S'il ne peut pas résoudre un problème (situation non-familière), il engage un processus de coopération pour demander de l'aide aux autres agents (Reconnaissance Prise de décisions).

## Systèmes Multi-agent

Les systèmes multi-agents mettent en œuvre des agents homogènes et hétérogènes ayant des buts communs ou distincts. Ils sont dynamiques. Un système multi-agent est un système distribué composé d'un ensemble d'agents qui interagissent le plus souvent, selon des modes de coopération, de concurrence ou de coexistence.

Un SMA est généralement caractérisé par :

1. chaque agent a des informations ou des capacités de résolution de problèmes limitées, ainsi chaque agent à un point de vue partiel.

2. il n'y aucun contrôle global du système multi-agents.

3. les données sont décentralisées.

4. le calcul est asynchrone.

## Des Objets aux Agents

D’un point de vue informatique, l’approche multi-agent peut être considérée comme une évolution du paradigme orienté-objet. Du point de vue conceptuel, un objet est simplement une structure de données à laquelle sont associées des fonctions. Les agents sont des entités autonomes, ce qui signifie que leur comportement ne dépend pas d’une pression extérieure, contrairement aux objets. -Agent: entité autonome interagissant avec son environnement -Objet: entité passive possédant un état et sur lequel on peut effectuer des opérations. (Chaib-draa, 2010) -Un agent est à un degré d’abstraction plus élevé qu’un objet. Un agent peut être constitué de plusieurs objets. -C’est un paradigme de programmation mettant en évidence l’autonomie et les interactions. (Programmation orientée-agent)

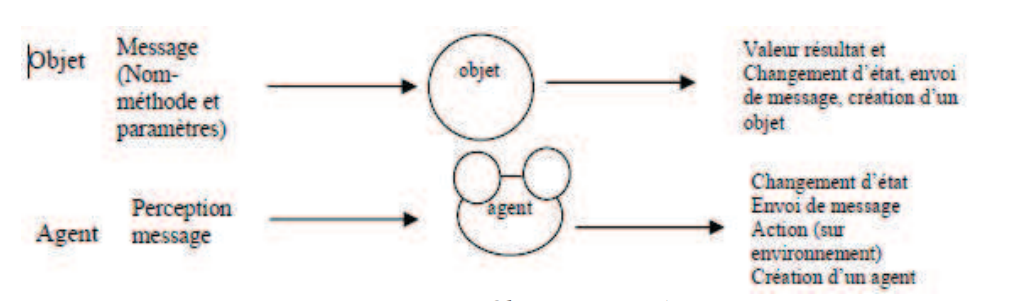
**Similarités** :

-Possèdent un «état interne»,

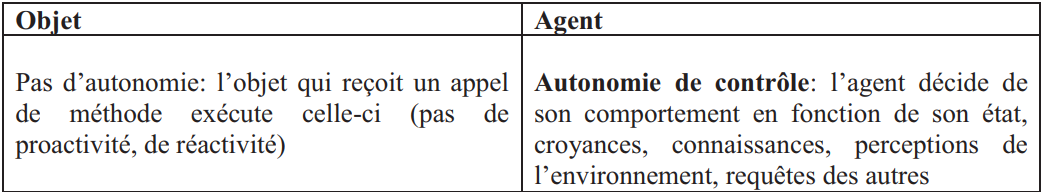
-Des unités de comportement modulaires (méthodes/compétences).

-Communiquent par envoi de messages,.

-Peuvent agir pour modifier leur état.

****

**Figure 16 :** Objet «versus» Agent

****

**Figure 17 :** Différence entre un agent et objet

**Utilité des systèmes multi agents**

Les systèmes multi-agents sont extrêmement adaptés à des environnements nécessitant l'interaction de multiples entités, notamment dans des contextes distribués tels que la coordination entre des frégates, la gestion du trafic aérien ou les bases de données coopératives. Ces domaines, intrinsèquement distribués, trouvent une solution aisée et efficace à leur modélisation à travers les systèmes multi-agents.

Une autre situation propice à l'utilisation des systèmes multi-agents se présente lorsque diverses entités et leurs données appartiennent à des organisations distinctes désirant préserver la confidentialité de leurs informations pour des raisons concurrentielles. Par exemple, dans les missions maritimes impliquant plusieurs pays, plusieurs bateaux de nations différentes doivent collaborer. Cependant, chaque nation souhaite garder confidentielles certaines informations, considérées comme des secrets militaires. De ce fait, la conception d'un système coordonnant ces bateaux exigerait une représentation précise des objectifs et des intérêts de chaque pays à travers la création d'agents propres à chacun. Ces agents pourraient ensuite communiquer entre eux pour coordonner la mission, tout en échangeant uniquement les informations nécessaires à cette coordination, préservant ainsi la confidentialité des données sensibles.

Même dans des domaines où l'utilisation de systèmes multi-agents n'est pas une nécessité absolue, leur utilisation comporte des avantages indéniables. Ces systèmes sont particulièrement adaptés à la résolution de problèmes offrant diverses méthodes, perspectives ou résolutions. Ils trouvent une application efficace dans la modélisation du raisonnement humain au sein de simulations complexes, comme celles des combats aériens. En outre, ils présentent des avantages inhérents à la résolution distribuée et concurrente de problèmes, notamment :

La modularité, simplifiant la programmation et permettant l'extensibilité du système en facilitant l'ajout de nouveaux agents par rapport à l'ajout de nouvelles capacités à un système monolithique.

La vitesse, grâce au parallélisme offert par la possibilité de travailler simultanément sur un problème par plusieurs agents.

La fiabilité, résultant du partage de contrôle et de responsabilités entre les différents agents, permettant au système de tolérer la défaillance d'une ou plusieurs entités sans compromettre l'ensemble du système.

## Propriétés d’un agent intelligent

Un agent peut être caractérisé par plusieurs propriétés, nous pouvons citer parmi elles :

1. **Autonomies :**

Un agent autonome possède la capacité d'opérer de manière indépendante, sans une intervention directe d'humains ou d'autres agents, tout en contrôlant ses actions et son état interne. Il est caractérisé par la poursuite de ses propres objectifs et la capacité à prendre des décisions relatives à ces objectifs, y compris la résolution de conflits internes.

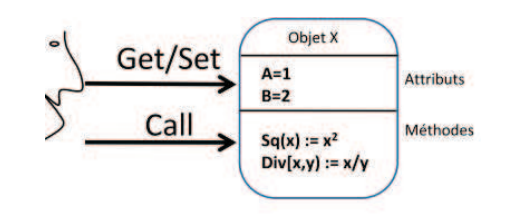
En tant qu'entité autonome, un tel agent est capable de :

1. Agir conformément à ses intentions, agissant de manière délibérée vers des objectifs spécifiques.
2. Adopter des objectifs qu'il estime réalisables tout en abandonnant ceux qu'il perçoit comme impossibles à atteindre.
3. Planifier ses propres actions tout en tenant compte des actions des autres agents ou entités.
4. Raisonner sur les différentes possibilités pour atteindre ses objectifs et résoudre les éventuels conflits internes concernant les objectifs sélectionnés.
5. **Réactivité :**

Les agents sont capables de percevoir leur environnement et d'ajuster leurs actions en réponse aux changements survenant dans cet environnement.

La réactivité des agents implique également leur capacité à adapter leur comportement en fonction des modifications des conditions environnementales.

En revanche, un objet est une entité passive, voire réactive, dans le sens où si aucune demande n'est faite pour obtenir la valeur d'un attribut ou pour activer une méthode spécifique de cet objet, celui-ci demeure inerte et ne réalise aucune action. Autrement dit, il ne réagit pas ou n'agit pas de manière autonome sans qu'une demande extérieure ne soit formulée.

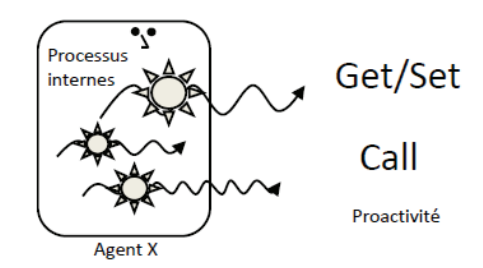


**Figure 18 :** Réactivité

1. **Proactivité**

Les agents ne se contentent pas de réagir à leur environnement ; ils démontrent la capacité de manifester un comportement orienté par des objectifs et d'initier des actions. La proactivité se définit par la faculté d'un agent à anticiper des situations et à modifier son comportement pour les prévenir.

En plus des attributs et des méthodes, un agent possède des processus internes qui demeurent actifs même en l'absence de sollicitations externes. Par conséquent, un agent est capable d'agir de manière autonome et proactive, sans qu'une demande externe ne soit nécessaire pour déclencher son action.



**Figure 19 :** Proactivité

1. **Adaptabilité**

Face à un environnement constamment changeant, un agent doit constamment ajuster son plan d'action pour atteindre ses objectifs. À cette fin :

1. L'agent doit procéder à une évaluation continue et à une perception continue de la situation (le contexte) dans lequel il agit.
2. Il doit être capable de construire des représentations en cours d'exécution (c'est-à-dire apprendre en cours de fonctionnement).
3. Il doit développer des plans dynamiques qui initient ou arrêtent des processus internes en réponse aux changements de l'environnement.

L'adaptabilité se définit comme la capacité d'un agent à s'ajuster à l'environnement dans lequel il opère. Un agent adaptatif est donc capable de moduler ses compétences (communicationnelles et comportementales) en fonction de l'environnement dans lequel il évolue et des autres agents avec lesquels il interagit.

Dans un environnement dynamique, l'adaptabilité représente une propriété cruciale pour un agent, lui permettant de s'ajuster et de réagir de manière appropriée aux changements continus de son environnement.

1. **Sociabilité**

La sociabilité d'un agent désigne sa capacité à s'intégrer efficacement dans un environnement où plusieurs agents interagissent en échangeant des messages dans le but de réaliser un objectif commun.

Cette caractéristique sera approfondie ultérieurement lorsque nous aborderons les propriétés d'un Système Multi-Agent (SMA), car elle englobe d'autres propriétés telles que la communication, la coopération et la délégation.

1. **Apprentissage**

L'apprentissage est une propriété singulière, car tous les agents ne sont pas nécessairement dotés de cette capacité. Les agents peuvent avoir besoin d'apprendre lorsqu'ils interagissent ou réagissent à leur environnement externe.

Cette caractéristique confère aux systèmes la faculté d'acquérir une compréhension des comportements au fil du temps, sans nécessiter une programmation manuelle de ces comportements.

La capacité d'apprentissage d'un agent se manifeste par sa capacité à utiliser de nouvelles connaissances pour ajuster son comportement.

Bien que l'apprentissage soit un élément essentiel de l'intelligence, il est rare de trouver des agents dotés de cette capacité.

1. **Sécurité**

La sécurité revêt une importance capitale, particulièrement dans le cadre de cette étude, car elle assure qu'en interagissant avec un agent, celui-ci n'a pas été altéré par un virus, des informations erronées, ou des connaissances dépourvues de sens

## Propriétés des systèmes multi-agents

**1. interaction agents**

Les agents collaborent pour atteindre un objectif commun. Contrairement à la compétition où les agents cherchent à maximiser leur propre satisfaction, dans la coopération, ils travaillent ensemble pour atteindre un but partagé. Cela implique une communication plus étroite et des interactions plus complexes entre les agents.

La coopération entre agents implique généralement un partage d'informations, une coordination des actions et une répartition des tâches pour atteindre l'objectif commun de manière efficace. Les agents doivent souvent négocier non seulement sur la manière d'atteindre l'objectif, mais aussi sur la manière de collaborer et de partager les ressources nécessaires.

Les systèmes multi agents offrent ainsi une flexibilité en permettant aux agents d'adopter différents modes d'interaction en fonction de la situation. Que ce soit pour simplement coexister, entrer en compétition pour des ressources limitées ou coopérer pour atteindre des objectifs communs, ces systèmes offrent des mécanismes pour gérer ces interactions de manière sophistiquée.

Dans un système où les agents coopèrent, leur objectif n'est plus seulement la maximisation de leur propre satisfaction, mais également de contribuer au succès du groupe dans l'accomplissement d'une tâche ou de la résolution d'un problème commun. Dans ce contexte, la communication entre les agents revêt une importance cruciale.

Les échanges entre les agents consistent souvent en des messages variés, sophistiqués et adaptés à la situation. Ces messages servent à améliorer la performance collective en facilitant la coordination des actions. Les agents peuvent partager des informations sur l'environnement pour enrichir leurs propres perceptions individuelles. De même, ils peuvent communiquer leurs intentions pour permettre aux autres agents de comprendre leurs actions et leurs objectifs.

1. **Coopération**

L'ajout d'un nouvel agent doit conduire à une augmentation significative des performances du groupe.

Les actions des agents doivent être orientées vers l'évitement ou la résolution de conflits potentiels ou réels.

La coopération entre agents peut être catégorisée selon deux approches principales :

1. **Coopération implicite :** Dans ce cas, les agents poursuivent un objectif commun implicite en exécutant des actions indépendantes. La communication entre les agents n'est pas nécessaire, elle reste facultative. Chaque agent travaille vers un but global sans nécessairement échanger d'informations avec les autres agents.
2. **Coopération explicite :** Dans cette forme de coopération, les agents exécutent des actions qui non seulement servent à atteindre leurs propres objectifs mais également contribuent à atteindre les objectifs des autres agents. Ce type de coopération exige une communication active entre les agents afin de comprendre les objectifs mutuels et coordonner leurs actions en conséquence.

La coopération entre agents peut se faire de manière implicite, où les agents agissent indépendamment vers un objectif commun sans communication obligatoire, ou de manière explicite, où les actions des agents sont coordonnées de façon à accomplir à la fois leurs objectifs individuels et ceux des autres agents, nécessitant ainsi une communication pour échanger des informations et s'aligner sur des buts communs.

1. **Coordination**

Dans un système multi-agents, la coordination des actions des différents agents permet d’assurer une cohérence du système. Il existe plusieurs mécanismes de coordinations parmi lesquels, nous retrouvons : l’organisation, la planification et la synchronisation.

1. **Organisation**

L'organisation dans le contexte des systèmes multi-agents représente un groupe d'agents travaillant ensemble pour accomplir des tâches, comme défini par Guessoum (1996). Différents travaux ont tenté de définir et de modéliser cette organisation. T. Bouron (Bourron, 1992) a catégorisé ces travaux en deux classes décrivant la nature de l'organisation :

Structure externe aux agents : Cette vision représente l'organisation comme une entité distincte des agents individuels, représentée par un objet ou un agent global.

Objet abstrait distribué parmi les membres de l’organisation : Cette approche considère l'organisation comme un concept abstrait dont la représentation est distribuée parmi les membres individuels de l'organisation.

Parmi les travaux existants, le concept du "réseau contractuel" introduit par R. Smith (Smith, 1980) est notable. Ce réseau repose sur un mécanisme d'attribution de tâches basé sur le principe d'appel d'offres, tel que décrit par Ferber (1995).

Dans ce modèle de réseau contractuel, un agent peut occuper deux rôles spécifiques par rapport à une tâche, celui de "manager" ou de "contractant". Le processus d'appel d'offres comprend généralement quatre étapes :

**Appel d'offres :** Le manager divise une tâche en sous-tâches et recherche des contractants potentiels en diffusant une annonce à tous les agents du système.

Envoi de propositions : Les agents intéressés élaborent des propositions détaillant comment ils comptent réaliser les sous-tâches et les soumettent au manager.

**Attribution du marché :** Le manager évalue les propositions reçues, attribue la tâche à un ou plusieurs agents en fonction des critères définis et informe l'ensemble des agents de sa décision.

**Établissement du contrat :** Les agents sélectionnés pour réaliser la tâche deviennent des contractants et confirment leur engagement auprès du manager pour accomplir la tâche conformément aux termes du contrat.

Ce processus d'appel d'offres et de sélection des contractants permet une organisation flexible des tâches au sein du système multi-agents, favorisant ainsi une répartition efficace des responsabilités et des compétences pour atteindre les objectifs fixés.

1. **Planification**

La planification multi agents représente une approche de coordination dans les systèmes basés sur des agents, visant à éviter les actions conflictuelles ou incohérentes en élaborant des plans détaillant les actions futures et les interactions nécessaires pour atteindre les objectifs des agents individuels (Oliveira, 1998).

Dans le contexte des systèmes multi-agents, le processus de planification se divise généralement en trois étapes principales :

**Construction des plans :** Les agents élaborent individuellement leurs propres plans d'actions pour atteindre leurs objectifs spécifiques, en prenant en compte les ressources disponibles et les contraintes environnementales.

**Synchronisation et coordination des plans** : Cette étape implique la communication et la coordination entre les agents pour s'assurer que leurs plans individuels s'alignent et se complètent sans générer de conflits. Cela peut impliquer la résolution des incohérences ou des divergences entre les plans des différents agents.

**Exécution des plans :** Une fois que la synchronisation est réalisée, les agents mettent en œuvre leurs plans d'actions dans l'environnement partagé, en effectuant les actions définies pour atteindre leurs objectifs respectifs.

Deux approches principales de planification dans les systèmes multi-agents sont généralement considérées :

**Planification centralisée pour agents multiples :** Cette approche suppose l'existence d'une vue globale du plan. Un seul agent, disposant d'une perspective globale, est capable de planifier et d'organiser les actions pour l'ensemble des agents. Cela implique une centralisation du processus de planification.

**Planification distribuée :** Chaque agent planifie individuellement ses actions en fonction de ses propres objectifs. Les agents élaborent leurs plans de manière autonome, puis communiquent leurs plans partiels aux autres agents pour détecter et résoudre les éventuels conflits ou incohérences. Cette approche permet une décentralisation du processus de planification.

1. **Synchronisation**

La synchronisation représente le niveau de base de la coordination dans lequel sont mis en place les mécanismes fondamentaux permettant de coordonner de manière adéquate les actions entre les différents agents.

Les chercheurs identifient deux types de synchronisation distincts :

**Synchronisation par mouvement :** Cette forme de synchronisation est employée lorsqu'il est nécessaire que plusieurs éléments ou agents se déplacent conjointement ou de manière coordonnée. Dans ce contexte, la synchronisation implique de coordonner le timing, la cadence ou le positionnement temporel des actions en fonction des événements qui surviennent. Par exemple, dans le domaine des systèmes multi-agents, cela peut concerner des actions coordonnées de plusieurs agents se déplaçant dans un environnement partagé.

**Synchronisation d'accès à une ressource :** Ce type de synchronisation intervient lorsque plusieurs agents ont besoin d'accéder et de partager une ressource commune. Dans ce cas, la synchronisation vise à organiser et à réguler l'accès à la ressource partagée afin d'éviter des conflits ou des incohérences dans son utilisation. Par exemple, plusieurs agents qui doivent accéder à une base de données ou à un équipement partagé nécessitent une synchronisation pour éviter les accès concurrents ou non autorisés.

1. **Compétition**

La compétition entre agents peut découler de diverses sources. Les objectifs individuels des agents peuvent être incompatibles, créant ainsi des situations de conflit. Par exemple, dans un contexte de jeu, chaque joueur aspire à la victoire, mais la réalisation de cet objectif pour un joueur peut rendre impossible la réussite des autres joueurs. De plus, la compétition peut émerger de la rareté des ressources, où l'utilisation d'une ressource par un agent peut entraver la capacité d'un autre agent à atteindre ses propres objectifs. Pour illustrer, si une imprimante est en cours d'utilisation, cela peut occasionner un délai pour imprimer un document, empêchant ainsi son envoi avant l'heure de départ du courrier.

Cette compétition engendre des situations de conflit, où les agents peuvent être confrontés à des choix conflictuels ou à des limitations dans l'accomplissement de leurs objectifs. La résolution de ces conflits peut se faire par deux moyens principaux : la lutte directe entre les agents pour l'accès à la ressource ou pour la réalisation de leur objectif, ou par la coordination des actions entre les agents.

La coordination entre les agents offre une voie pour résoudre ces situations conflictuelles, permettant aux agents de travailler ensemble de manière organisée afin de minimiser les frictions et d'optimiser l'utilisation des ressources disponibles. En favorisant la communication, la négociation ou la répartition équitable des ressources, la coordination offre des mécanismes pour atténuer les conflits résultant de la compétition entre les agents.

1. **Délégation**

La délégation dans les systèmes multi-agents se réfère à la capacité d'un agent à exécuter des tâches pour le compte d'un autre agent. Elle offre la possibilité à un agent confronté à des contraintes telles que des ressources limitées ou des compétences insuffisantes, ou désireux de soulager sa charge de travail, de demander à d'autres agents d'accomplir des objectifs en son nom.

Cette fonctionnalité de délégation revêt une grande importance, notamment lorsque la coordination des actions est soutenue par une structure organisationnelle définie. Dans un environnement de système multi-agents caractérisé par une variabilité, exigeant ainsi une redistribution ou une mise à jour des tâches à accomplir par les différents agents, la délégation devient cruciale. Par exemple, dans un contexte organisationnel hiérarchisé, lorsque de nouveaux objectifs doivent être atteints, la délégation permet à une entité supérieure de déléguer des sous-tâches à des agents subalternes et de les ajuster en conséquence.

La délégation est souvent largement utilisée dans la gestion de réseaux, où elle apporte une flexibilité au système. Elle permet une répartition dynamique des tâches au sein du réseau, facilitant ainsi l'adaptation aux changements de l'environnement et aux nouveaux objectifs à atteindre. En offrant cette flexibilité et cette capacité d'adaptation, la délégation contribue à améliorer l'efficacité et la résilience des systèmes multi-agents dans des contextes variés et changeants

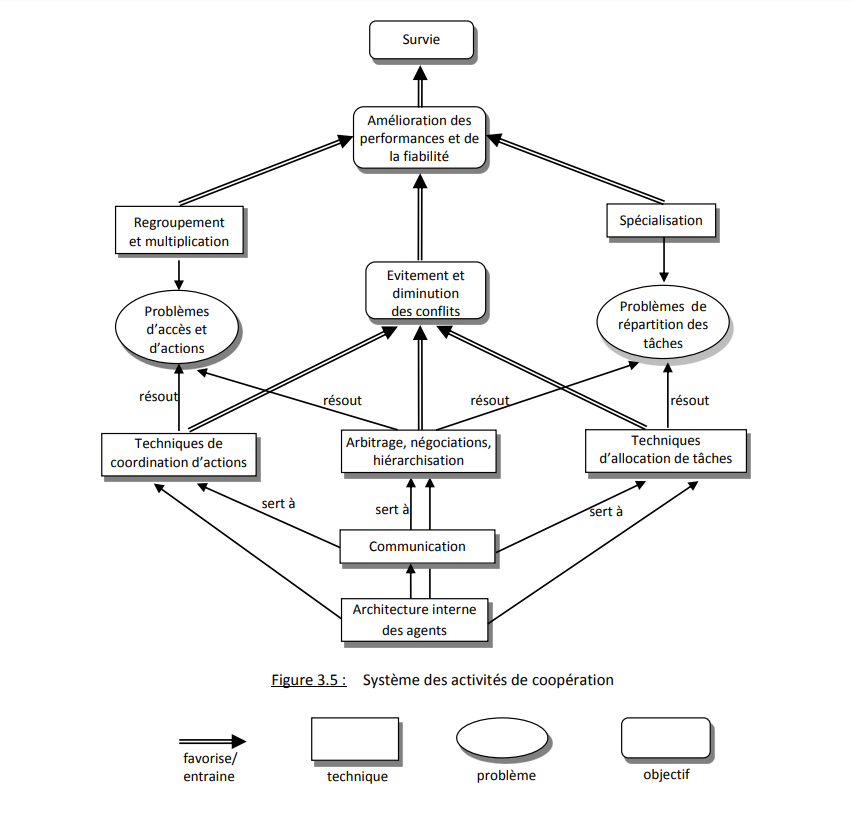
1. **Communication**

Dans un SMA, les agents communiquent entre eux en s’échangeant des informations via un langage de communication agent. Le langage, le plus utilisé aujourd’hui est KQML (Knowledge Query and Manipulation Language) qui est un langage de haut niveau utilisant une liste de types de messages, appelés performatifs. Les agents peuvent également communiquer en utilisant d’autres mécanismes tels que le mécanisme de tableau noir.

Il existe plusieurs formalismes destinés à modéliser la communication et ses attributs. Comme illustré dans la figure 20, la communication revêt un rôle crucial dans divers aspects tels que la négociation, la coordination, l'allocation de tâches, voire même la coopération et la collaboration.

En effet, la négociation implique fondamentalement deux éléments essentiels : la communication et la prise de décision consécutive à un accord entre les parties. La communication constitue un moyen d'action indispensable. Sans elle, aucune action ne peut être accomplie, ce qui rend la négociation impossible. Celle-ci représente une forme d'interaction où l'action et la réaction s'inscrivent dans une relation d'influence mutuelle.

De la même manière, l'absence de communication entrave le processus d'allocation de tâches, qui nécessite une négociation préalable et un engagement mutuel des parties concernées. La communication permet de coordonner les actions afin que les interactions ne nuisent pas au groupe, mais au contraire contribuent à son efficacité en limitant ou en gérant les conflits.

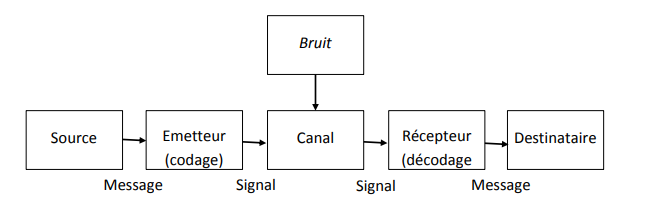
****

**Figure 20 :** Système des activités de coopération

La communication au sein des Systèmes Multi-Agents (SMA) constitue le fondement de leurs interactions et de leur organisation sociale. Sans communication, un agent se retrouve en tant qu'individu isolé, incapable d'interagir avec les autres agents, restreint à son propre cycle perception-délibération-action.

Les modalités de communication au sein des SMA incluent l'échange de messages directs, l'utilisation de mémoires partagées et la mise en place de conventions préétablies. Ces échanges se réalisent à travers des langages de communication, les ACL (Agent Communication Language), qui utilisent soit une approche procédurale, basée sur des directives pour les échanges d'informations, soit une approche déclarative qui représente la communication par des énoncés descriptifs.

La conception des langages de communication entre les agents a été un sujet de recherche important au sein de la communauté scientifique. La théorie de la communication de Claude Shannon a eu une influence majeure dans ce domaine. Elle est brièvement présentée dans la figure 3.6 suivante :



**Figure 21 :** Système de communication d’après Claude Shannon

Émetteur, canal et récepteur = « intermédiaires techniques »

Bruit = toute source d’interférence susceptible de détériorer le signal est donc d’affecter la communication.

1) La source d’information énonce un message…

2) … que l’émetteur va décoder et transformer en signal

3) Lequel va être acheminé par le canal

4) Puis décodé par le récepteur, qui reconstitue un message à partir du signal

5) Et le transmet enfin au destinataire.

## Etapes de réalisation d’un SMA

La démarche à suivre pour réaliser un SMA est:

1. Déterminer :

* Les agents
* L’environnement

1. Décrire les lois de l’environnement.

3. Identifier les perceptions et les influences (actions) produites par les agents.

4. Déterminer les variables internes et capacités des agents.

5. Définir les comportements des agents:

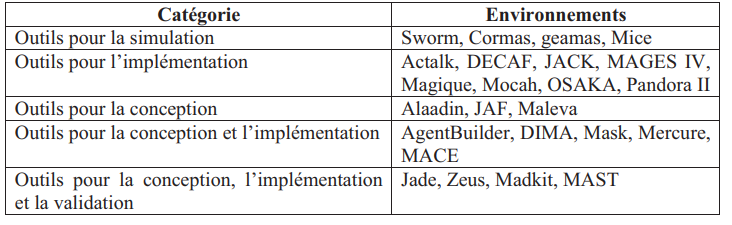
* Si les agents sont cognitifs: décrire la relation entre croyances, buts et actions
* Si les agents sont réactifs: décrire les stimuli, les tropismes (attraction, répulsion, évitement) ainsi que les tâches (combinaisons d’actions élémentaires).

## Plates-formes de développement

Les environnements de développements des SMA, les plus connus sont :

* JADE (Java Agent DEvelopement) est une plate-forme de développement de systèmes multi agents, crée par le laboratoire TILAB et répond aux spécifications FIPA (Foundation for Intelligent Physical Agents).
* ZEUS est un environnement intégré qui utilise la méthodologie Rôle Modeling pour la construction rapide d’applications à base d’agents collaboratifs
* MadKit est une plate-forme multi agents modulaire écrite en Java et construite autour du modèle organisationnel Agent/Groupe/Rôle.
* KQML (KnowledgeQueryandManipulation Language) : Langage de communication entre agents ACL (Agent communication Language). Et d’autres moins connus :
* ADELPHE, AgentBuilder, Madkit, SamllTalk, LISP, LISP, C++, Prolog, SMALTALK, AgentBuilder, Mask, Mercure, MACE, DIMA, SWORM, Cormas, geamas, Mice, FIPA-OS, LEAP, Actalk, DECAF, JACK, MAGES IV, Magique, Mocah, OSAKA, Pandora II, Alaadin, JAF, Maleva Agent Oriented Programming in Java ,Java Agent Services. (Boissier & all, 1999) Les plates-formes fournies comme logiciels libres:
* JADE, MACE, ZEUS, et MADKIT pour les agents cognitifs,
* Et SWORM pour les agents réactifs.

Les différents environnements de programmation dans le tableau suivant :

****

**Figure 16 :** Environnements de développement

# **Conclusion**

L'émergence de l'intelligence artificielle distribuée est une réponse aux défis de complexité rencontrés par les grands programmes d'intelligence artificielle. Par exemple, pour résoudre des problèmes de classification, l'exécution peut être distribuée tandis que le contrôle reste souvent centralisé. Cette approche diffère des Systèmes Multi-Agents (SMA), où chaque agent détient un contrôle autonome sur son comportement. La conception de programmes relativement petits (les agents) en interaction est souvent préférable à un seul gros programme pour résoudre des problèmes complexes. L'autonomie des agents permet au système de s'adapter de manière dynamique aux changements imprévus dans son environnement.

Ce qui rend remarquable la montée en puissance de ces nouveaux outils est leur diffusion généralisée dans divers domaines scientifiques. En effet, les Systèmes Multi-Agents ont trouvé des applications dans de nombreux secteurs technologiques ainsi que dans les sciences sociales et humaines.

Ainsi, la technologie des systèmes multi-agents apparaît comme une solution prometteuse pour le développement des logiciels de demain. Nous avons fait appel à cette technologie dans un contexte très pointu de classification automatique de textes, et les premiers résultats semblent être bénéfiques. Ces résultats seront confirmés ultérieurement, mais cette approche semble offrir des perspectives prometteuses dans ce domaine spécifique.

# **Sockets**

# **Introduction**

La notion de sockets a été introduite dans les distributions de Berkeley, un système de type UNIX largement utilisé dans de nombreuses distributions actuelles. On les désigne parfois comme des sockets BSD (Berkeley Software Distribution) en raison de leur origine.

Les sockets représentent un modèle permettant la communication inter-processus (IPC - Inter Processus Communication). Cela permet à différents processus de communiquer, que ce soit sur une même machine ou à travers un réseau TCP/IP.

La communication via les sockets est souvent comparée aux interactions humaines, et elle se divise en deux modes distincts :

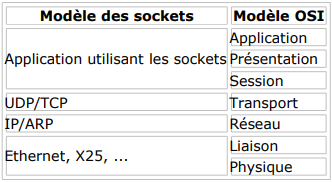
1- Le mode connecté : comparable à une communication téléphonique, utilise le protocole TCP. Dans ce mode, une connexion persistante est établie entre les deux processus. Ainsi, la socket de destination n'a pas besoin d'être spécifiée à chaque envoi de données, offrant une communication plus fiable et garantissant l'intégrité des données transmises.

2- Le mode non connecté : similaire à une communication par courrier, utilise le protocole UDP. Ce mode requiert l'adresse de destination à chaque envoi de données, et aucune confirmation de réception n'est générée. Il offre une communication plus rapide mais moins fiable, car il ne garantit pas la réception des données ni leur intégrité.



## Position des sockets dans le modèle OSI

Les sockets se situent juste au-dessus de la couche transport du modèle OSI (protocoles UDP ou TCP), elle-même utilisant les services de la couche réseau (protocole IP / ARP)



**Figure 22 :** tableau de situation des sockets dans le modèle OSI

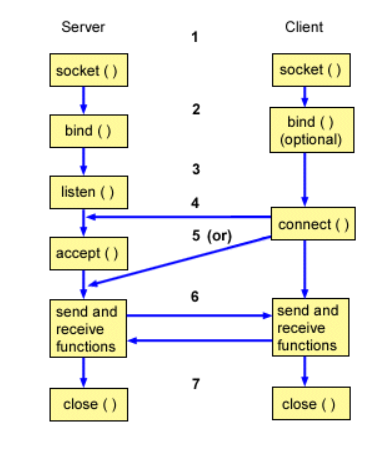
## Fonctionnement des sockets

Les sockets sont largement employés pour faciliter l'interaction entre un client et un serveur. Dans une configuration système typique, le serveur est situé sur une machine distincte, tandis que les clients se trouvent sur d'autres machines. Les clients établissent une connexion avec le serveur, échangent des informations, puis se déconnectent.

Un socket suit généralement un flux d'événements spécifique. Dans un modèle client-serveur basé sur la connexion, le socket du processus serveur attend les demandes provenant des clients. Pour ce faire, le serveur initialise (ou lie) d'abord une adresse à laquelle les clients peuvent se référer pour trouver le serveur. Une fois que cette adresse est établie, le serveur est en attente des requêtes de services de la part des clients.

L'échange de données entre le client et le serveur a lieu lorsqu'un client se connecte au serveur via un socket. Le serveur traite alors la demande émise par le client et retourne la réponse correspondante au client. Cette communication bidirectionnelle entre les sockets client et serveur permet d'assurer l'exécution des services requis par les clients et la transmission des résultats vers ces derniers.

La figure suivante illustre le flux typique d'événements (et la séquence des API émises) pour une session de socket orientée connexion. Une explication de chaque événement suit la figure.



**Figure 23 :** flux typique d'événements

Voici un déroulement typique des événements pour un socket orienté connexion :

1- L'API socket() crée un point final de communication et retourne un descripteur de socket représentant ce point final.

2- Lorsqu'une application dispose d'un descripteur de socket, elle peut associer un nom unique à ce socket. Les serveurs doivent attribuer un nom pour que le socket soit accessible via le réseau.

3- L'API listen() signale la disposition à accepter les demandes de connexion des clients. Une fois qu'un socket a émis l'API listen(), il ne peut pas initier activement des demandes de connexion. Cette API est émise après avoir alloué un socket via socket() et lié un nom via bind(). Une API listen() doit précéder une API accept().

4- L'application cliente utilise l'API connect() sur un socket de flux pour établir une connexion vers le serveur.

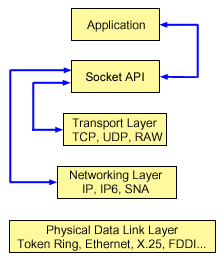
5- L'application serveur utilise l'API accept() pour accepter une demande de connexion provenant du client. Le serveur doit préalablement émettre les API bind() et listen() pour être prêt à accepter une connexion via accept().

6- Lorsqu'une connexion est établie entre des sockets de flux (client et serveur), vous pouvez utiliser différentes APIs de transfert de données telles que send(), recv(), read(), write(), etc., pour échanger des données.

7- Lorsqu'un serveur ou un client souhaite mettre fin à ses opérations, il émet une API close() pour libérer toutes les ressources système utilisées par le socket.

Ce processus séquentiel illustre les étapes habituelles pour établir, gérer et clôturer une communication via des sockets orientés connexion entre les applications cliente et serveur.

**Remarque :** Les API de socket se trouvent dans le modèle de communication entre la couche application et la couche transport. Les API de socket ne sont pas une couche du modèle de communication. Les API de socket permettent aux applications d'interagir avec les couches de transport ou de réseau du modèle de communication standard. Les flèches de la figure suivante montrent la position d'un socket et la couche de communication fournie par le socket.



**Figure 24 :** API socket

# **Conclusion**

En somme, les sockets constituent un moyen flexible et puissant pour permettre aux processus de communiquer de manière fiable, que ce soit localement sur la même machine ou à travers un réseau, en utilisant différents protocoles adaptés à différents besoins de communication.

# **Expérimental**

# **Conception**

Après avoir achevé les phases d’analyse et de l’étude technique de notre projet de fin d’études, nous entamons dans ce chapitre, la conception détaillée de l’application, à savoir les diagrammes de séquences et les diagrammes de classes.

# **Introduction**

L'évolution de la technologie a suscité un intérêt croissant pour les systèmes multi-agents en tant que solution prometteuse pour résoudre des problèmes complexes de routage, notamment dans le contexte du Vehicle Routing Problem (VRP). Cette problématique, souvent caractérisée par la nécessité d'optimiser les itinéraires de livraison ou de service, bénéficie des avancées notables en apprentissage automatique, modèles d'apprentissage profond et réseaux neuronaux.

L'application de l'apprentissage automatique et des modèles de réseaux neuronaux profonds offre une approche novatrice pour améliorer les stratégies de routage dans le VRP. Ces techniques permettent d'apprendre des politiques de routage efficaces en exploitant des données historiques et en analysant des schémas complexes pour optimiser les itinéraires, minimiser les distances parcourues et réduire les temps de trajet.

Les modèles d'apprentissage automatique peuvent également prédire avec précision la demande future des clients, permettant ainsi une planification plus efficace des itinéraires. En intégrant des données de localisation en temps réel, ces modèles peuvent s'adapter aux changements dynamiques tels que les conditions de trafic, les préférences des clients et les perturbations imprévues.

Par ailleurs, les réseaux neuronaux offrent la capacité de modéliser des interactions complexes entre différents éléments du système, tels que les véhicules, les entrepôts et les destinations, permettant une optimisation fine des itinéraires.

Dans cette étude, nous avons commencé par un diagramme de cas d’utilisation, à partir duquel nous avons développé les use case en utilisant des diagrammes de séquence système. Nous avons ensuite réalisé un diagramme de classe, qui nous a permis de créer des diagrammes de séquence détaillés.

## Diagramme de classe

Les diagrammes de classes représentent l'un des six types de diagrammes structurels. Ils jouent un rôle crucial dans la modélisation des objets en dépeignant la configuration statique d'un système.

Selon la complexité du système, un unique diagramme de classes peut suffire pour saisir l'ensemble du système, mais il est également envisageable d'utiliser plusieurs diagrammes de classes pour cerner les divers composants d'un système.

Ces diagrammes sont véritablement les plans ou schémas directeurs de votre système ou sous-système. Leur utilisation permet de modéliser les entités constituant le système, de représenter les liens qui les unissent, tout en décrivant leurs attributs et les services qu'ils offrent.

Les diagrammes de classes s'avèrent bénéfiques à plusieurs étapes du processus de conception. En phase d'analyse, ils contribuent à comprendre les exigences spécifiques au domaine d'étude et à identifier ses différentes composantes. Dans un contexte de développement logiciel orienté objet, les classes esquissées dans les premiers diagrammes de classes se transposeront souvent en classes et objets logiciels concrets au moment de la phase de codage.

Par la suite, vous avez la possibilité d'élaborer vos modèles d'analyse et de conception initiaux à travers des diagrammes de classes dédiés, mettant en lumière des composants spécifiques du système, des interfaces utilisateurs ou encore des implémentations logiques. Ainsi, ces diagrammes de classes offrent une vue précise du fonctionnement de votre système, des interactions entre ses composants à divers niveaux et des orientations quant à leur implémentation.

En somme, ces diagrammes de classes sont des outils visuels pour visualiser, définir et documenter les fonctions structurelles au sein de vos modèles. Durant les phases d'analyse et de conception du cycle de développement, ils trouvent leur utilité dans la représentation des éléments suivants :

* Capturer et définir la structure des classes et autres discriminants
* Définir les relations entre les classes et discriminants
* Illustrer la structure d'un modèle à l'aide d'attributs, d'opérations et de signaux
* Afficher les rôles et responsabilités de discriminant communs qui définissent le comportement du système
* Afficher les classes d'implémentation dans un package
* Afficher la structure et le comportement d'une ou plusieurs classes
* Afficher une hiérarchie d'héritage entre des classes et discriminants
* Afficher les travailleurs et entités comme des modèles objet métier

Pendant la phase d'implémentation d'un cycle de développement d'application, nous pouvons utiliser des diagrammes de classes pour convertir nos modèles en code et pour convertir note code en modèles.

Les rubriques suivantes décrivent les éléments de modèle dans les diagrammes de classes :

* [**Classes**](https://www.ibm.com/docs/fr/SS5JSH_9.5.0/com.ibm.xtools.modeler.doc/topics/cclss.html)  
  Dans le langage UML, une *classe* représente un objet ou un ensemble d'objets qui partagent une structure et un comportement communs. Les classes ou instances de classes, sont des éléments de modèle communs dans les diagrammes UML.
* [**Objets**](https://www.ibm.com/docs/fr/SS5JSH_9.5.0/com.ibm.xtools.modeler.doc/topics/cobj.html)  
  Dans les modèles UML, les *objets* sont des éléments de modèle représentant les instances d'une ou plusieurs classes. Vous pouvez ajouter des objets à votre modèle pour représenter des instances concrètes et prototypiques. Une instance concrète représente une personne ou une chose réelle. Par exemple, une instance concrète d'une classe Customer représente un client réel. Une instance prototypique d'une classe Customer contient des données représentant un client type.
* [**Packages**](https://www.ibm.com/docs/fr/SS5JSH_9.5.0/com.ibm.xtools.modeler.doc/topics/cpkg.html)  
  Les packages regroupent des éléments de modèle associés de tout type, y compris d'autres packages.
* [**Signaux**](https://www.ibm.com/docs/fr/SS5JSH_9.5.0/com.ibm.xtools.modeler.doc/topics/csignal.html)  
  Dans les modèles UML, les *signaux* sont des éléments de modèle indépendants des discriminants qui les gèrent. Les signaux spécifient des communications asynchrones oneway entre des objets actifs.
* [**Enumérations**](https://www.ibm.com/docs/fr/SS5JSH_9.5.0/com.ibm.xtools.modeler.doc/topics/cenum.html)  
  Dans les modèles UML, les *énumérations* sont des éléments de modèle dans les diagrammes de classes qui représentent des types de données définis par l'utilisateur. Les énumérations contiennent des ensembles d'identificateurs nommés qui représentent les valeurs de l'énumération. Ces valeurs sont appelées littéraux d'énumération.
* [**Type de données**](https://www.ibm.com/docs/fr/SS5JSH_9.5.0/com.ibm.xtools.modeler.doc/topics/cdatatypes.html)  
  Dans les diagrammes UML, les *types de données* sont des éléments de modèle qui définissent des valeurs de données. Vous utilisez généralement les types de données pour représenter des types primitifs, tels que les types integer (entier) ou string (chaîne), et des énumérations, telles que des types de données définis par l'utilisateur.
* [**Artefacts**](https://www.ibm.com/docs/fr/SS5JSH_9.5.0/com.ibm.xtools.modeler.doc/topics/cartifact.html)  
  Dans les modèles UML, les *artefacts* sont des éléments de modèle qui représentent les entités physiques dans un système logiciel. Les artefacts représentent des unités physiques d'implémentation, telles que des fichiers exécutables, des bibliothèques, des composants de logiciel, des documents, et bases de documents.
* [**Relations dans les diagrammes de classes**](https://www.ibm.com/docs/fr/SS5JSH_9.5.0/com.ibm.xtools.modeler.doc/topics/crelsme_clssd.html)  
  En langage UML, une relation est une connexion entre des éléments de modèle. Une relation UML est un type d'élément de modèle qui ajoute une sémantique à un modèle en définissant la structure et le comportement entre les éléments de modèle.
* [**Qualificateurs sur les extrémités d'association**](https://www.ibm.com/docs/fr/SS5JSH_9.5.0/com.ibm.xtools.modeler.doc/topics/cqualassocend.html)  
  Dans UML, les *qualificateurs* sont des propriétés applicables aux associations binaires et des composants facultatifs des extrémités d'association. Un qualificateur contient une liste d'attributs d'association, doté chacun d'un nom et d'un type. Les attributs d'association modélisent les clés utilisées pour indexer un sous-ensemble d'instances de relation.

L'illustration suivante est un diagramme de classes qui décrit les différents classes de notre application avec le nom du classe, les différents attributs et les méthodes en respectant les principes de la programmation oriente objets (encapsulation qui dit que les attributs sont privées +, et les méthodes sont public -). Ce diagramme montre comment une classe représentant un :

AgentOne : recevoir les informations sur les véhicules stocker dans une liste de classe Car et les sérialiser après il envoie l’information vers l’AgentThree.

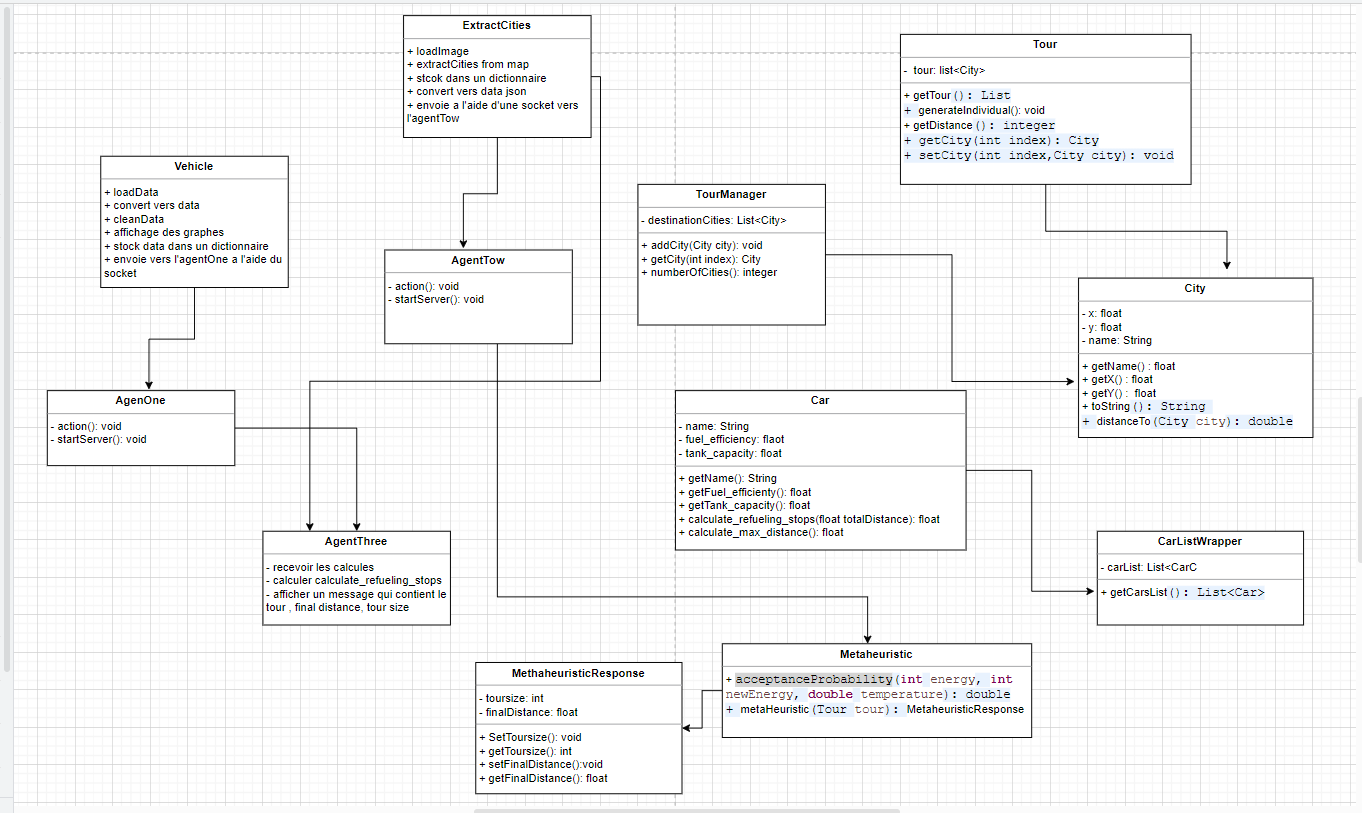
AgentTow : recevoir les informations des villes les coordonnes et le nom de la ville stocker dans une liste de classe City pour calculer le trajet minimal en utilisant diffèrent méthode pour ca après il envoie le résultat de calcul vers l’AgentThree.

City : modéliser les différentes destinations de notre voyageur vendeur.

TourManager : suivre les villes

Tour : modéliser une tournée de voyageur de commerce.

Metaheuristic : Cree l’algorithme SA



**Figure 25 :** diagramme de classe du système

## Diagramme de séquence

Pour appréhender la nature d'un diagramme de séquence, il est crucial de comprendre le rôle de l'Unified Modeling Language (UML), connu sous le nom d'UML. L'UML représente un outil de modélisation qui oriente la conception et la notation de divers types de schémas, incluant les diagrammes comportementaux, d'interaction et structurels.

Le diagramme de séquence appartient à la catégorie des diagrammes d'interaction, décrivant ainsi le fonctionnement et l'ordre des interactions entre plusieurs objets. Ces diagrammes sont utilisés tant par les développeurs logiciels que par les responsables d'entreprise pour analyser les besoins d'un nouveau système ou documenter des processus existants. Ils peuvent également être désignés comme des diagrammes d'événements ou des scénarios d'événements.

Il est à noter qu'il existe deux types de diagrammes de séquence : les diagrammes UML et ceux basés sur du code. Ces derniers émanent directement du code de programmation et ne seront pas abordés dans cette explication. Le logiciel de création de diagrammes UML de Lucidchart est doté de toutes les formes et fonctionnalités nécessaires pour modéliser ces deux types de diagrammes.

### **Avantages de diagramme de séquences**

Les diagrammes de séquence offrent une utilité pratique pour diverses entreprises et organisations. Ils servent à :

* Illustrer en détail un cas d'utilisation UML, mettant en scène les interactions entre les acteurs et le système pour représenter des fonctionnalités spécifiques.
* Modéliser la logique d'une procédure, fonction ou opération complexe en présentant de manière séquentielle les étapes et les interactions entre les divers acteurs ou composants impliqués.
* Examiner l'interaction entre objets et composants pour visualiser comment ils collaborent pour réaliser un processus spécifique. Cela permet de saisir l'interdépendance et la séquence des actions.
* Schématiser et analyser en profondeur le déroulement détaillé d'un scénario actuel ou futur. Cela permet de comprendre comment différentes entités interagissent dans ce scénario particulier, aidant à déterminer les ajustements ou les améliorations potentielles.

En somme, les diagrammes de séquence sont des outils polyvalents permettant de représenter visuellement les interactions et les séquences d'actions entre les composants, les objets ou les acteurs, offrant ainsi une vision claire et détaillée des processus, des opérations et des interactions au sein d'un système.

### **Cas d’utilisation de diagramme de séquence**

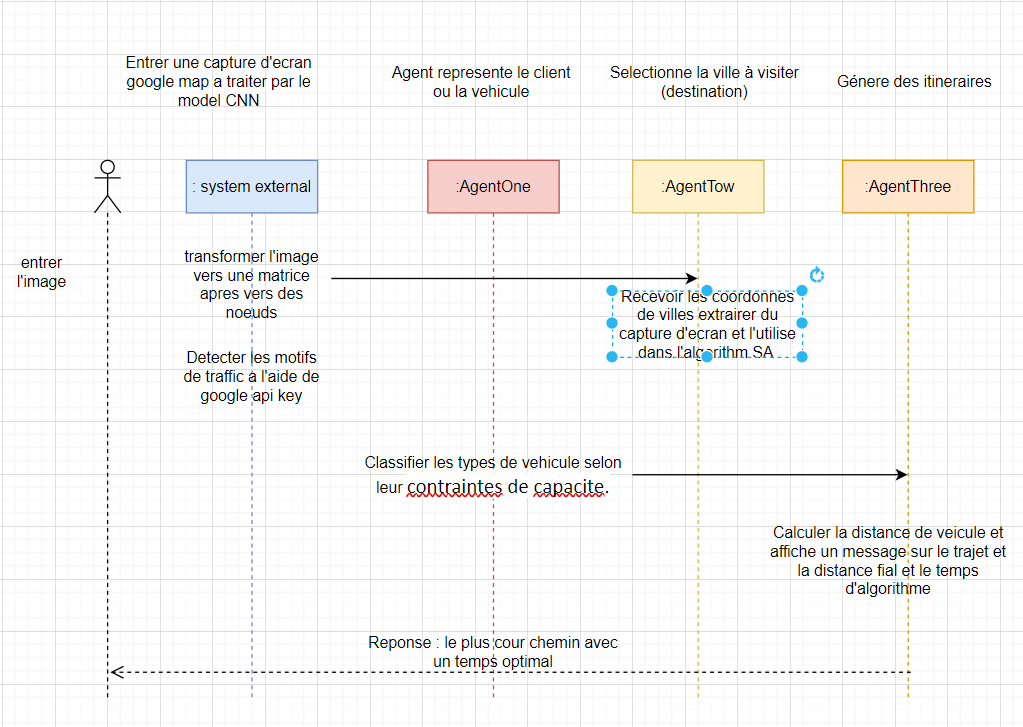
**Scenario d’utilisation :** Un scénario d'utilisation représente un schéma décrivant l'éventuelle utilisation de votre système. Il sert d'outil efficace pour garantir la prise en compte de la logique inhérente à tous les cas d'utilisation du système.

**Logique de méthode :** Tout comme le diagramme de séquence UML est utile pour analyser la logique d'un cas d'utilisation, il peut également être employé pour examiner la logique d'une fonction, d'une procédure ou même d'un processus complexe.

**Logique de service :** Si l'on envisage un service comme une méthode générale exploitée par divers clients, un diagramme de séquence s'avère l'outil privilégié pour le représenter visuellement.

**Diagramme de séquence Visio : Lucidchart offre la possibilité de charger tout diagramme de séquence créé avec Visio. Il prend en charge l'importation de fichiers aux formats .vsd et .vdx, offrant ainsi une alternative solide à Microsoft Visio. La grande majorité des images présentées dans la section UML de ce site ont été créées en utilisant Lucidchart.**

L'illustration suivante est un diagramme de séquence qui décrit le processus de notre application les messages envoyés par chaque agent et les reponses.



**Figure 26 :** diagramme de séquence du système.

Pour comprendre le diagramme de séquence de notre application, nous devons connaître ses symboles et ses composants. Les diagrammes de séquence sont composés des icônes et des éléments suivants :

|  |  |  |
| --- | --- | --- |
| Symbole | Nom  Symbole d’acteur  Symbole d’objet  Symbole de ligne de vie  Symbole de message synchones  Symbole de messages de retour asynchrones | Description  Montre les entités qui interagissent avec le système ou qui sont extérieures à lui.  Représente une classe ou un objet en langage UML. Le symbole objet montre comment un objet va se comporter dans le contexte du système. Les attributs de classe ne doivent pas être énumérés dans cette forme.  Représente le passage du temps qui se prolonge vers le bas. Cette ligne verticale en pointillés montre les événements séquentiels affectant un objet au cours du processus schématisé. Les lignes de vie peuvent commencer par une forme rectangulaire avec un intitulé ou par un symbole d'acteur.  Représentés par une ligne pleine terminée par une pointe de flèche pleine. On utilise ce symbole lorsqu'un expéditeur doit attendre une réponse à un message avant de continuer. Le diagramme doit montrer à la fois l'appel et la réponse.  Représentés par une ligne en pointillés terminée par une tête de flèche. |



### **Diagramme de cas d’utilisation**

En langage UML, les diagrammes de cas d'utilisation servent à modéliser le comportement d'un système en capturant ses exigences. Ils dépeignent les fonctions globales et la portée d'un système, mettant en évidence les interactions entre le système et ses acteurs. Ces diagrammes présentent ce que le système effectue et comment les acteurs l'utilisent, sans détailler son fonctionnement interne.

Les diagrammes de cas d'utilisation définissent le contexte et les exigences d'un système dans son intégralité ou pour des composants spécifiques. Ils peuvent représenter un système complexe dans un unique diagramme ou être déclinés en plusieurs diagrammes pour modéliser distinctement les composants du système. Ces diagrammes sont généralement élaborés dans les premières phases d'un projet et servent de référence tout au long du processus de développement.

Les diagrammes de cas d'utilisation se révèlent utiles dans différentes situations :

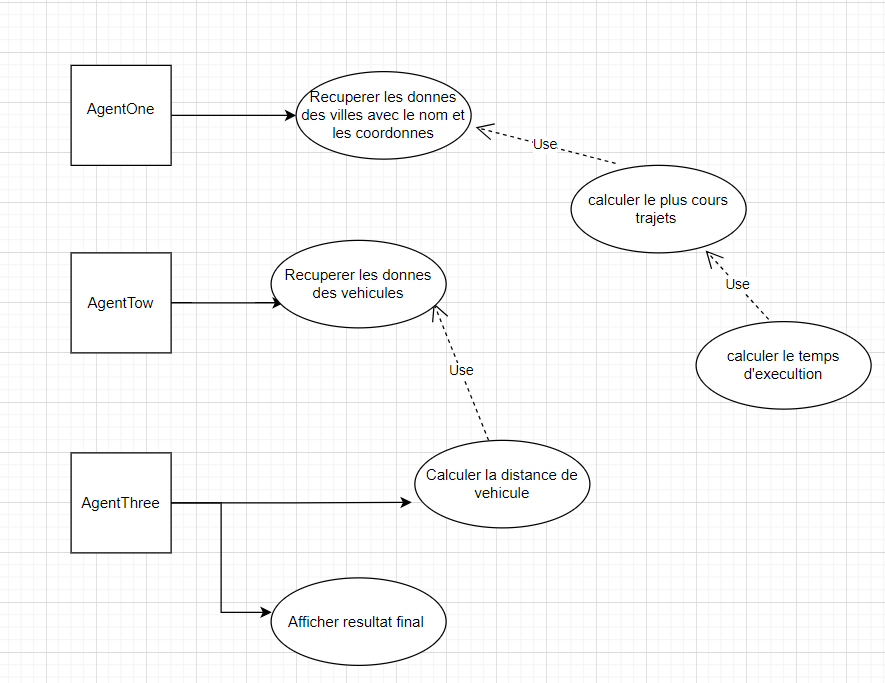
* En amont d'un projet, ils permettent de modéliser l'entreprise pour donner une vision globale des employés, des clients et des activités à tous les participants du projet.
* Pendant la collecte des exigences, ils captent les besoins du système et présentent de façon visuelle ce que le système doit accomplir.
* Durant les phases d'analyse et de conception, les cas d'utilisation et les acteurs identifiés dans ces diagrammes aident à déterminer les classes nécessaires.

En phase de test, ces diagrammes aident à identifier les tests à effectuer pour le système.

Ainsi, les diagrammes de cas d'utilisation sont des outils polyvalents qui facilitent la compréhension, la communication et la planification tout au long du cycle de développement d'un système.

Les rubriques suivantes décrivent les éléments de modèle dans les diagrammes de cas d'utilisation :

* [**Cas d'utilisation**](https://www.ibm.com/docs/fr/SS8PJ7_9.5.0/com.ibm.xtools.modeler.doc/topics/cuc.html)  
  Un cas d'utilisation décrit une fonction qu'un système exécute pour atteindre l'objectif de l'utilisateur. Un cas d'utilisation doit renvoyer un résultat observable qui est utile pour l'utilisateur du système.
* [**Acteurs**](https://www.ibm.com/docs/fr/SS8PJ7_9.5.0/com.ibm.xtools.modeler.doc/topics/cactor.html)  
  Un acteur représente un rôle d'un utilisateur qui interagit avec le système que vous modélisez. L'utilisateur peut être un utilisateur humain, une organisation, une machine ou un autre système externe.
* [**Sous-systèmes**](https://www.ibm.com/docs/fr/SS8PJ7_9.5.0/com.ibm.xtools.modeler.doc/topics/csubsys.html)  
  Dans les modèles UML, les sous-systèmes sont un type de composant stéréotypé représentant des unités comportementales indépendantes dans un système. Les sous-systèmes sont utilisés dans les diagrammes de classes, de composants et de cas d'utilisation pour représenter des composants de grande taille dans le système que vous modélisez.
* [**Relations dans les diagrammes de cas d'utilisation**](https://www.ibm.com/docs/fr/SS8PJ7_9.5.0/com.ibm.xtools.modeler.doc/topics/crelsme_ucd.html)  
  En langage UML, une relation est une connexion entre des éléments de modèle. Une relation UML est un type d'élément de modèle qui ajoute une sémantique à un modèle en définissant la structure et le comportement entre les éléments de modèle.



**Figure 27 :** donne une présentation synthétisée du diagramme de cas d’utilisation de notre système multi agent..

Le diagramme de cas d’utilisation est un diagramme d’UML 2.0 qui montre les interactions fonctionnelles entre les acteurs et le système étudié. En général les acteurs sont des acteurs externes (humain ou non, ou autres systèmes externes). Un cas d'utilisation est un ensemble de scénarii liés par un but utilisateur commun. Un scénario est une séquence d'étapes décrivant une interaction entre un utilisateur et le système.

Malgré que les agents d’un SMA ne soient pas des acteurs externes au système auquel ils appartiennent, nous avons décidé de présenter ce diagramme pour donner un aperçu d’un certain nombre de buts des agents du système et les relations fonctionnelles entre ces buts et les agents. Le diagramme de cas d’utilisation peut permettre de valider les buts du système et déboucher éventuellement sur un raffinement des rôles qui en découlent.

# **Conclusion**

La conception de systèmes complexes repose souvent sur l'utilisation judicieuse de divers diagrammes pour représenter, modéliser et comprendre les différentes facettes et interactions entre les éléments du système. À travers cette exploration des différents diagrammes utilisés dans la conception, nous constatons l'importance cruciale de ces représentations visuelles pour la clarté, la communication et la structuration des idées.

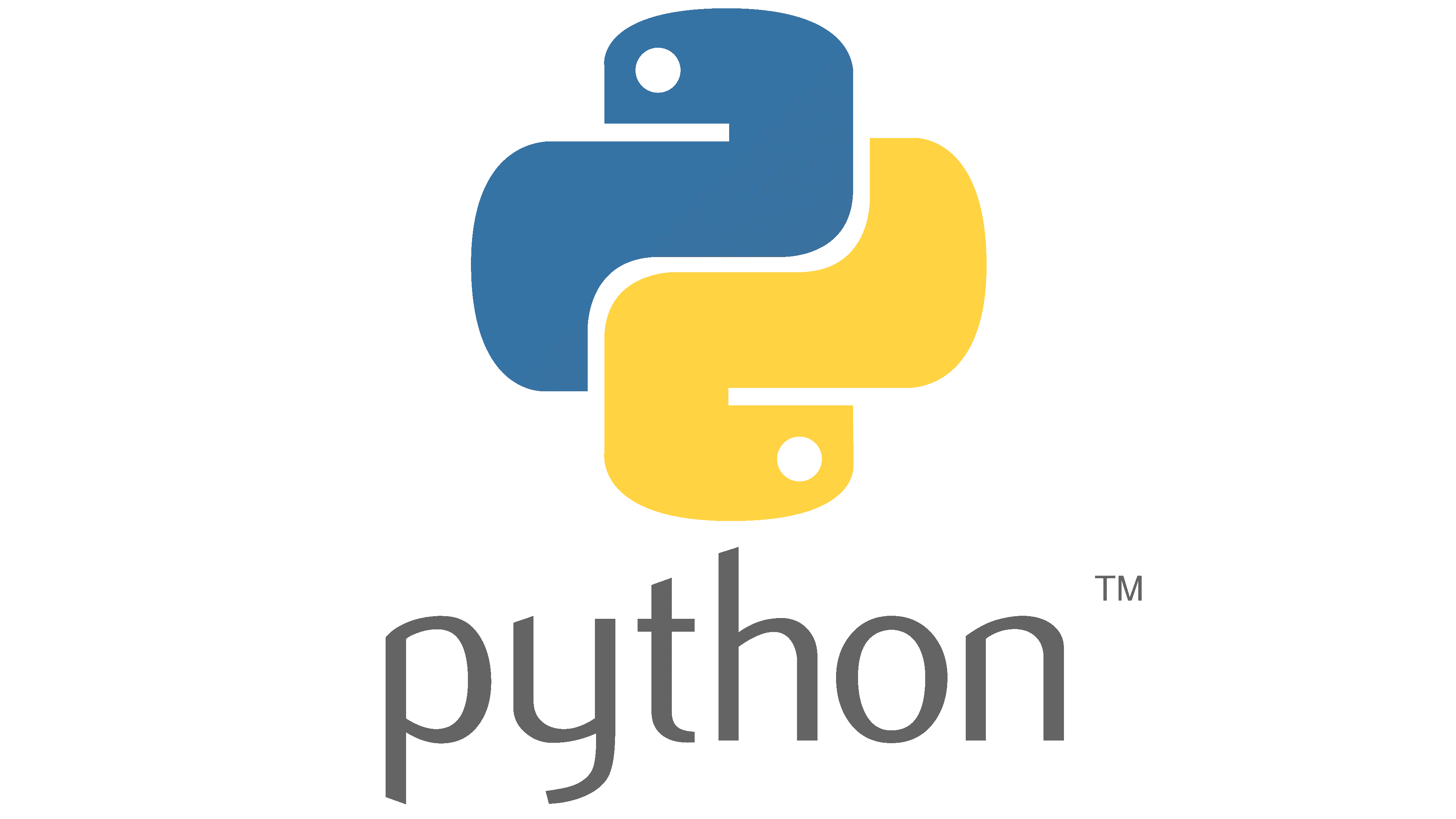
Chaque type de diagramme apporte une perspective unique et indispensable à la conception globale du système. L'utilisation appropriée et combinée de ces différents schémas permet de capturer efficacement les différentes dimensions et interrelations du système, facilitant ainsi la communication, la compréhension et la mise en œuvre réussie des projets complexes.

# **Environnement de travail**

Nous allons, à travers Cette partie de ce chapitre, décrire les différents outils de travail concernant les langages de programmation tel que java et python les bibliothèques tel que openCV, Pytesseract, les Framework d'architectures multi agent tel que JADE.

## Python

### **Les origines de langages de programmation python**

**Guido van Rossum**, un développeur néerlandais, **invente Python en 1989**. Il avait à cœur d’automatiser les tâches répétitives liées à **l’écriture de scripts informatiques.**

Personne n’imaginait alors que Python puisse connaître un tel succès. La première version du langage a été écrite durant une semaine de vacances et le nom "Python" est une référence aux humoristes du groupe : les "Monty Python". En **1991**, Guido van Rossum publie la **première version publique du langage**sur le forum Usenet.

Depuis sa création, plusieurs versions de Python ont été développées. En **2000**, l’équipe de **PythonLabs publie la version 2.0** du langage. **Fin 2008**, les développeurs de Python décident de nettoyer la bibliothèque du langage et de clarifier son code : la **version 3.0 de Python** est née.

Les différences entre Python 2 et 3 sont telles que la plupart des lignes de code écrites avec Python 2 ne sont plus compatibles avec Python 3. Par exemple, pour afficher un résultat avec Python, on utilise la fonction “print”.

Avec Python 2, la fonction s’écrit :

###### print 'Hello, World!'

Avec Python 3, la fonction s’écrit :

###### print('Hello, World!')

L’objectif de ces deux instructions est le même : afficher les caractères qui forment la phrase :

###### Hello, World!

Cependant, en informatique, la **syntaxe est très importante**. C’est pourquoi Python 3 n’arrivera pas à comprendre et à exécuter l’instruction écrite avec Python 2.

Cependant, **les deux versions continuent de coexister** aujourd’hui mais Python 2 a vocation à disparaître dans les prochaines années.

Aujourd’hui, **Python 3** est devenu un **langage très populaire**. Par exemple, si tu as utilisé une calculatrice Casio ou Texas Instrument au lycée, tes calculs ont été exécutés grâce à Python 3.

[Le classement Tiobe](https://www.tiobe.com/tiobe-index/), qui recense les **langages les plus populaires** et fait autorité dans le monde de la programmation, place même Python en première place depuis décembre 2021 !





### **Les avantages et inconvénients de Python**

**Python est un langage simple, puissant et facile à apprendre**. Son premier avantage est d’utiliser des fonctions en anglais : si tu as quelques notions d’anglais, il sera facile de te souvenir à quoi servent ces fonctions.

De plus, la **syntaxe de Python** a beau présenter quelques subtilités, elle est beaucoup moins lourde et rigide que celle des autres langages. Par exemple, on peut insérer un commentaire avec un simple hashtag (#) au début du commentaire, alors qu’en Java et en C il faut ouvrir le commentaire par /\* et ne pas oublier de le refermer avec \*/.

**L’apprentissage de Python** est donc très accessible (même pour les débutants), et c’est souvent avec Python qu’on **débute la programmation**.

Les développeurs expérimentés tirent également avantage de la **souplesse d’utilisation du langage Python**. Une syntaxe simple permet d’écrire des programmes fonctionnels plus rapidement.

En effet, pour qu’une **ligne de code fonctionne**, elle doit être **correctement écrite**.

Par exemple, aucune messagerie n’empêche d’envoyer un message qui contient des fautes d'orthographe. Malgré la mauvaise expérience du lecteur, le contenu du mail peut être compris par celui ou celle qui le recevra. Avec un langage de programmation, c’est une autre histoire : s’il existe une faute de syntaxe dans une ligne de code, celle-ci ne fonctionnera pas. C’est pourquoi certains développeurs passent beaucoup de temps à débugger leur code afin qu’il puisse fonctionner.

Lorsqu’on **débute avec un langage de programmation**, trouver la petite faute qui empêche le code de fonctionner peut être un exercice chronophage et assez décourageant. La **simplicité de la syntaxe**du langage **Python**est un grand **avantage**puisque cela te permet d’écrire des lignes de code qui fonctionnent ou d’identifier rapidement ce qu’il faut corriger.

Ainsi, **coder en Python** permet aux développeurs de se concentrer sur le but de leur programme, sans avoir à débugger leur code en permanence pour corriger des erreurs de syntaxe.

Parallèlement, **Python**est un **langage**extrêmement **polyvalent**et utilisable dans de nombreux contextes. Il est utile autant pour les **programmeurs**, qui développent **applications et logiciels**, que pour les **professionnels de la Data science**. En effet, Python est compatible avec n’importe quel système d’exploitation : Windows, macOS, UNIX, Linux…

Cette polyvalence ne nuit pas à la qualité du langage. **Python est un langage puissant et complet**. A condition d’être un bon développeur, Python permet de réaliser tout type de projet avec un **niveau d’exigence élevé**. C’est pourquoi de grandes entreprises comme Google, la Nasa, Microsoft ou Instagram (pour n’en citer que quelques-unes) utilisent Python.

Il y a donc de nombreuses [raisons d’apprendre Python en 2023](https://www.data-bird.co/blog/apprendre-python) !

En particulier, le langage Python est un **incontournable de la Data Science.** Si le [métier de Data Analyst](https://www.data-bird.co/blog/data-analyst) t’intéresse, maîtriser Python est une compétence très recherchée par les recruteurs. Il est également très **apprécié par les Data Scientists** : la majorité d’entre eux travaillent avec Python au quotidien.

Enfin, **Python est un langage qui ne vieillit pas**. L'équipe de développeurs met à jour le langage et développe de nouvelles fonctionnalités régulièrement. Tu peux te référer à la documentation officielle du site [**python.org**](https://www.python.org/). En parallèle, le langage bénéficie d’une **communauté**importante et **très active**. Quel que soit le challenge technique que tu dois relever, il est probable que quelqu’un ait déjà réussi et si ce n’est pas le cas, tu peux compter sur l’aide d’un forum, comme Stackoverflow ou Reddit.

**La faiblesse de Python est sa lenteur**. Comparé à d’autres langages comme C ou Java, Python demande plus de temps d’exécution. Il reste possible d’accélérer la vitesse d’exécution de certaines manipulations avec certaines bibliothèques comme NumPy et Pandas (voir plus bas).

### **Que peut-on faire avec Python ?**

1. **Un langage puissant et polyvalent**

**Python a de nombreux domaines d’application**, contrairement à HTML ou JavaScript qui ne servent qu’au développement web.

Avec Python, on peut :

* créer et administrer un site Web
* développer des logiciels et des applications, aussi bien pour ordinateur que pour téléphone
* automatiser des scripts systèmes et des interactions ordinateur – navigateur Web.

Ainsi **Python**rend possible **l’écriture et l’exécution de programmes** à la complexité croissante. Tu peux tout autant programmer un court script chargé d’une tâche précise sur ton ordinateur, faire fonctionner en réseau plusieurs logiciels professionnels ou même développer un jeu vidéo… sur une calculatrice !

Chez Databird, nous apprenons à nos élèves, les “databirdies”, à :

* **Aspirer des données du web**, en utilisant les fonctionnalités de [web-scraping de Python.](https://www.data-bird.co/blog/web-scraping-python)
* **Rédiger des scripts** permettant de nettoyer et d’analyser une base de données.
* **Visualiser les résultats** sous forme graphique, en utilisant les fonctionnalités de data visualisation de Python.
* **Se connecter à des services web, en réalisant une**[API REST](https://www.data-bird.co/blog/api-rest)**avec Python.**

1. **Un langage complété par les bibliothèques Python**

Si Python est utilisable dans autant de domaines, c’est grâce à la richesse de ses bibliothèques (ou « *librairies*» en anglais).

Une [bibliothèque Python](https://www.data-bird.co/blog/bibliotheque-python) regroupe en un même endroit des fonctions qui ont une thématique commune. Ces fonctions n’existent pas dans le pack originel Python, mais elles ont été codées par des développeurs qui ont mis leur travail en open-source.

Les bibliothèques Python les plus utilisées par les Data Analyst sont :**‍**

* ‍**Pandas**, qui permet de manipuler et d’analyser des tables de données à la manière d’un « Excel sous stéroïdes ».**‍**
* **NumPy**, qui permet d’effectuer des calculs scientifiques (en particulier statistiques et probabilistiques).**‍**
* **Scikit-Learn** et **Tensorflow**, qui aident au développement de modèles de Machine Learning et de Deep Learning.**‍**
* **Scrapy**et **BeautifulSoup**, qui permettent d’extraire des données directement depuis le Web (ce qu’on appelle le « scraping »).**‍**
* **Seaborn**et **Matplotlib**, qui aident à la visualisation de données, en proposant notamment des outils de construction de graphiques.

Pour utiliser une bibliothèque, il te suffit de l’importer au début du programme (une ligne de code suffit). C’est très pratique car tu peux alors utiliser toutes les fonctions qu’elle contient.

**Remarque**: il est possible qu’il y ait parfois des doublons, c'est-à-dire que deux fonctions soient disponibles dans deux bibliothèques différentes et sous deux noms différents alors qu’elles servent à la même chose.

La richesse des bibliothèques Pythonpermet de repousser les limites du langage et de s’attaquer à des projets ambitieux et exigeants dans énormément de secteurs d’application. Par exemple, dans le domaine de la recherche scientifique, la bibliothèque biopython permet de traiter et d'analyser plus facilement des données biologiques. Dans le domaine des jeux vidéos, la bibliothèque pyGame est utilisée pour créer des jeux vidéo en 2D ou 3D.

Les bibliothèques Python participent ainsi aux deux forces principales du langage :

1. Simplicité d’utilisation,
2. Sa polyvalence.

Les formations de Databird te permettront de prendre en main les bibliothèques Python et d’apprendre à tirer parti de leurs complémentarités pour mener à bien tes projets data.

1. **Langage de référence du Machine Learning et de la Data science**

**Python et R** sont les langages les plus couramment associés aux tâches de **Machine Learning** et de **Data Science**. Alors, qui choisir de Python ou R ?



**Figure 28 :** python vs R

**R est un langage créé pour l’analyse statistique**. Sa syntaxe est plus adaptée que celle de Python pour résoudre des problèmes mathématiques matriciels et vectoriels. Toutefois, Python est mieux armé pour l'[analyse de données](https://www.data-bird.co/blog/analyse-de-donnees-definition), surtout lorsqu'elles sont en grandes quantités.   
Entre R vs Python, nous avons choisi d'enseigner le langage le plus simple à apprendre et le plus populaire au monde : Python. Lors de notre formation, les “databirdies” sont initiés au **Maching Learning avec Python.** Cette initiation inclut également des exercices pratiques avec [Dataiku](https://www.dataiku.com/" \t "_blank) : un fleuron français de la discipline. Pour en savoir plus sur le programme de la formation de Databird, télécharge le programme du bootcamp !

**Python**est aussi un **langage plus généraliste** que R : comme expliqué plus haut, il est possible d’administrer un site web avec Python, ce qui est inenvisageable avec R !

Grâce à la **richesse de ses bibliothèques**, Python est particulièrement bien adapté au **traitement de la donnée** : collecte avec BeautifulSoup, traitement et nettoyage avec Pandas, modélisation avec Scikit-Learn et Tensorflow, visualisation avec Matplotlib… pour donner quelques exemples.

Python est le langage le plus adapté au Big Data, c’est pourquoi il est devenu le langage le plus utilisé en Machine Learning. Si la Data Science t’intéresse, apprendre à lire et à coder en Python est quasiment incontournable.

### **Différents bibliothèques que nous avons utilisées dans notre projet**



#### **PytesserAct**



##### **C’est quoi Tesseract**

[Tesseract](https://github.com/tesseract-ocr/tesseract) est un moteur OCR open-source qui extrait le texte imprimé ou écrit des images. Il a été développé à l’origine par Hewlett-Packard, et son développement a ensuite été repris par Google. C’est pourquoi il est maintenant connu sous le nom de “Google Tesseract OCR”.

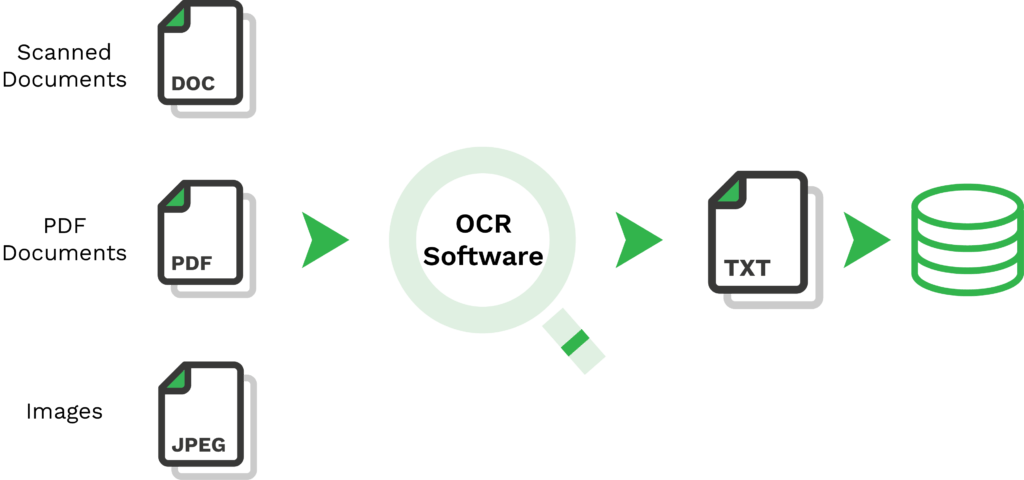
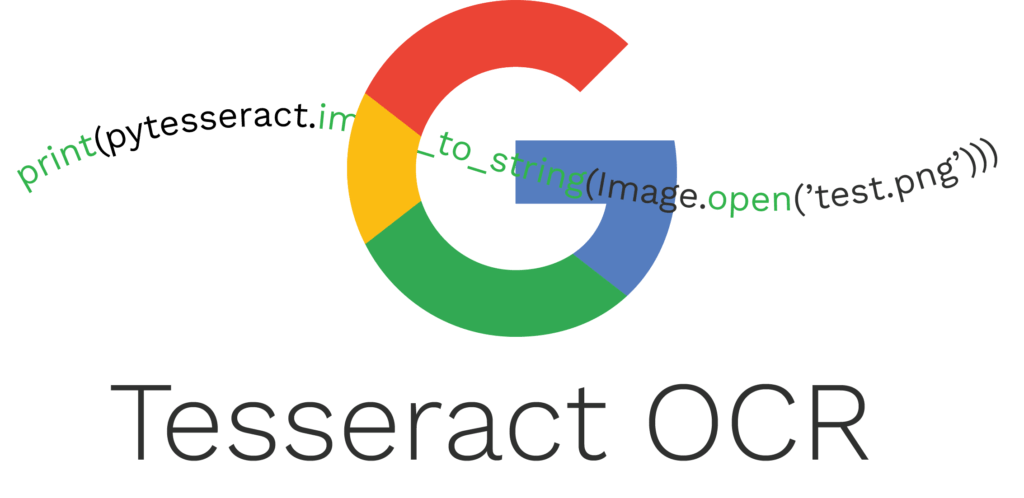


Figure 29 : moteur OCR

**Figure 18 :**

Mais qu’est-ce qu’un OCR ‘open-source’? Cela signifie simplement qu’il est disponible pour que chacun puisse l’utiliser librement, soit directement, soit en utilisant une interface de programmation d’applications (API). Avec l’OCR de Tesseract, les utilisateurs peuvent extraire du texte des images grâce à une reconnaissance efficace des lignes et des caractères du moteur d’OCR. À l’heure actuelle, Tesseract prend déjà en charge la reconnaissance linguistique pour plus de 100 langues “prêtes à l’emploi”. La version la plus récente de Tesseract (4.0) intègre l’IA par le biais du réseau neuronal LSTM pour mieux détecter et reconnaître les entrées de tailles diverses.L’une des grandes forces de Tesseract est qu’il est compatible avec de nombreux langages et cadres de programmation grâce à des wrappers tels que Pytesseract, également connu sous le nom de Python-Tesseract. Examinons de plus près cette connexion entre Tesseract OCR et Python.

**Pytesseract** n’est pas seulement un OCR en Python, un logiciel open-source ou une bibliothèque Python, mais sert également de wrapper pour le moteur OCR Tesseract de Google. Ce qu’il fait est d’envelopper le code Python autour de Tesseract OCR, assurant la compatibilité et la capacité de fonctionner avec différentes structures logicielles.



**Figure 30 :** moteur Tesseract de google

Notez qu’il existe d’autres bibliothèques et wrappers Python OCR qui peuvent être couplés avec Tesseract, notamment:

* **PYOCR** – permet plus d’options pour la détection des phrases, des chiffres et des mots
* **Textract** – permet l’extraction de données PDF pour les fichiers et paquets volumineux
* **OpenCV** – bibliothèque open-source de fonctions de programmation centrées sur la vision d’ordinateur (CV) en temps réel
* **Leptonica** – permet des fonctions de traitement d’images et des applications d’analyse d’images avec sa bibliothèque d’imagerie
* **Pillow** – une autre bibliothèque d’imagerie Python, qui prend en charge l’ouverture, la manipulation et l’enregistrement d’une liste étendue de formats de fichiers d’image

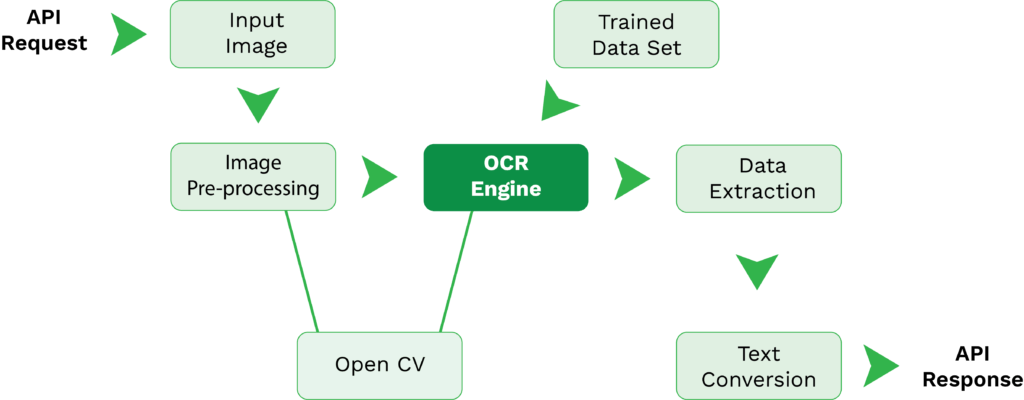
Maintenant que nous avons expliqué ce qu’est Tesseract et quel est son lien avec Python, voyons les étapes du processus d’OCR de Tesseract.



##### **Étapes du processus d’OCR de Tesseract**

Pour vous aider à comprendre à quoi ressemble normalement le processus d’OCR de Tesseract, nous l’avons décomposé en plusieurs étapes suivantes:

1. **Demande d’API**– L’accès à Tesseract OCR ne peut se faire que par l’intégration d’une API. Une fois que la connexion entre votre solution et Tesseract est établie, vous pouvez envoyer des requêtes API de votre solution au moteur OCR de Tesseract.
2. **Image d’entrée** – Avec une demande API, vous pouvez envoyer votre image d’entrée pour l’extraction de texte.
3. **Pré-traitement de l’image** – Avant l’extraction des données, les fonctions de pré-traitement de l’image du moteur OCR de Tesseract entrent en jeu. Cette étape existe pour s’assurer que la qualité de l’image est aussi élevée que possible pour obtenir des résultats d’extraction de données précis. OpenCV est souvent couplé à Tesseract pour améliorer la qualité de l’image avant l’extraction des données.
4. **Extraction des données** – Avec des ensembles de données entraînés et Leptonica ou OpenCV, le moteur OCR de Tesseract traite l’image d’entrée et extrait les données.
5. **Conversion du texte** – Les données (texte) ayant été extraites de l’image d’entrée, elles peuvent maintenant être converties dans un format souhaité pris en charge par Tesseract, notamment PDF, texte brut, HTML, TSV et XML.
6. **Réponse API** – Une fois que la sortie est prête, votre solution recevra une réponse API avec la sortie finalisée.



**Figure 31 :** Processus de tesseract



##### **Amélioration du traitement des images en combinant OpenCV et Tesseract**

Pour comprendre pourquoi OpenCV est souvent combiné avec l’OCR de Tesseract, nous devons expliquer la vision de l’ordinateur. La vision par ordinateur est un sous-domaine de l’intelligence artificielle (IA) qui permet aux ordinateurs et aux logiciels de voir, d’observer et de comprendre des images numériques, des vidéos ou d’autres données visuelles. Mais qu’est-ce que cela a à voir avec OpenCV?

OpenCV est une bibliothèque open-source de fonctions de vision par ordinateur qui peut améliorer l’extraction de données des moteurs OCR tels que Tesseract. Pour ce faire, vous pourriez utiliser la bibliothèque OpenCV pour intégrer les fonctions suivantes dans la solution OCR:

* **Détection d’objets** – permet à la solution de détecter une grande variété d’objets
* **Réseaux neuronaux profonds (DNN)** – permet à la solution de classer les images
* **Traitement de l’image** – permet à la solution de mieux traiter les images d’entrée grâce à diverses techniques telles que la détection des bords, la manipulation des pixels, le redressement, etc.

Sans OpenCV, Tesseract n’est pas aussi sophistiqué que ce que l’on pourrait attendre des solutions d’OCR actuelles, car beaucoup d’entre elles appliquent diverses technologies d’IA.

Maintenant que vous savez que l’OCR Tesseract peut être amélioré avec d’autres bibliothèques de fonctions de programmation telles qu’OpenCV, examinons de plus près l’un des wrappers Tesseract les plus utilisés en Python: PyTesseract.



##### **Comment fonctionne (Py)Tesseract?**

Jusqu’à présent, nous savons que Pytesseract est un wrapper pour l’OCR Tesseract de Google en Python avec des fonctionnalités supplémentaires que Tesseract seul n’a pas. Quelles sont donc ces fonctionnalités et comment cela fonctionne-t-il?

Pytesseract peut être utilisé comme un script autonome pour Tesseract ce qui lui permet d’imprimer le texte reconnu au lieu de le convertir dans un fichier. Pytesseract peut lire tous les fichiers images supportés par les bibliothèques d’imagerie telles que Leptonica et Pillow, y compris JPEG, PNG, GIF, BMP, TIFF, et beaucoup d’autres. Il est donc souvent utilisé dans les cas d’utilisation de l’OCR Python de l’image au texte.

Le fonctionnement de Pytesseract est le suivant: il convertit les éléments textuels et graphiques d’une image numérisée en un bitmap. Ce bitmap est simplement une construction de points blancs et noirs. Comme pour toute OCR, l’image passe par une phase de prétraitement pour les ajustements de luminosité et de contraste avant l’extraction et la conversion des données.

Le framework Pytesseract est optimisé pour une meilleure détection de la langue, ce qui profite également à l’OCR Tesseract de Google. En outre, ce cadre est excellent pour détecter les polices de caractères utilisées et l’orientation du texte sur l’image d’entrée. Par exemple, il peut fournir un chiffre de confiance d’orientation pour assurer la détection de l’orientation. Cependant, l’une de ses caractéristiques les plus importantes est qu’il peut vous fournir des informations sur la boîte de contour de l’OCR.



##### **Limite de Tesseract**

L’OCR de Tesseract peut être très utile dans de nombreuses situations et cas d’utilisation. Cependant, comme toute autre solution open-source, il y a toujours des inconvénients à prendre en compte. Dans cette section, nous allons éclairer ces limitations une par une:

* Tesseract n’est pas aussi précis que des solutions plus avancées intégrant de l’IA
* Tesseract est sujet à des erreurs si la séparation entre le premier plan et l’arrière-plan de l’image n’est pas importante
* Vous avez besoin de beaucoup de ressources et de temps pour développer votre propre solution à l’aide de l’OCR de Tesseract
* Tesseract ne prend pas en charge tous les formats de fichiers par lui-même
* Tesseract ne reconnaît pas l’écriture manuscrite
* La qualité de l’image doit atteindre un certain seuil de points par pouce (DPI) pour qu’elle fonctionne
* Tesseract doit être développé davantage et nécessite l’intégration de l’IA pour pouvoir automatiser certains processus documentaires (par exemple, la vérification, la validation par recoupement, etc.)
* Tesseract n’a pas d’interface utilisateur graphique (GUI), ce qui signifie que vous devez le connecter à votre GUI existante ou en faire développer une
* Le développement supplémentaire vous coûtera du temps et de l’argent



#### **OpenCV — WikipédiaOpenCV**

OpenCV est l’énorme bibliothèque open source pour la vision par ordinateur, l’apprentissage automatique et le traitement d’images et joue maintenant un rôle majeur dans le fonctionnement en temps réel, ce qui est très important dans les systèmes d’aujourd’hui. En l’utilisant, on peut traiter des images et des vidéos pour identifier des objets, des visages, ou même l’écriture d’un humain. Lorsqu’il est intégré à diverses bibliothèques, telles que NumPy, python est capable de traiter la structure de tableau OpenCV pour l’analyse. Pour identifier le modèle d’image et ses diverses caractéristiques, nous utilisons l’espace vectoriel et effectuons des opérations mathématiques sur ces caractéristiques.

La première version d’OpenCV était 1.0. OpenCV est publié sous licence BSD et est donc gratuit pour un usage académique et commercial. Il a des interfaces C++, C, Python et Java et prend en charge Windows, Linux, Mac OS, iOS et Android. Lorsque OpenCV a été conçu, l’accent était mis sur les applications en temps réel pour l’efficacité informatique. Toutes les choses sont écrites en C/C++ optimisé pour tirer parti du traitement multicœur.



##### **Les applications d’OpenCV**

* reconnaissance faciale
* Inspection et surveillance automatisées
* nombre de personnes – compté (circulation à pied dans un centre commercial, etc.)
* Véhicule comptant sur les autoroutes avec leurs vitesses
* Installations artistiques interactives
* Détection d’anomalie (défaut) dans le processus de fabrication (les produits défectueux impairs)
* Assemblage d’images Street View
* Recherche et récupération de vidéos et d’images
* Navigation et contrôle sans robot et sans conducteur
* reconnaissance d’objets
* Analyse d’images médicales
* Films – Structure 3D à partir du mouvement
* Reconnaissance publicitaire des chaînes de télévision



##### **Les fonctionnalites d’openCV**

* Image/vidéo E/S, traitement, affichage (core, imgproc, highgui)
* Détection d’objets/fonctionnalités (objdetect, features2d, nonfree)
* Vision par ordinateur monoculaire ou stéréo basée sur la géométrie (calib3d, couture, videostab)
* Photographie computationnelle (photo, vidéo, superres)
* Apprentissage automatique et clustering (ml, flann)
* Accélération CUDA (gpu)



##### **Image processings**

Le traitement d’image est une méthode pour effectuer certaines opérations sur une image, afin d’obtenir une image améliorée et ou d’en extraire des informations utiles.

Si nous parlons de la définition de base du traitement d’image, alors « le traitement d’image est l’analyse et la manipulation d’une image numérisée, surtout pour améliorer sa qualité ».

**Image numérique :**

Une image peut être définie comme une fonction bidimensionnelle f(x, y), où x et y sont des coordonnées spatiales(plan), et l’amplitude de la graisse toute paire de coordonnées (x, y) est appelée l’intensité ou le niveau de gris de l’image à ce point.

Dans un autre mot Une image n’est rien de plus qu’une matrice bidimensionnelle (3-D dans le cas d’images colorées) qui est définie par la fonction mathématique f(x, y) à n’importe quel point donne la valeur de pixel à ce point d’une image, la valeur de pixel décrit à quel point ce pixel est lumineux, et de quelle couleur il devrait être.

Le traitement d’image est essentiellement le traitement du signal dans lequel l’entrée est une image et la sortie est une image ou des caractéristiques selon les exigences associées à cette image.

**Le traitement des images comprend essentiellement les trois étapes suivantes :**

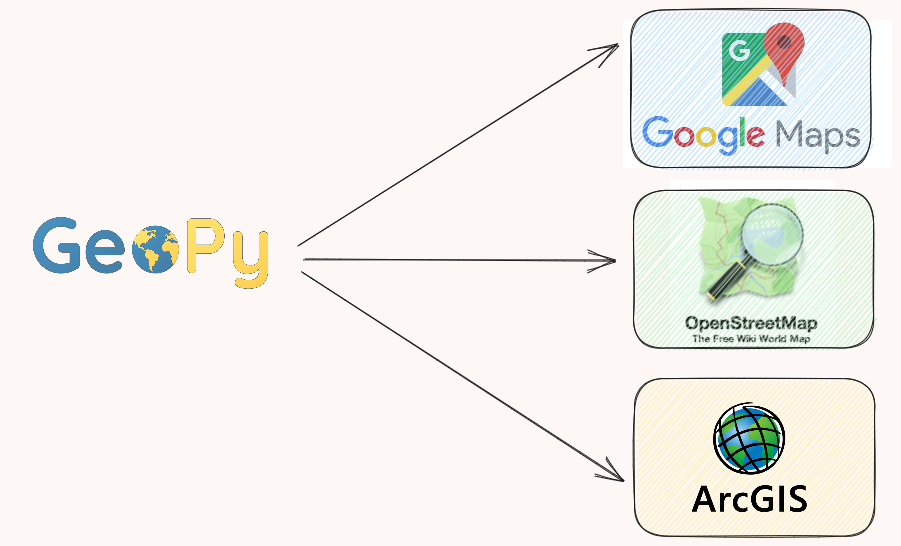
* 1. Importation de l’image
  2. Analyser et manipuler l’image
  3. Sortie dans laquelle le résultat peut être modifié image ou rapport qui est basé sur l’analyse d’image



#### **Geopy**

GeoPy est une bibliothèque Python open-source conçue pour faciliter le traitement géographique et la manipulation des données de localisation. Elle offre des fonctionnalités de géocodage (conversion d'adresses en coordonnées géographiques et inversement), de calcul de distances entre points géographiques, de recherche de lieux et d'adresses, ainsi que d'autres opérations liées à la géolocalisation.

Cette bibliothèque utilise des services de géocodage en ligne tels que Google Maps API, OpenStreetMap, Bing Maps API, etc., pour obtenir des informations géographiques précises et fournir des fonctionnalités de géolocalisation robustes. Elle permet aux développeurs d'intégrer facilement des fonctionnalités de géolocalisation dans leurs applications Python, ce qui est utile dans de nombreux domaines tels que la cartographie, la logistique, les applications basées sur la localisation, etc.



**Figure 32 :** les applications de geopy

## JAVA

Java est un langage de programmation orientée objet développé par Sun Microsystems en 1995, et racheté depuis par [Oracle](https://www.futura-sciences.com/tech/definitions/informatique-oracle-19087/). Le principal avantage de Java est son [interopérabilité](https://www.futura-sciences.com/tech/dossiers/technologie-avenir-univers-virtuels-776/page/4/) : la technologie fonctionne aussi bien sur Windows que Mac ou [Linux](https://www.futura-sciences.com/tech/actualites/cybersecurite-fbi-nsa-decouvrent-russes-visent-desormais-linux-82563/), et sur une [myriade](https://www.futura-sciences.com/sciences/definitions/univers-myriade-3805/) d'appareils : [centres de données](https://www.futura-sciences.com/tech/definitions/informatique-data-center-15675/), [ordinateur](https://www.futura-sciences.com/tech/definitions/informatique-ordinateur-586/), téléphone mobile, lecteur Blu-ray, périphériques TV, [consoles de jeux](https://www.futura-sciences.com/tech/comparatifs/meilleure-console-de-jeu-comparatif/), appareils connectés... Un autre avantage est son caractère universel : le même système peut être utilisé pour une grande variété d'[applications](https://www.futura-sciences.com/sciences/definitions/mathematiques-application-13200/). Le langage Java est basé sur le C++, mais avec une approche simplifiée et des fonctionnalités plus avancées.

### **Applications de java**

La technologie Java est très répandue et à la base de la plupart des applications en réseau. Elle est notamment au cœur du [système d'exploitation](https://www.futura-sciences.com/tech/definitions/informatique-systeme-exploitation-11820/) Android, et exploitée dans le monde entier pour développer des applications mobiles, des jeux, du contenu Web ou des logiciels d'entreprise. Le [langage Java](https://www.futura-sciences.com/tech/questions-reponses/internet-meilleurs-langages-developpement-moment-10578/) permet notamment de développer des forums, des formulaires en ligne, d'écrire des applications, ou créer des programmes exécutables dans un [navigateur](https://www.futura-sciences.com/tech/definitions/internet-navigateur-3986/) web.

Le langage Java a d'ailleurs construit sa notoriété sur les applets, des petits programmes animés qui s'exécutent dans un logiciel de navigation (ce type d'applet a depuis été largement remplacé par le [Flash](https://www.futura-sciences.com/tech/definitions/internet-flash-476/), puis par des standards ouverts comme HTML5, WebGL et WebAssembly). Attention : il ne faut pas confondre le [JavaScript](https://www.futura-sciences.com/tech/definitions/internet-javascript-509/), qui fonctionne uniquement sur les navigateurs Internet, et le Java en général qui peut être utilisé n'importe où.



### **Plateformes de java**

Les plateformes Java sont des collections de programmes permettant de développer des applications. Il en existe plusieurs, dont le Java SE (Standard Edition), destiné aux applications pour les postes de travail. Le JavaFX, lancé en 2014, est quant à lui une bibliothèque de packages graphiques et multimédia écrits en code Java et permettant aux développeurs de créer et de déployer facilement des applications Internet contenant par exemple des vidéos, de la musique, des effets graphiques, etc.



### **Avantage de JAVA**

Java présente des avantages significatifs par rapport à d'autres langages et environnements qui le rendent adapté à n'importe quelle tâche de programmation.

Les avantages de Java sont les suivants:

* Java est facile à apprendre.

Java a été conçue pour être facile à utiliser et est donc facile à écrire, compiler, déboguer et apprendre que d'autres langages de programmation.

* Java est orienté objet.

Cela vous permet de créer des programmes modulaires et du code réutilisable.

* Java est indépendante de la plateforme.

L'un des avantages les plus importants de Java est sa capacité à se déplacer facilement d'un système informatique à un autre. La possibilité d'exécuter le même programme sur de nombreux systèmes différents est cruciale pour le logiciel World Wide Web, et Java y parvient en étant indépendant de la plateforme aux niveaux source et binaire.

En raison de la robustesse, de la facilité d'utilisation, des capacités multiplateformes et des fonctions de sécurité, de Java, il est devenu un langage de choix pour fournir des solutions Internet partout dans le monde.

Parmi les Framework utilises pour manipuler et implémenter les systèmes multi agents est JADE qui est un middleware qui facilite le développement des [systèmes multi agents (SMA)](http://fr.wikipedia.org/wiki/Syst%C3%A8me_multi-agents)



### **JADE ( java agent development framwork)**

JADE (Java Agent DEvelopment Framework) est une plateforme open source pour les applications basées sur des agents peer-to-peer.

JADE simplifie la mise en œuvre de systèmes multi-agents grâce à un middleware conforme aux spécifications FIPA pour les systèmes multi-agents intelligents interopérables.

JADE fournit un modèle d’exécution et de composition de tâches simple mais puissant, une communication d’agent pair à pair basée sur le passage de messages asynchrones, un service de pages jaunes prenant en charge le mécanisme de découverte des abonnements et de nombreuses autres fonctionnalités avancées qui facilitent le développement de systèmes distribués.

Un système basé sur JADE peut être distribué sur Internet et peut déployer des agents sur des appareils Android et J2ME-CLDC MIDP 1.0. La configuration de la plate-forme ainsi que le numéro d’agent et les emplacements peuvent être modifiés à l’exécution, selon les besoins. En outre, des configurations appropriées peuvent être spécifiées pour exécuter des agents JADE dans des réseaux caractérisés par une connectivité partielle, y compris NAT et pare-feu, ainsi que des modifications intermittentes de couverture et d’adresse IP.



#### **Caractéristiques**

Ce système présente les fonctionnalités suivantes :

* Support de l'envoi de messages, compatible avec plusieurs protocoles, permettant la diffusion d'événements en local.
* Utilisation de Java RMI pour la diffusion interne à une plateforme et du protocole FIPA 2000 Message Transport Protocol.
* Adoption d'un modèle de concurrence à deux niveaux : entre agents (utilisation de Threads Java avec préemption) et interne aux agents (utilisation de classes de comportements "behaviour" de manière coopérative).
* Mobilité des agents, offrant une faible mobilité entre plateformes.
* Implémentation en Java d'un framework orienté objet, conforme aux spécifications FIPA, incluant une plateforme agent, un langage de communication entre agents, une ontologie de gestion des agents, des protocoles d'interaction standard ainsi que des langages définis par les utilisateurs et les ontologies.
* Encapsulation des spécifications de la FIPA par JADE (Java Agent DEvelopment Framework), fournissant des éléments tels qu’AMS (Agent Management System), DF (Directory Facilitator), MTS (Message Transport System) et ACC (Agent Communication Channel). L'enregistrement d'un agent dans la plateforme se fait via un constructeur d'agents, avec des informations telles que le nom et l'adresse. La classe DFService permet l'accès au DF pour la recherche d'agents.
* Système de gestion d'événements dans le noyau de la plateforme, offrant des fonctionnalités d'observation de la plateforme, des messages, du transport de messages et des agents.
* Outils de gestion basés sur des agents spéciaux (RMA, Sniffer, Introspector) permettant la communication avec FIPA ACL. Il existe des extensions à l'ontologie fipa-management-ontology pour inclure des actions spécifiques, ainsi qu'une ontologie dédiée pour l'observation jade-introspection.
* Agents utilitaires comme le DummyAgent tool pour permettre aux utilisateurs d'interagir avec les agents déployés sur la plateforme, ainsi que le Sniffer Agent utilisé pour observer les messages.

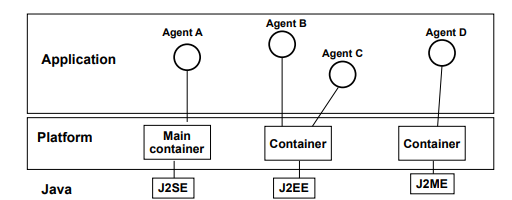
#### **Platforme de JADE**

• Une application JADE se présente sous la forme d'une plateforme déployée sur une ou plusieurs machines.

• Cette plateforme héberge un groupe d'agents, chacun ayant une identité unique et capable de communiquer de manière bidirectionnelle avec les autres agents.

• Chaque agent fonctionne à l'intérieur d'un conteneur dédié (container) qui lui fournit son environnement d'exécution. Il a également la capacité de se déplacer à l'intérieur de la plateforme.

• Chaque plateforme doit être dotée d'un conteneur principal, responsable de l'enregistrement des autres conteneurs présents dans le système.

• Ainsi, une plateforme se compose d'un ensemble de conteneurs actifs, chacun jouant un rôle spécifique dans le fonctionnement global de la plateforme.****

**Figure 33 :** explication de platforme jade

#### **Présentation de l’architecture de JADE**

La représentation architecturale d'une plateforme JADE est présentée dans la figure 31 ci-dessous. Cette plateforme se compose d'un ensemble de conteneurs, qui sont des instances de JADE. Ces conteneurs peuvent être situés sur une seule machine ou être distribués sur plusieurs machines interconnectées au sein d'un réseau.

Chaque instance ou conteneur a la capacité d'accueillir plusieurs agents. Dans le cadre de JADE, chaque plateforme est caractérisée par la présence d'un conteneur principal, désigné dans ce document comme MC (Main Container). Ce conteneur principal joue un rôle crucial, étant le point central où tous les autres conteneurs se connectent et s'enregistrent lors de leur démarrage. Chaque conteneur est identifié par un nom logique qui lui est propre et qui lui est associé pour une identification précise au sein de la plateforme.



**Figure 34 :** Architecture d’une plateforme JADE

# **Conclusion**

Cette section de notre rapport a porté sur les divers outils et technologies utilisés pour mettre en place notre système multi-agents, depuis la phase d'analyse jusqu'à la conception. Dans le cadre de la conception et de la mise en œuvre de notre système, notre choix s'est porté sur l'utilisation de JADE. Nous avons sélectionné JADE car nous l'avons identifié comme l'une des plateformes ou frameworks de développement et de gestion d'applications SMA les plus performants, répondant aux normes établies par la FIPA.

La partie suivante présente quelques résultats de nos implémentations et tests.

# **Implémentation et résultats**

## Introduction

Dans cette partie nous allons présenter nos différentes expérimentations en commençant par décrire les conditions ainsi que les données et outils ayant servi à leur réalisation. La comparaison des résultats de nos expérimentations aux résultats attendus ou aux résultats obtenus dans d’autres conditions ou des suites d’autres travaux de recherche utilisant d’autres approches, nous permettra de juger de la qualité et de la pertinence de nos solutions.

Nous allons dans un premier temps tester nos fonctions de calcul de trajet et une fois rassuré de leur robustesse nous passerons à la mise en place d’un jeu de données devant nous permettre de mettre en place nos fonctions de notre système de notre approche évidentielle.

Nous allons analyser nos résultats et les comparer à celles obtenus dans un de nos travaux antérieurs et ceux d’autres chercheurs.



## Framework et environnement de développement

Les différents frameworks et environnements de développement utilisés sont

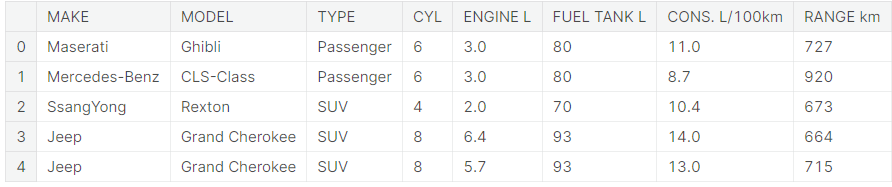
* OpenCV, bibliothèque graphique libre pour le traitement d’images et de vidéos.
* Eclipse, notre environnement de programmation en java
* Visuel studio code environnement de programmation de python
* JADE, framework open source de développement et de gestion de systèmes multi-agents.
* Geopy, bibliotheque de google maps api.
* Matplotlib, bibliothèque de visualisation des donnes
* JAVA Swing, une bibliothèque graphique pour le langage de programmation Java. Elle offre un ensemble d'outils et de composants permettant de créer des interfaces utilisateur (UI) pour des applications de bureau. Swing fait partie de la plateforme Java Standard Edition (Java SE).

## Donnes utilisés

### **Donnes des véhicules :**

Nous allons commencer par la collection des donnes des véhicules, Le but de ça est d’aider les gens qui veulent prendre un véhicule pour un voyage touristique par exemple. Par conséquent, nous prenons ces données du site Web de Kaggle (source de données : https://www.kaggle.com/datasets/adityamahimkar/car-driving-distance-range-dataset). Dans cet ensemble de données, nous connaissons les informations sur la distance totale parcourue par différents types (marques) de véhicules. Ainsi, nous pouvons facilement choisir un véhicule bénéfique pour les gens. De plus, nous comprenons ces fonctionnalités du site Kaggle et de divers sites Web. Après cela, nous traitons les valeurs manquantes et les types de données manquantes. Enfin, nous effectuons une analyse exploratoire des données (EDA) avec cet ensemble de données. Nous obtenons donc des informations importantes à partir de ces données.

Nous pouvons décider quel véhicule est le meilleur pour les deux véhicules de type carburant. En outre, nous obtenons des informations sur le type de véhicule le plus utilisé pour les deux carburants. Par conséquent, les gens obtiennent des informations importantes selon le type de véhicule (passager, SUV, etc.). Enfin, nous pouvons dire quel véhicule est le plus avantageux pour eux. Essentiellement, chaque personne a des besoins différents. Certaines personnes veulent le véhicule le plus puissant, et d’autres veulent moins cher. Donc, nous intégrons les donnes de véhicules dans notre système pour ces points.



**Figure 35 :** dataframe des donnes des véhicules

Il a été observé que les données des véhicules comprennent à la fois des données numériques, telles que la capacité du moteur (Engine L) et la capacité du réservoir de carburant, ainsi que des données discrètes, notamment les modèles de véhicules et leurs types.

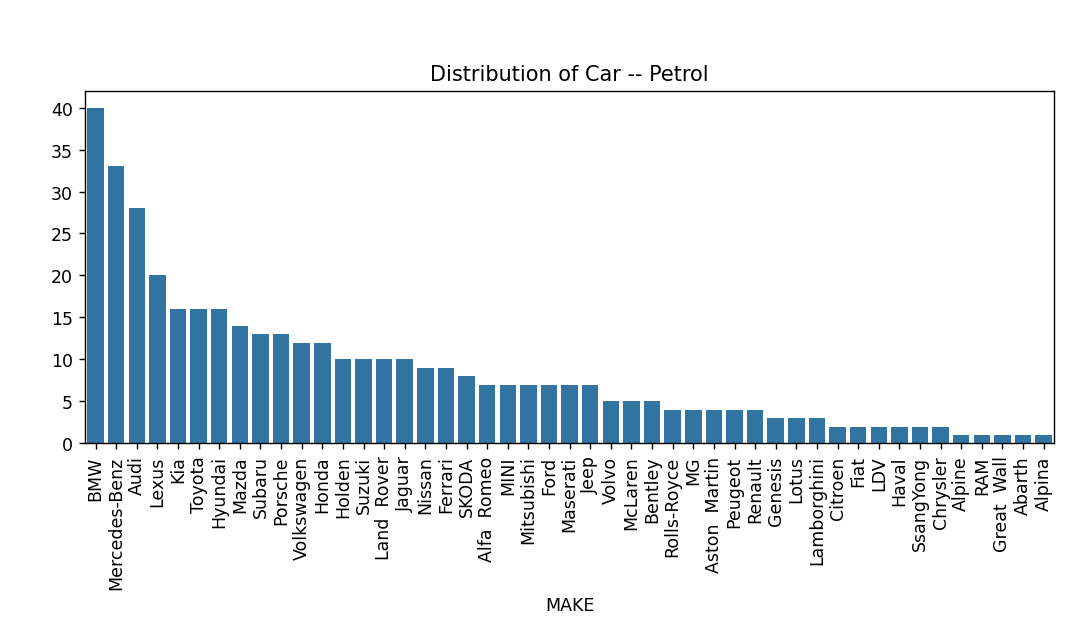
Il y a une variété de données dans le jeu de données des véhicules, certaines étant quantitatives et mesurables (telles que les attributs numériques comme la capacité du moteur et du réservoir), tandis que d'autres sont qualitatives et catégoriques (comme les noms de modèles et les types de véhicules).

Après avoir nettoyé les données relatives aux véhicules et effectué des conversions de types nécessaires, nous avons affiché les graphiques suivants :

1- Un graphique en secteurs (pie chart) montrant la répartition des types de véhicules dans l'ensemble des données, mettant en évidence la part de chaque type de véhicule par rapport à l'ensemble.

2- Un diagramme à barres représentant la distribution des modèles de véhicules, illustrant la fréquence de chaque modèle dans l'ensemble des données.

Chaque graphique vise à fournir une visualisation claire et informative des caractéristiques et des tendances importantes présentes dans les données nettoyées des véhicules. Ces représentations graphiques peuvent aider à mieux comprendre les différentes caractéristiques et relations au sein de l'ensemble de données des véhicules après les étapes de nettoyage et de conversion de types.



**Figure 36 :** histogrammedistribution de voitures

D’après les données fournies, BMW et Mercedes-Benz semblent avoir la plus grande distribution de voitures, avec Audi et Lexus approchant. D’autres marques automobiles, telles que Kia, Toyota et Hyundai, ont également une bonne présence sur le marché.

Toutefois, les données fournies sont insuffisantes pour permettre une analyse exhaustive. Pour une analyse exhaustive, des renseignements supplémentaires, comme :

Pourcentage de modèles automobiles produits par chaque fabricant d’essence

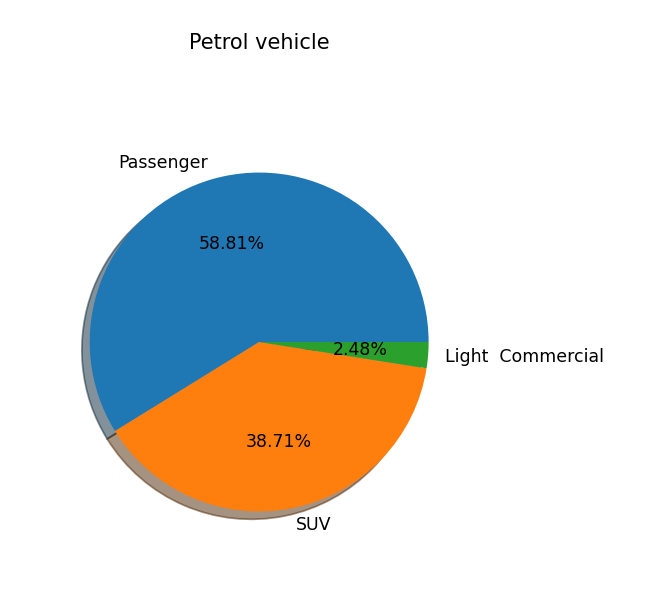
Pourcentage de modèles de voitures produits par chaque fabricant de diesel

Pourcentage de modèles de voitures produits par chaque fabricant électrique

Nombre total de modèles automobiles produits par chaque constructeur l’année dernière

Part de marché de chaque usine dans l’ensemble de l’industrie automobile

En considérant ces informations supplémentaires, nous pouvons mieux comprendre la part de marché et les performances de chaque constructeur automobile. En outre, les données des dernières années et des régions du monde peuvent fournir des informations précieuses sur les tendances et les stratégies à long terme de l’industrie automobile.



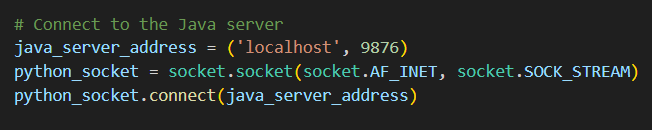
**Figure 37 :** diagramme circulaore de la distribution des vehicules

Le résultat de l'analyse indique que la distribution des véhicules dans le parking est assez uniforme. On observe une répartition significative de voitures particulières (Passenger) et de SUV (Sport Utility Vehicle), ce qui est assez logique pour un parking de supermarché.

Parmi les autres véhicules, on remarque une part assez importante de véhicules légers de commercialisation (Light Commercial) qui est peut-être utilisée par les employés du supermarché pour leurs courses et déplacements professionnels.

Il est important de noter que les chiffres fournis ne tiennent pas compte des changements saisonniers et des heures de fonctionnement du supermarché, qui pourraient affecter la répartition des véhicules.

Ensuite, après avoir converti notre DataFrame en un dictionnaire Python, nous l'avons transformé en format JSON avant de l'envoyer à l'agentOne à l’aide des sockets pour effectuer les traitements requis.



Cette conversion vers un dictionnaire Python a été entreprise pour faciliter la manipulation des données dans un format adapté à la communication avec l'agentOne. Ensuite, en transformant ce dictionnaire en format JSON, nous avons utilisé cette structure de données standard pour transmettre les informations de manière efficace et organisée à l'agentOne. Cette transmission de données sous forme de fichier JSON permet à l'agentOne de traiter, analyser ou effectuer toute opération spécifique nécessaire sur ces données, dans le contexte de notre application ou système.



### **Données des villes**

Nous avons utilisé des captures d'écran de Google Maps pour extraire les données essentielles de chaque ville. Cela incluait des informations telles que les noms, les latitudes et longitudes des villes ainsi que les niveaux de congestion du trafic. Ces données ont été extraites des captures d'écran, puis transformées en format JSON avant d'être transmises à l'agentTwo.

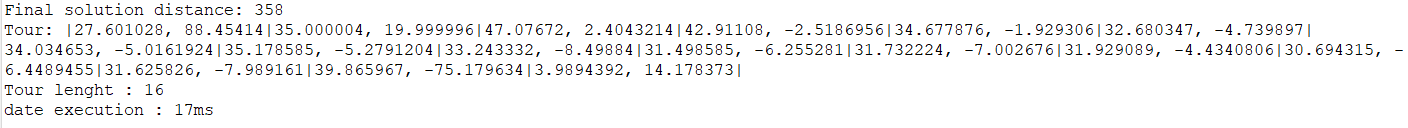
L'objectif de cette démarche était de récupérer des détails spécifiques sur les villes, notamment leurs emplacements géographiques (coordonnées de latitude et longitude) et les niveaux de congestion du trafic, en utilisant les captures d'écran de Google Maps comme source d'informations visuelles. En convertissant ces informations en format JSON, nous avons rendu les données facilement interprétables par l'agentTwo, lui permettant ainsi d'effectuer des analyses ou des opérations spécifiques basées sur ces détails géographiques et de trafic pour chaque ville.



**Figure 38 :** capture d’écran du maroc.

## Résultats du système

Dans le précédent chapitre, nous avons abordé les différentes étapes d'un algorithme visant à déterminer le chemin le plus court entre des villes, en se basant sur des captures d'écran de Google Maps. Lors de l'implémentation de cet algorithme, nous avons obtenu des résultats indiquant une distance minimale ainsi qu'un trajet plus court entre ces localités.

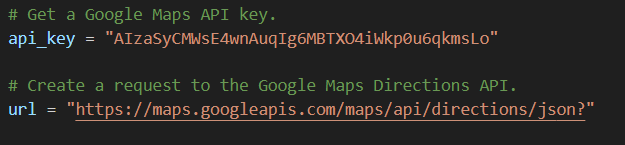


En utilisant l'API Directions de Google Maps est un service fourni par Google qui permet aux développeurs d'accéder et d'intégrer des informations de routage, de calcul de matrices de distance et de navigation dans leurs applications. Elle fournit des instructions détaillées pour se déplacer entre différents lieux en utilisant différents modes de transport tels que la conduite, la marche, le vélo ou les transports en commun.

L'API offre plusieurs fonctionnalités :

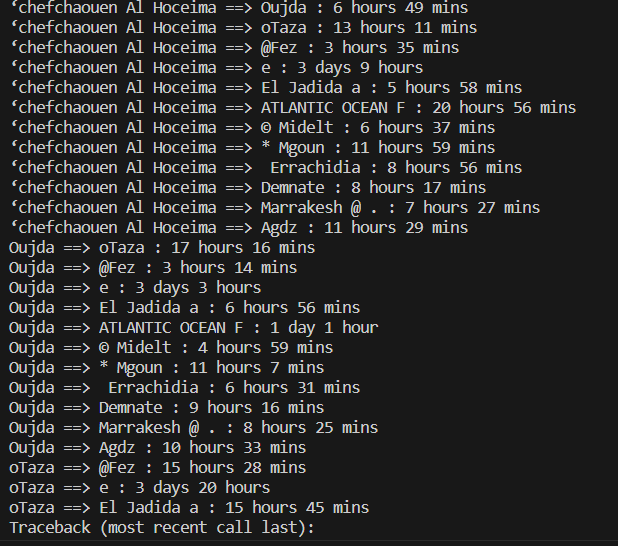
* **Obtenir des Directions :** Obtenir des instructions détaillées entre des lieux spécifiques, y compris des étapes à suivre, des distances, des durées et plus encore.
* **Spécifier les Itinéraires et les Modes de Transport** : Les développeurs peuvent définir des points de passage, optimiser les itinéraires et spécifier différents modes de transport (conduite, marche, vélo, transport en commun) pour le routage.
* **Informations sur le Trafic :** Accéder aux données de trafic en temps réel et estimer les temps de trajet en fonction des conditions de circulation actuelles (si disponibles dans la région demandée).
* **Calcul de Matrices de Distance :** Calculer les distances de déplacement et les temps entre plusieurs origines et destinations, utile pour des scénarios tels que la logistique ou la recherche de lieux à proximité.

Pour utiliser l'API Directions de Google Maps, les développeurs doivent obtenir une clé API auprès de la plateforme Google Cloud et l'utiliser dans leurs applications avec les points de terminaison (endpoints) API et les paramètres appropriés.



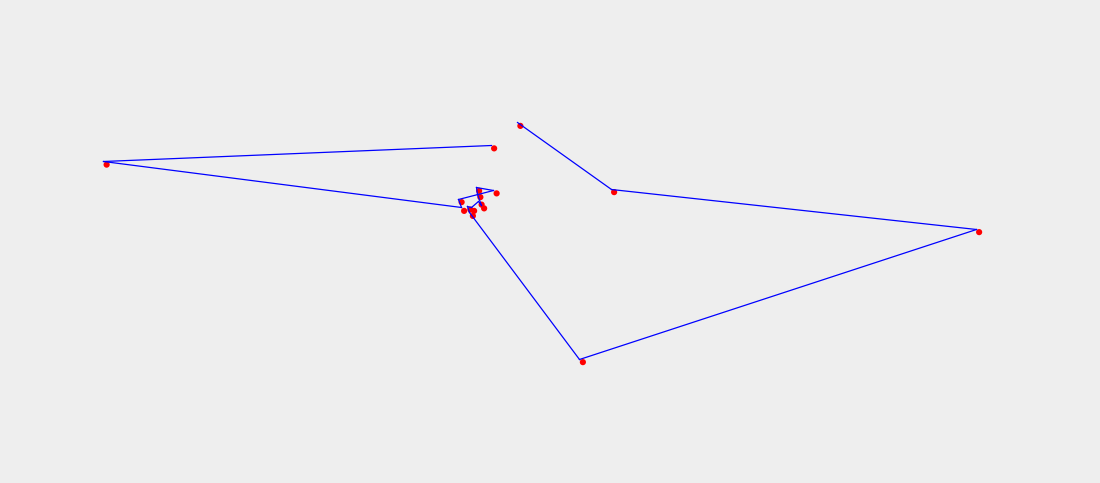
Les développeurs intègrent souvent cette API dans diverses applications telles que les applications de covoiturage, les planificateurs de voyage, les solutions logistiques, etc., afin de fournir aux utilisateurs des informations précises sur les itinéraires et une assistance à la navigation.

Nous avons pu obtenir des informations sur les niveaux de congestion du trafic dans chaque ville en spécifiant le mode de transport conduite.



**Figure 39 :** Congestion de trafic de chaque ville

En se basant sur les données des véhicules et la distance obtenue, il est possible de prédire l'ensemble de véhicules approprié pour ce trajet spécifique.



**Figure 40 :** Résultat du système

En utilisant Java Swing, nous avons développé une interface graphique pour visualiser le trajet le plus efficace calculé par notre algorithme. Cela nous a permis d'évaluer l'efficacité de notre algorithme en générant des itinéraires plus simples que les routes initiales, offrant ainsi un aperçu de sa capacité à optimiser les trajets.

Nous voulions trouver la voiture dans une liste des véhicules qui nécessite le moins d'arrêts de ravitaillement pour parcourir une distance calculé par notre algorithme SA en divisant la distance finale par le maximum distance de chaque vehicule

Nous avons obtenus les résultats suivants de chaque véhicule :

distanceRef pour Maserati : 1.4916667

distanceRef pour Mercedes-Benz : 1.4916667

distanceRef pour SsangYong : 2.557143

distanceRef pour Jeep : 0.60147846

distanceRef pour Jeep : 0.6753443

distanceRef pour Lexus : 1.0848485

distanceRef pour Jaguar : 0.87317073

distanceRef pour Land Rover : 0.87317073

distanceRef pour Land Rover : 0.68846154

distanceRef pour BMW : 1.196524

distanceRef pour BMW : 1.196524

distanceRef pour BMW : 0.98028475

distanceRef pour BMW : 0.98028475

distanceRef pour BMW : 0.98028475

distanceRef pour BMW : 0.98028475

distanceRef pour Mercedes-Benz : 1.0529412

distanceRef pour Porsche : 0.99444443

distanceRef pour Mercedes-Benz : 1.11875

distanceRef pour Maserati : 1.1776316

distanceRef pour Lexus : 1.5497836

distanceRef pour Lexus : 1.4206349

distanceRef pour Maserati : 1.4916667

distanceRef pour Audi : 1.8941799

distanceRef pour Audi : 1.8941799

distanceRef pour Land Rover : 1.4552846

distanceRef pour Mercedes-Benz : 1.4916667

distanceRef pour Jaguar : 1.6126126

distanceRef pour Mercedes-Benz : 1.4916667

distanceRef pour Mercedes-Benz : 1.4916667

distanceRef pour BMW : 1.437751

distanceRef pour Jaguar : 1.4552846

distanceRef pour BMW : 1.437751

distanceRef pour Porsche : 1.325926

distanceRef pour Porsche : 1.3716475

distanceRef pour Haval : 2.2375

distanceRef pour LDV : 2.3866668

distanceRef pour SKODA : 2.9833333

distanceRef pour Holden : 2.9344263

distanceRef pour Holden : 2.9344263

distanceRef pour Lexus : 2.7121212

distanceRef pour Volvo : 2.5211267

distanceRef pour Mercedes-Benz : 2.7121212

distanceRef pour Jaguar : 2.8412697

distanceRef pour Lexus : 2.4861112

distanceRef pour Mercedes-Benz : 2.7121212

distanceRef pour Audi : 2.8412697

distanceRef pour SKODA : 2.7121212

distanceRef pour BMW : 2.6323528

distanceRef pour Land Rover : 2.182927

distanceRef pour Jaguar : 2.4189188

distanceRef pour Land Rover : 1.7211539

distanceRef pour Genesis : 1.2235132

distanceRef pour Subaru : 1.6574074

distanceRef pour Holden : 1.6302369

distanceRef pour Holden : 1.6302369

distanceRef pour Jeep : 1.506734

distanceRef pour Holden : 1.3622527

distanceRef pour Jeep : 1.0692952

distanceRef pour Hyundai : 1.4406439

distanceRef pour Kia : 1.4406439

distanceRef pour Nissan : 1.4011742

distanceRef pour Toyota : 1.4206349

distanceRef pour Kia : 1.8080808

distanceRef pour Subaru : 2.3866668

distanceRef pour Nissan : 1.9616438

distanceRef pour Mazda : 1.9888889

distanceRef pour Mazda : 1.9888889

distanceRef pour Kia : 2.9833333

distanceRef pour Suzuki : 9.322916

distanceRef pour Citroen : 6.629629

distanceRef pour Audi : 4.475

distanceRef pour MINI : 4.475

distanceRef pour Volkswagen : 4.475

distanceRef pour MINI : 3.7291667

distanceRef pour MG : 5.303704

distanceRef pour Audi : 5.9666667

distanceRef pour MINI : 5.9666667

distanceRef pour Suzuki : 6.911197

distanceRef pour Suzuki : 6.911197

distanceRef pour Holden : 4.824798

distanceRef pour Suzuki : 9.675675

distanceRef pour Audi : 8.95

distanceRef pour Nissan : 7.7826085

distanceRef pour Volkswagen : 8.95

distanceRef pour SKODA : 7.9555554

distanceRef pour Mazda : 3.7291667

distanceRef pour Hyundai : 4.9722223

distanceRef pour Honda : 5.9666667

distanceRef pour Suzuki : 5.9666667

distanceRef pour Honda : 5.9666667

distanceRef pour Ford : 5.303704

distanceRef pour Mazda : 5.4242425

distanceRef pour Suzuki : 6.911197

distanceRef pour Kia : 5.6825395

distanceRef pour Suzuki : 8.063063

distanceRef pour Kia : 7.9555554

distanceRef pour Mazda : 2.4689655

distanceRef pour Mazda : 3.1964285

distanceRef pour Jaguar : 2.8412697

distanceRef pour Mercedes-Benz : 1.3560606

distanceRef pour Mercedes-Benz : 1.3560606

distanceRef pour Toyota : 1.7047619

distanceRef pour Lexus : 1.5497836

distanceRef pour Genesis : 1.8080808

distanceRef pour BMW : 1.8358974

distanceRef pour BMW : 1.8358974

distanceRef pour Mercedes-Benz : 1.8080808

distanceRef pour Mercedes-Benz : 1.8080808

distanceRef pour BMW : 1.8358974

distanceRef pour BMW : 1.8358974

distanceRef pour Audi : 2.0574713

distanceRef pour BMW : 2.022599

distanceRef pour Porsche : 1.5911111

distanceRef pour Audi : 1.7047619

distanceRef pour BMW : 1.9888889

distanceRef pour Land Rover : 1.325926

distanceRef pour Audi : 2.1284184

distanceRef pour Alfa Romeo : 1.9288793

distanceRef pour Alfa Romeo : 2.1284184

distanceRef pour Porsche : 1.645977

distanceRef pour Lexus : 2.864

distanceRef pour Toyota : 2.864

distanceRef pour Lexus : 2.169697

distanceRef pour Holden : 3.2545455

distanceRef pour Genesis : 2.9833333

distanceRef pour Subaru : 2.9833333

distanceRef pour Ford : 2.9833333

distanceRef pour Mercedes-Benz : 3.509804

distanceRef pour SKODA : 3.58

distanceRef pour Volvo : 2.9833333

distanceRef pour Lexus : 2.9833333

distanceRef pour Mercedes-Benz : 2.9833333

distanceRef pour Volkswagen : 2.9833333

distanceRef pour SKODA : 3.2545455

distanceRef pour Volvo : 2.9833333

distanceRef pour Volvo : 2.9833333

distanceRef pour Land Rover : 2.6716418

distanceRef pour Land Rover : 2.6716418

distanceRef pour Porsche : 2.3866668

distanceRef pour BMW : 3.0338984

distanceRef pour Mercedes-Benz : 2.7121212

distanceRef pour BMW : 2.7538462

distanceRef pour BMW : 2.7538462

distanceRef pour Lexus : 2.7121212

distanceRef pour Alfa Romeo : 2.796875

distanceRef pour Mercedes-Benz : 2.7121212

distanceRef pour Alfa Romeo : 3.086207

distanceRef pour Audi : 2.557143

distanceRef pour Audi : 3.3148148

distanceRef pour Volkswagen : 2.7121212

distanceRef pour BMW : 2.9833333

distanceRef pour Volkswagen : 3.6161616

distanceRef pour Citroen : 4.221698

distanceRef pour Peugeot : 4.221698

distanceRef pour Peugeot : 3.9955359

distanceRef pour Subaru : 3.7291667

distanceRef pour Peugeot : 3.6088707

distanceRef pour MG : 4.339394

distanceRef pour Ford : 3.9777777

distanceRef pour SKODA : 5.114286

distanceRef pour Volkswagen : 4.4088674

distanceRef pour SKODA : 5.114286

distanceRef pour Audi : 4.73545

distanceRef pour Mercedes-Benz : 6.4042935

distanceRef pour Mercedes-Benz : 5.295858

distanceRef pour Subaru : 1.6574074

distanceRef pour Jeep : 1.8645834

distanceRef pour Renault : 2.3866668

distanceRef pour Nissan : 2.3866668

distanceRef pour Toyota : 2.3866668

distanceRef pour Subaru : 2.3866668

distanceRef pour Subaru : 2.273016

distanceRef pour Mazda : 2.3096774

distanceRef pour Lexus : 2.557143

distanceRef pour Toyota : 2.6036363

distanceRef pour Kia : 2.4059138

distanceRef pour Hyundai : 2.1309524

distanceRef pour Kia : 2.1309524

distanceRef pour Mitsubishi : 2.3677247

distanceRef pour Subaru : 3.7291667

distanceRef pour Nissan : 2.9833333

distanceRef pour Honda : 3.1403508

distanceRef pour Kia : 2.887097

distanceRef pour Hyundai : 2.887097

distanceRef pour Toyota : 3.2545455

distanceRef pour Hyundai : 2.557143

distanceRef pour Kia : 2.557143

distanceRef pour Audi : 3.086207

distanceRef pour BMW : 2.7538462

distanceRef pour Mitsubishi : 2.8412697

distanceRef pour BMW : 2.7538462

distanceRef pour Honda : 3.7291667

distanceRef pour Hyundai : 3.6088707

distanceRef pour Holden : 4.339394

distanceRef pour Honda : 4.1871343

distanceRef pour Honda : 4.2619047

distanceRef pour Fiat : 8.523809

distanceRef pour Fiat : 8.523809

distanceRef pour Abarth : 7.3061223

distanceRef pour Kia : 7.8681316

distanceRef pour Mitsubishi : 8.523809

distanceRef pour Kia : 10.228572

distanceRef pour Kia : 1.3560606

distanceRef pour Honda : 2.7121212

distanceRef pour Mitsubishi : 1.9888889

distanceRef pour Great Wall : 2.5718389

distanceRef pour Toyota : 1.6574074

distanceRef pour RAM : 0.640888

distanceRef pour Jeep : 1.1981258

distanceRef pour Mazda : 2.9833333

distanceRef pour Mazda : 2.8078432

distanceRef pour Mazda : 3.509804

distanceRef pour Mazda : 3.509804

distanceRef pour Lexus : 3.8085105

distanceRef pour Toyota : 4.625323

distanceRef pour Subaru : 2.3866668

distanceRef pour Ford : 2.9933112

distanceRef pour Honda : 3.8085105

distanceRef pour Hyundai : 3.58

distanceRef pour Mercedes-Benz : 3.509804

distanceRef pour Subaru : 2.9833333

distanceRef pour BMW : 3.58

distanceRef pour Volvo : 3.3148148

distanceRef pour Volkswagen : 3.2545455

distanceRef pour Volkswagen : 3.58

distanceRef pour MINI : 3.509804

distanceRef pour Mercedes-Benz : 3.509804

distanceRef pour Audi : 3.58

distanceRef pour Audi : 3.2545455

distanceRef pour Audi : 3.2545455

distanceRef pour BMW : 3.509804

distanceRef pour Jaguar : 2.6323528

distanceRef pour BMW : 2.9344263

distanceRef pour Mazda : 3.509804

distanceRef pour Renault : 3.9777777

distanceRef pour Toyota : 4.625323

distanceRef pour Toyota : 4.419753

distanceRef pour Lexus : 4.419753

distanceRef pour Alfa Romeo : 3.3148148

distanceRef pour Holden : 4.661458

distanceRef pour Haval : 4.339394

distanceRef pour SsangYong : 5.0780144

distanceRef pour MG : 4.9722223

distanceRef pour BMW : 4.6797385

distanceRef pour BMW : 4.6797385

distanceRef pour MINI : 4.6797385

distanceRef pour Ford : 4.5897436

distanceRef pour BMW : 4.7733335

distanceRef pour BMW : 4.7733335

distanceRef pour SKODA : 4.7733335

distanceRef pour Suzuki : 5.44073

distanceRef pour Suzuki : 5.44073

distanceRef pour Audi : 4.4088674

distanceRef pour Volkswagen : 5.114286

distanceRef pour Audi : 5.114286

distanceRef pour Audi : 5.114286

distanceRef pour Alfa Romeo : 4.2619047

distanceRef pour Mercedes-Benz : 6.4042935

distanceRef pour Mercedes-Benz : 6.4042935

distanceRef pour Mercedes-Benz : 6.4042935

distanceRef pour Renault : 5.006993

distanceRef pour Toyota : 5.966666

distanceRef pour Peugeot : 5.6289306

distanceRef pour MG : 7.4583335

distanceRef pour Volkswagen : 8.95

distanceRef pour Audi : 7.16

distanceRef pour Jeep : 2.2601008

distanceRef pour Mitsubishi : 2.3677247

distanceRef pour Kia : 3.58

distanceRef pour Hyundai : 3.58

distanceRef pour Hyundai : 3.58

distanceRef pour Hyundai : 3.58

distanceRef pour Kia : 3.58

distanceRef pour Subaru : 3.58

distanceRef pour Toyota : 3.58

distanceRef pour Mitsubishi : 2.8412697

distanceRef pour Nissan : 2.7538462

distanceRef pour Subaru : 2.8412697

distanceRef pour Honda : 4.2316785

distanceRef pour Honda : 3.9777777

distanceRef pour Toyota : 4.625323

distanceRef pour Hyundai : 4.475

distanceRef pour Kia : 4.475

distanceRef pour Hyundai : 4.475

distanceRef pour Kia : 4.475

distanceRef pour Hyundai : 4.475

distanceRef pour Suzuki : 4.760638

distanceRef pour Hyundai : 4.9722223

distanceRef pour Honda : 5.0780144

distanceRef pour Mitsubishi : 3.7883599

distanceRef pour Holden : 5.327381

distanceRef pour Alpine : 4.419753

distanceRef pour Rolls-Royce : 0.66149294

distanceRef pour Rolls-Royce : 0.66149294

distanceRef pour Lamborghini : 0.6479638

distanceRef pour Ferrari : 0.5986622

distanceRef pour Ferrari : 0.5986622

distanceRef pour Ferrari : 0.6244549

distanceRef pour Bentley : 0.662963

distanceRef pour Lamborghini : 0.8605769

distanceRef pour Aston Martin : 0.88264304

distanceRef pour Aston Martin : 0.88264304

distanceRef pour Ford : 1.1737705

distanceRef pour Lexus : 1.0848485

distanceRef pour Jaguar : 1.0228572

distanceRef pour Lexus : 0.87317073

distanceRef pour Maserati : 1.0156028

distanceRef pour Maserati : 0.88570017

distanceRef pour Mercedes-Benz : 1.0156028

distanceRef pour Alpina : 1.2327824

distanceRef pour BMW : 1.196524

distanceRef pour BMW : 1.196524

distanceRef pour Porsche : 1.3984375

distanceRef pour McLaren : 1.2430556

distanceRef pour McLaren : 1.2430556

distanceRef pour Mercedes-Benz : 1.11875

distanceRef pour Aston Martin : 1.2260274

distanceRef pour Bentley : 0.99444443

distanceRef pour Aston Martin : 1.1474359

distanceRef pour Ferrari : 1.1768572

distanceRef pour Ferrari : 1.1768572

distanceRef pour Ferrari : 1.0673822

distanceRef pour Ferrari : 1.0673822

distanceRef pour Ferrari : 1.1474359

distanceRef pour Ferrari : 1.0087348

distanceRef pour Porsche : 1.4061273

distanceRef pour McLaren : 1.3084795

distanceRef pour McLaren : 1.3084795

distanceRef pour Nissan : 1.2731153

distanceRef pour McLaren : 1.3084795

distanceRef pour Nissan : 1.3438438

distanceRef pour Lotus : 2.1309524

distanceRef pour Honda : 1.7635468

distanceRef pour Lotus : 1.7047619

distanceRef pour Lexus : 1.5497836

distanceRef pour Lexus : 1.2473868

distanceRef pour Porsche : 1.8645834

distanceRef pour BMW : 1.9888889

distanceRef pour BMW : 2.2948718

distanceRef pour BMW : 2.2948718

distanceRef pour Audi : 2.0574713

distanceRef pour Mercedes-Benz : 1.4916667

distanceRef pour Jaguar : 1.7047619

distanceRef pour Mercedes-Benz : 1.7047619

distanceRef pour BMW : 1.754902

distanceRef pour Audi : 2.1284184

distanceRef pour Audi : 2.6036363

distanceRef pour Porsche : 2.2375

distanceRef pour Subaru : 3.58

distanceRef pour Toyota : 3.58

distanceRef pour MINI : 4.475

distanceRef pour Mazda : 3.9777777

distanceRef pour BMW : 3.4423077

distanceRef pour Audi : 3.2545455

distanceRef pour Audi : 3.2545455

distanceRef pour Porsche : 2.796875

distanceRef pour Jaguar : 2.8412697

distanceRef pour BMW : 3.4423077

distanceRef pour Lexus : 2.7121212

distanceRef pour Audi : 3.3148148

distanceRef pour Mercedes-Benz : 2.9833333

distanceRef pour Lotus : 4.9722223

distanceRef pour Alfa Romeo : 5.2647057

distanceRef pour MINI : 5.9666667

distanceRef pour Mazda : 5.303704

distanceRef pour Toyota : 2.2948718

distanceRef pour Ford : 2.6381724

distanceRef pour Hyundai : 3.58

distanceRef pour Hyundai : 4.475

distanceRef pour Rolls-Royce : 0.5936982

distanceRef pour Rolls-Royce : 0.5936982

distanceRef pour BMW : 0.69541574

distanceRef pour Chrysler : 0.79910713

distanceRef pour Bentley : 0.662963

distanceRef pour Mercedes-Benz : 0.74583334

distanceRef pour Bentley : 0.7019608

distanceRef pour Lexus : 0.45512334

distanceRef pour Nissan : 0.45663264

distanceRef pour Land Rover : 0.68846154

distanceRef pour BMW : 0.98028475

distanceRef pour BMW : 1.0431235

distanceRef pour Lamborghini : 1.0529412

distanceRef pour Bentley : 1.0529412

distanceRef pour Mercedes-Benz : 0.895

distanceRef pour Porsche : 0.99444443

distanceRef pour Mercedes-Benz : 1.11875

distanceRef pour Maserati : 1.1776316

distanceRef pour Lexus : 1.2473868

distanceRef pour Mercedes-Benz : 1.7047619

distanceRef pour Maserati : 1.4916667

distanceRef pour Audi : 1.4039216

distanceRef pour Porsche : 1.5911111

distanceRef pour Mercedes-Benz : 1.325926

distanceRef pour Audi : 1.4552846

distanceRef pour BMW : 1.5299145

distanceRef pour Land Rover : 1.1474359

distanceRef pour Chrysler : 1.4206349

distanceRef pour LDV : 2.3866668

distanceRef pour Volkswagen : 4.6493506

distanceRef pour Renault : 5.3273807

distanceRef pour Volkswagen : 5.4242425

distanceRef pour Toyota : 1.4612244

La voiture avec le moins d'arrêts de ravitaillement nécessaire est : Lexus

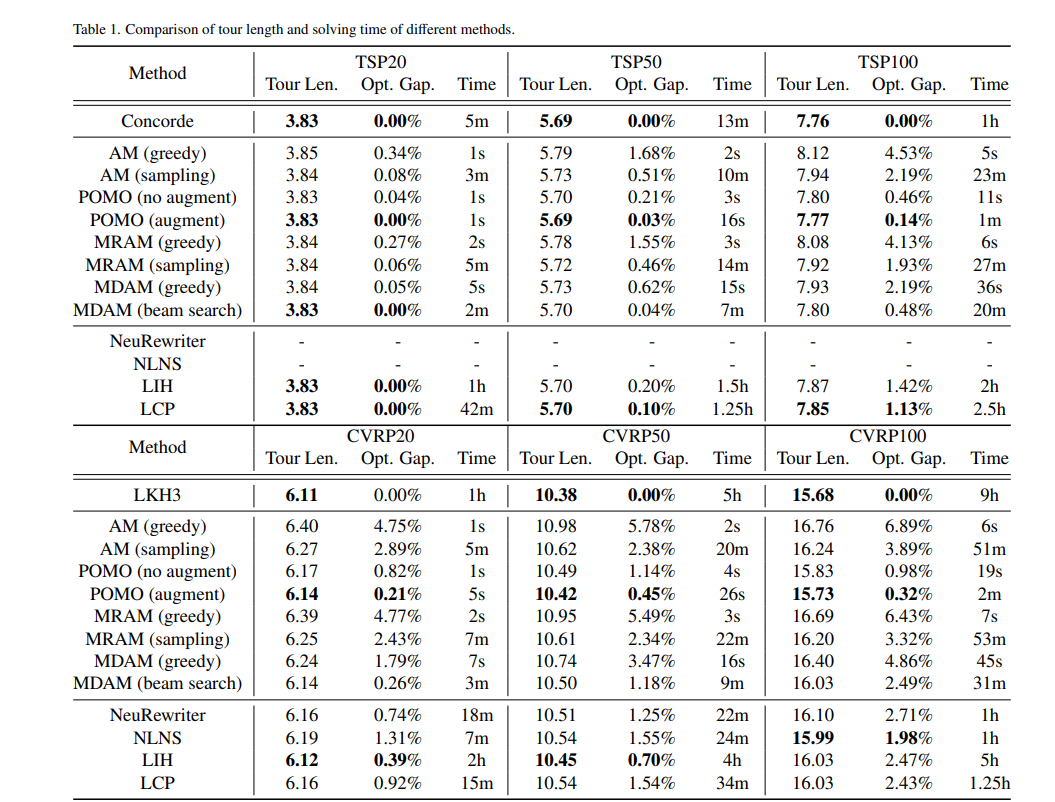
Nombre d'arrêts : 0.45512334

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Approche** | **Tour size** | **Temps d’execusion** | **Distance finale** | **Nombre d’arrets** | **Voiture** |
| Recuit simule | 16 | 17 ms | 358 Km | 0.45512334 | Lexus |

**Tableau 1:** Résultats du système

Ces résultats s'avèrent prometteurs, surtout si l'on considère les contraintes de travail et les estimations parfois imprécises que nous avons utilisées. Ils renforcent la qualité de notre algorithme SA et valident nos méthodes ainsi que nos conceptions du système.

## Comparaison



## En comparant nos résultats avec d'autres articles et projets réalisés par des chercheurs adoptant différentes approches, nous observons l'efficacité de notre algorithme. Il se démarque par sa capacité à générer des itinéraires plus optimaux, réduisant ainsi le temps de trajet au minimum.

## Conclusion

La mise en œuvre d'un système multi-agent avec JADE (Java Agent DEvelopment Framework) et l'intégration de réseaux de neurones convolutionnels (CNN) en Python pour l'extraction des villes et le calcul des distances ont offert une approche novatrice pour résoudre les problèmes de planification d'itinéraires. En utilisant l'algorithme de Recuit simulé, nous avons pu optimiser le trajet pour réduire le temps de parcours et déterminer les véhicules nécessitant le moins d'arrêts de ravitaillement.

L'utilisation de ces technologies a permis de démontrer une amélioration significative dans la génération d'itinéraires plus optimaux, prenant en compte non seulement la distance finale mais également le temps d'exécution. Les résultats obtenus ont validé l'efficacité de notre approche, mettant en évidence la capacité de notre système à optimiser les trajets en minimisant les arrêts nécessaires pour le ravitaillement des véhicules.

En conclusion, l'intégration de différentes technologies telles que JADE pour la gestion des agents, Python avec CNN pour l'extraction des données géographiques, l'application de l'algorithme de Recuit simulé et la sélection de véhicules avec le moins d'arrêts de ravitaillement a abouti à un système robuste et performant pour la planification d'itinéraires efficaces et optimisés. Ces avancées offrent des perspectives prometteuses pour des solutions de planification de trajets plus efficaces dans divers domaines logistiques et de transport.

# **Conclusion général et perspectives**

La conception et l'implémentation d'un système intégrant des technologies telles que JADE pour la gestion des agents, Python avec des réseaux de neurones convolutionnels (CNN) pour l'extraction des données géographiques, ainsi que l'algorithme de Recuit simulé pour la planification des itinéraires, représentent une avancée significative dans le domaine de la planification logistique.

Cette recherche a démontré la capacité de ce système à générer des itinéraires plus optimaux en minimisant le nombre d'arrêts de ravitaillement pour les véhicules, offrant ainsi une solution plus efficace pour la planification de trajets. Les résultats obtenus ont confirmé la pertinence et l'efficacité de notre approche dans l'optimisation des déplacements, prenant en compte à la fois la distance finale et le temps d'exécution.

Ces avancées ouvrent la voie à plusieurs perspectives :

* Améliorations algorithmiques : Explorer et affiner davantage l'algorithme de Recuit simulé pour une optimisation encore plus fine des itinéraires.
* Intégration de données en temps réel : Intégrer des données en temps réel sur le trafic pour permettre une planification plus dynamique et réactive.
* Expansion vers d'autres domaines : Adapter cette solution pour des secteurs tels que la gestion de flotte, la logistique urbaine, ou même les systèmes de transport public.
* Optimisation de la consommation de carburant : Considérer des facteurs supplémentaires comme l'optimisation de la consommation de carburant pour une approche plus respectueuse de l'environnement.

En somme, cette recherche représente un jalon important dans le développement de systèmes intelligents pour la planification d'itinéraires efficaces. Les perspectives futures offrent un terrain fertile pour des améliorations continues et l'application de ces technologies innovantes dans divers secteurs pour des solutions logistiques et de transport plus performantes.