République Algérienne Démocratique et Populaire

Ministère de l'enseignement supérieur et de la recherche scientifique

Université Kasdi Merbah - Ouargla

Faculté des Nouvelles Technologies de l'Information et de la Communication Département d'Informatique et de la Technologie de l'Information

Mémoire

Présenté en vue de l'obtention du diplôme de **Master Professionnel**

Domaine : Mathématiques et informatique.

Filière : Informatique.

Spécialité : Administration et Sécurité des Réseaux.

Thème

L'adaptation d'un protocole IoT dans une architecture IoV

Présenté par :

- *DJOUKLAFI IMED.*
- *BENSLIMANE AZZEDDINE.*

Encadré par :

- *BENMIR ABDELKADER.*
- *AZZAOUI NADJET*

Promotion : 2019/2020.

Remerciements

Je rends

Grâce à dieu le tout puissant pour le courage et la patience qu'il nous 'a accordé pour mener à bien notre mémoire de fin d'études.

Nous adressons nos remerciements les plus vifs et témoigne de notre gratitude à notre encadreur

" Melle Azzaoui Nadjet" pour son encadrement technique, son suivi et ses conseils tout au long de l'année. Nous lui disons un grand merci pour l'aide et conseils qu'il nous a prodiguées.

Nous exprimons également notre gratitude aux membres du Jurys qui nous ont honorées en acceptant de juger ce travail.

Nous tenons à remercier nos très chers parents pour le sacrifice qu'ils ont constitué pendant la durée de Nos études et qui nous fournis au quotidien soutien et une confiance sans faille.

Nous remercions en Particulièrement nos familles et nos amis(es) qui ont su nous soutenir, nous encourager, nous aider et nous supporter tout au long de l'année.

Ensuite nous tiens à remercier l'ensemble des enseignants auxquels nous exprimons toute notre gratitude et notre sympathie et que sans leur collaboration et soutient de ce travail n'aurait pu être réalisé.

Enfin, nous adressons nos plus sincères remerciements

À tous nos collègues

Merci à tous et à toutes

Promotion 2019/2020.

Azzedine, Imed

Résumé

Récemment, les réseaux de véhicules (VANET) ont attiré beaucoup d'attention en raison de la demande croissante de services que ces derniers peuvent fournir sur les routes, qui contribuent à assurer la sécurité et le confort du conducteur et du véhicule et à réduire les accidents de la circulation. Les infrastructures et les voitures en particulier peuvent communiquer entre eux via des mécanismes de communication sans fil tels que p802.11 et LTE, appelés nœuds dans le réseau (VANET), et ces derniers communiquent avec RSU, qui est un élément de la couche Fog qui facilite le transfert et le traitement des données en temps réel, tout cela se fait en intégrant un type de protocole pour assurer un transfert fluide et sécurisé des données. Le mécanisme de transfert assure une bonne communication entre les couches Cloud et Fog. Plusieurs protocoles ont été consacrés pour faciliter ce processus, le protocole RESThttp que nous avons utilisé dans notre recherche. Nous avons utilisé des bibliothèques pré-existantes pour créer un environnement IoV miniature et y implémenter ce protocole. Afin d'en extraire des résultats satisfaisants pour les autres.

Mots clés : "VANET", "DSRC ", " LTE", "RSU", "Fog", "Cloud", " httpREST", " IoV".

ملخص

مؤخرا أصبحت شبكات المركبات (VANET) تحظى بالكثير من الاهتمام نظرا لتزايد الطلب على الخدمات التي يمكن أن تقدمها هذه الأخيرة على الطرقات والتي تساهم في توفير الرفاهية والأمان للسائق والمركبة والحد من حوادث المرور. يمكن للمنشآت والسيارات بصفة خاصة أن تتصل مع بعضها البعض من خلال آليات الاتصال اللاسلكية مثل 802.11 p و LTE و التي تسمى بالعقد في شبكة (VANET) و تتصل هذه الاخيرة مع RSU الذي يعتبر عنصر من طبقة Fog التي تسهل في نقل البيانات و معالجتها في وقت فعلي، كل هذا يتم عن طريق دمج نوع من البروتوكولات لضمان نقل سلس و آمن للبيانات. تعمل آلية النقل على توفير التواصل الجيد بين طبقات Fog و Cloud . سخرت عدة بروتوكولات لتسهيل هذه العملية من بينها RESThttp الذي اسلتعملناه في مجال بحثنا فقد قمنا باستعمال بعض المكتبات الموجودة مسبقا في إنشاء بيئة IoV مصغرة وتنفيذ هذا البروتوكول فيها لنستخلص منها نتائج مرضية تكون مرجع لغيرنا.

الكلمات المفتاحية : "VANET", "DSRC", "LTE", "RSU", "Fog", "Cloud", "httpREST", "IoV".

Abstract

Recently, Vehicle Networks (VANETs) have gained a lot of attention due to the increasing demand for the services they can provide on the roads, which help to ensure the safety and comfort of the driver and the vehicle and reduce the traffic accidents. Infrastructure and cars in particular can communicate with each other via wireless communication mechanisms such as p802.11 and LTE, they are called nodes in VANETs, also they communicate with RSU, which is a part of the Fog layer. This layer facilitates the transfer and processing of data in real time, all this is done by integrating a type of protocol to ensure a smooth and secure transfer of data. The transfer mechanism ensures a good communication between the Cloud and Fog layers. Several protocols have been dedicated to facilitate this process, we used the httpREST protocol in our research, also we used a pre-existing library to create a miniature IoV environment and implement this protocol in it. In order to extract satisfactory results to be a reference for others.

Key words: "VANET", "DSRC ", " LTE", "RSU", "Fog", "Cloud", " httpREST", " IoV".

Table des matières

La liste des Figures

Table des tableaux

Introduction Général

Ces dernières années, l'industrie automobile a connu une évolution et les véhicules ne sont plus considérés comme des systèmes thermomécaniques contrôlés par quelques composantes électroniques. Les véhicules d'aujourd'hui sont des systèmes complexes qui sont contrôlés par des systèmes embarqués ou des réseaux d'ordinateurs. Plusieurs facteurs ont contribué à cette évolution dans le monde de l'industrie automobile telle que l'augmentation des coûts de carburants, la pollution causée par la combustion des carburants et la haute densité automobiles en particulier dans les grandes villes.

Par ailleurs, les systèmes de transport actuels fournissent très peu d'informations sur les conditions routières. Ils sont l'une des causes des difficultés de circulation entraînant des dégâts environnementaux ainsi qu'une piètre qualité de vie. Selon le rapport réalisé par l'OMS1 sur la sécurité routière dans le monde en 2013, les traumatismes dus aux accidents de circulation représentent la huitième cause de décès dans le monde et la première cause de décès de la population dans la tranche d'âge est comprise entre 15 et 29 ans. En effet, ces systèmes de transport sont à la base des embouteillages qui coûtent près de 266 milliards de dollars par an et ralentissent de façon considérable les usagers dans leurs trajets.

Accompagnant l'évolution de l'industrie automobile, les véhicules sont de plus en plus équipés de capteurs, dont le but d'améliorer à la fois la sécurité des passagers et le confort de conduite. Prises individuellement, les informations locales provenant des différents capteurs d'un même véhicule fournissent le plus souvent une connaissance imparfaite, tronquée ou bien incomplète de l'environnement relatif à la route. Cependant, en confrontant les informations issues de plusieurs véhicules, il devient possible d'étoffer et de compléter la perception qu'ont les véhicules de leur environnement. En d'autres termes, en échangeant et en comparant les informations émanant de diverses automobiles, l'information devient plus précise et plus pertinente.

Le but de notre mémoire est de simuler un environnement de IoV dans une architecture spécifique et l'implémentation d'un protocole de communication inter couche.

Le premier chapitre fait l'objet d'une présentation de quelques généralités sur les réseaux VANETs Et accédez au concept de stockage cloud.

Dans le deuxième chapitre on a parlé des différents protocoles de communication existant déjà et la comparaison entre eux en termes de vitesse de transfert du donné et de sécurité et leur propriété avec la présentation des différents architectures d'IoV.

Dans le troisième chapitre, nous décrivons l'environnement de travail, les bibliothèques aideront et choisissez le protocole approprié et son implémentation.

Chapitre

I

Généralité sur Les Réseaux véhiculaires

1. Introduction

L'évolution récente de la technologie dans le domaine de la communication sans fil pousse aujourd'hui les chercheurs à faire des efforts pour mieux assurer la fonction des réseaux, à savoir l'accès rapide à l'information indépendamment du lieu et du temps. Dans le passé, les réseaux sans fil se basaient sur des infrastructures planifiées. Avec la croissance importante qu'ont connue les réseaux sans fil, des besoins en termes d'auto-organisation, d'indépendance et de réduction de couts se sont réalisés. Les réseaux mobiles ad hoc répondent à ces problèmes. Contrairement aux réseaux sans fil basés sur la communication cellulaire, les réseaux ad hoc n'ont aucune administration centralisée, ce sont les hôtes mobiles eux-mêmes qui forment une infrastructure du réseau. Aucune limitation n'est faite sur la taille des réseaux ad hoc, le réseau peut contenir des centaines ou des milliers de nouds mobiles. [3] [5]

Dans MANET, les nœuds sont connectés via des canaux sans fil dans un réseau et chaque nœud agit comme un routeur et comme un hôte. Un des scénarios de MANET est la base des réseaux ad hoc de véhicules. La nouvelle ère de l'Internet des objets (ioT) entraîne l'évolution des réseaux ad hoc de véhicules conventionnels vers l'Internet des véhicules (IoV). Avec le développement rapide des technologies de calcul et de communication, l'IoV promet un intérêt commercial et une valeur de recherche énormes, attirant ainsi un grand nombre d'entreprises et de chercheurs. [5]

Dans ce chapitre, nous présenterons l'un des deux types de réseaux mobiles en fonction de sa structure de réseau (ad-hoc). Après cela, nous listerons une explication sur les réseaux ad hoc véhiculaires et nous énumérerons ses architectures, ainsi que toutes ses applications et les types de technologies de communication qu'il fournit.

2. Les réseaux Ad-hoc :

Un réseau ad hoc est un type de mode de réseau sans fil poste à poste où les appareils sans fil communiquent directement entre eux, sans l'aide d'un appareil WAP (Wireless Access Point). Les réseaux sans fil dépendent généralement d'une station de base ou d'un périphérique WAP pour gérer et diriger le flux de données entre les périphériques sans fil. Dans une configuration ad hoc, le réseau se construit spontanément au fur et à mesure que les appareils communiquent entre eux. Ces appareils devraient idéalement être à proximité les uns des autres ; cependant, la qualité de la connexion et la vitesse du réseau en souffriront à mesure que davantage d'appareils seront ajoutés au réseau. La sécurité d'un réseau ad hoc est inexistante, car les normes de sécurité sans fil telles que WAP2, WAP et WEP ne sont pas autorisées dans un tel réseau direct. [1] [2] [4]

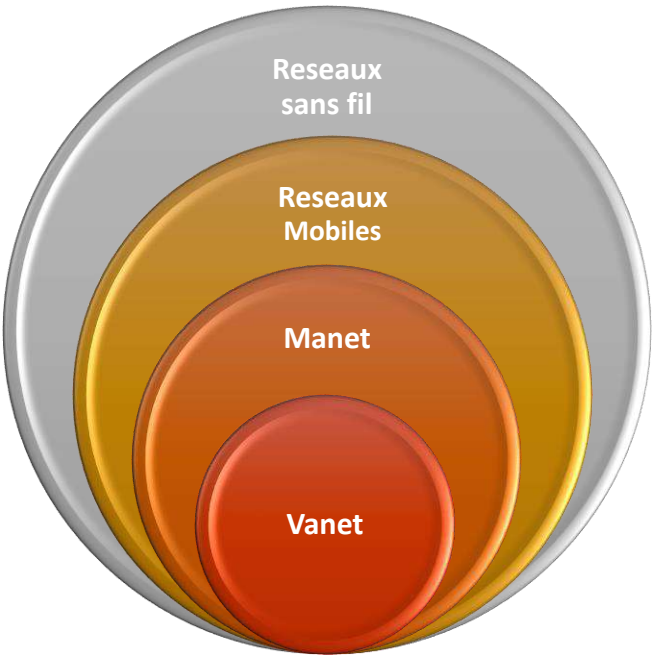

Figure 1 L'évolution des réseaux sans fils

3. Les Types des Réseaux Ad Hoc :

La nature autoportante des réseaux ad hoc les rend très utiles dans des situations telles que les catastrophes naturelles, les opérations militaires d'urgence, ou même pour transférer rapidement des informations entre deux ordinateurs à la maison. [1]

Il existe quatre types, qui sont les suivants :

- **Réseaux mobiles ad hoc (MANET) :**

  Un réseau mobile ad hoc est un réseau auto-formé d'appareils mobiles connectés sans fil.

- **Réseau maillé sans fil (WMN) :**

  Un réseau maillé sans fil est un réseau de communication de nœuds radio structurés dans une topologie maillée. Les clients du réseau sont généralement des ordinateurs portables, des téléphones portables et d'autres équipements sans fil. Le réseau maillé, à l'aide de routeurs et de passerelles, transmet les données vers et depuis les appareils sans fil. La communication est dans le maillage et non vers Internet.

- **Réseau de capteurs sans fil (WSN) :**

  Un réseau de capteurs sans fil utilise des dispositifs basés sur des capteurs pour observer conjointement les paramètres physiques ou environnementaux tels que le son, la pression, les changements climatiques, etc. Les réseaux de capteurs sans fil sont utilisés dans un large éventail de domaines : contrôle du trafic, détection des véhicules, surveillance des serres, etc.

- **Réseaux véhiculaires ad hoc (VANET) :**

  VANET est similaire à MANET en termes, c'est-à-dire qu'il n'a pas besoin d'infrastructure pour la transmission de données. Il peut être défini comme un composant intelligent du système de transport car les véhicules sont capables de communiquer entre eux ainsi que la station de base en bordure de route, qui se trouve aux points critiques de la route.

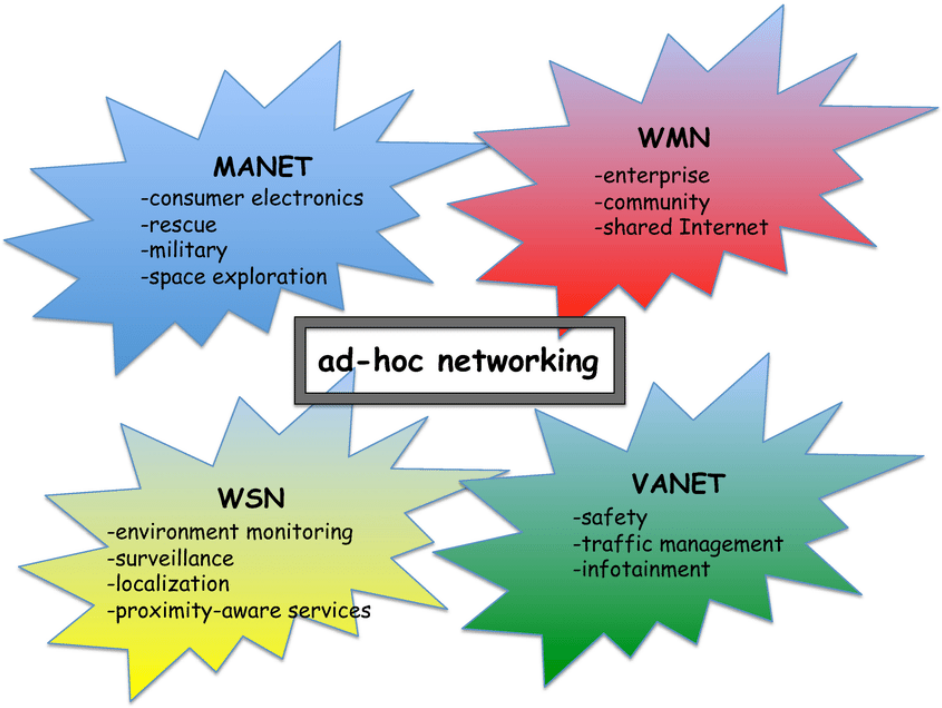

Figure 2 : types of Ad Hoc networks

Puisque notre étude tourne autour des réseaux mobiles, nous discuterons dans les lignes suivantes le concept de (Manet) et (Vanet).

4. Les Réseaux mobile Ad Hoc (Manet) :

Un réseau mobile ad-hoc (MANET) se compose d'hôtes mobiles équipés de dispositifs de communication sans fil. La transmission d'un hôte mobile est reçue par tous les hôtes dans sa portée de transmission en raison de la nature de diffusion de la communication sans fil et des antennes omnidirectionnelles. Si deux hôtes sans fil sont hors de leur portée de transmission dans les réseaux ad hoc, d'autres hôtes mobiles situés entre eux peuvent transférer leurs messages, ce qui crée efficacement des réseaux connectés parmi les hôtes mobiles dans la zone déployée. En raison de la mobilité des hôtes sans fil, chaque hôte doit être équipé de la capacité d'un système autonome ou d'une fonction de routage sans infrastructure statiquement établie ni administration centralisée. Les hôtes mobiles peuvent se déplacer arbitrairement et peuvent être activés ou désactivés sans avertir les autres hôtes. La mobilité et l'autonomie introduisent une topologie dynamique des réseaux non seulement

parce que les hôtes finaux sont transitoires mais aussi parce que les hôtes intermédiaires sur un chemin de communication sont transitoires. [2]

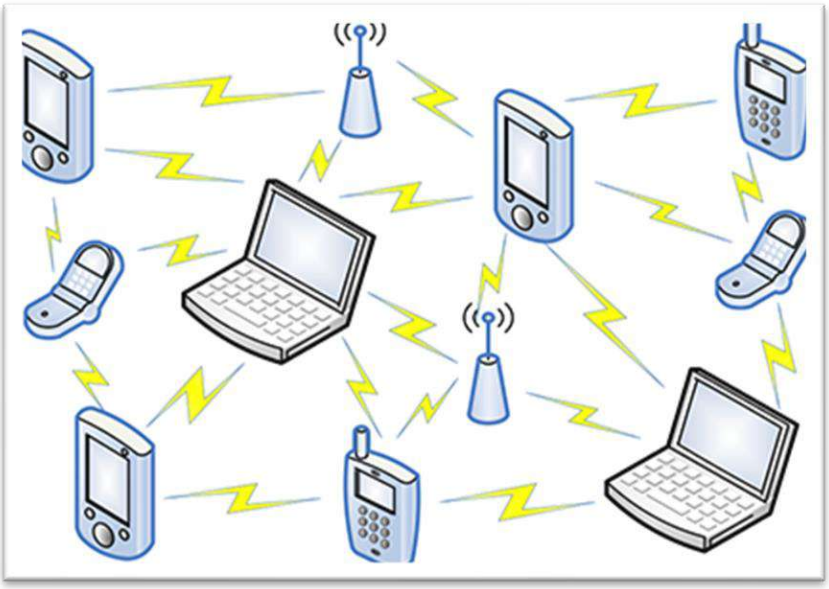

Figure 1 : Exemple de Réseaux Mobile Ad Hoc

5. Les Réseaux véhiculaire Ad Hoc :

Un réseau de véhicule VANET est un type de réseau MANET qui communique sans réseau central et où les nœuds sont équipés de capacités de mise en réseau. Les nœuds mobiles sont des véhicules plutôt intelligents équipés par des ordinateurs, capteurs et cartes réseaux, Comme les autres réseaux ad hoc, les véhicules peuvent communiquer entre eux pour l'échangeur des informations sur le trafic, ou avec d'autres stations occupent le long des routes (pour échanger ou fournir des informations ou se connecter à Internet, etc.). [3] [9]

VANET, d'autre part, est devenu une classe ou une variation difficile et plus responsable de MANET. La liberté des nœuds d'entrer ou de quitter le réseau dans VANET nécessite des protocoles de routage différents de MANET.

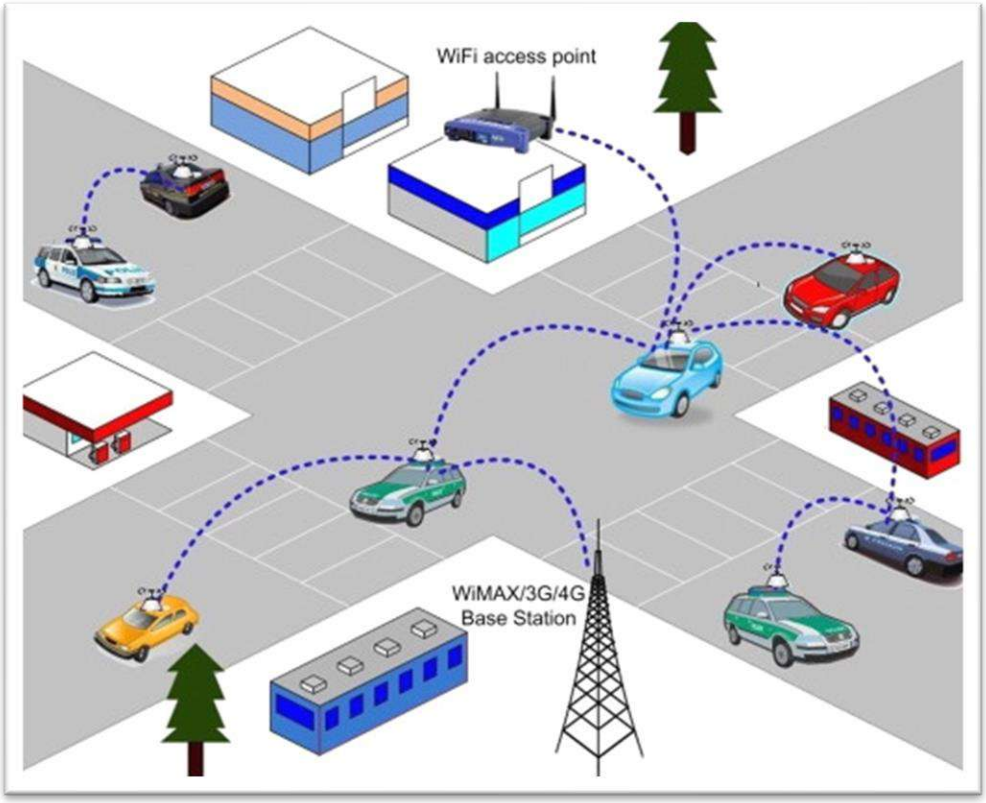

Figure 2 : Infrastructure de Vanet

6. La voiture intelligente :

La voiture intelligente a des fonctionnalités supplémentaires par rapport aux autres voitures, car elle peut se connecter à Internet et à des voitures similaires pour former un réseau qui contient également des capteurs et des ordinateurs, comme le montre la figure suivante

- Collecteur de données (EDR) :

Event Data Recorder est un système matériel de stockage inviolable non volatile utilisé pour enregistrer toutes les informations d'urgence reçues via le VANET, y compris les données de vitesse, l'heure, l'emplacement du véhicule, les données d'accélération, etc.

- Radar avant/arriéré : C'est un capteur qui capture diverses données telles que les distances, les voitures à proximité …
- Interface Homme-machine : C'est une interface de communication entre les humains et les voitures

- Plateforme de traitement : Centre de traitement des données.
- Equipment de communication : L'élément responsable des communications entre les voitures et les poteaux

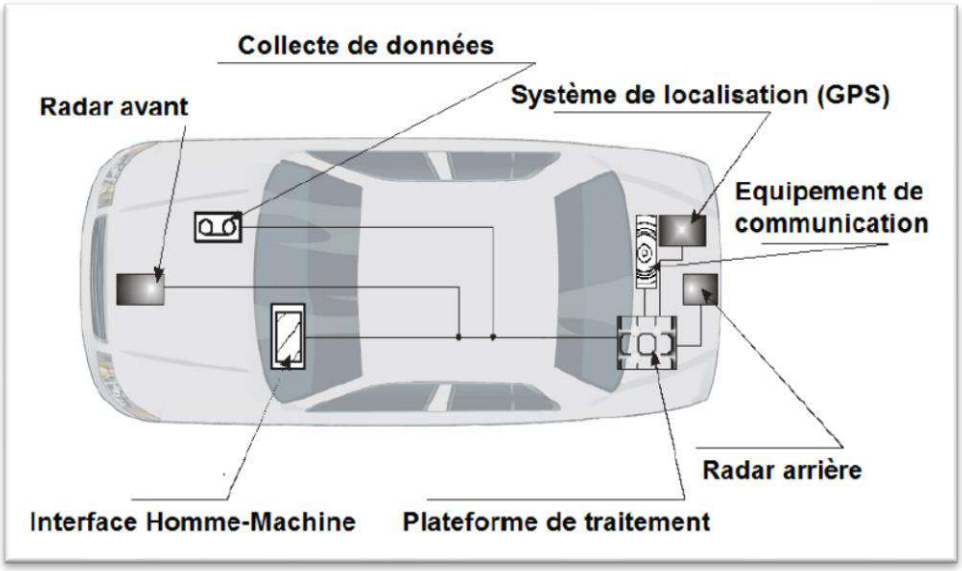

Figure 3 : Un exemple de voiture intelligente

7. Architecture des réseaux véhiculaires

En principe, il n'y a pas d'architecture fixe ou de topologie qu'un VANET doit suivre. Cependant, un VANET général consiste en des véhicules en mouvement communiquant entre eux ainsi qu'avec certains RSU à proximité. Un VANET est différent d'un MANET dans le sens où les véhicules ne se déplacent pas de façon aléatoire comme le font les nœuds dans les MANET, les véhicules en mouvement suivent plutôt des chemins fixes tels que les routes urbaines et les autoroutes. S'il est facile de considérer les VANET comme faisant partie des MANET, il est également important de considérer les VANET comme un domaine de recherche individuel, en particulier lorsqu'il s'agit de concevoir une architecture de réseau. Dans l'architecture VANET, une unité embarquée (OBU) dans un véhicule se compose d'un émetteur et d'un récepteur sans fil. [4] [5] [6]

Au sens large, nous pouvons définir de manière approximative trois scénarios de communication possibles pour les véhicules.

a. La communication de véhicule à véhicule(V2V) :

La première possibilité est que les véhicules communiquent directement entre eux et qu'aucune RSU ne soit nécessaire. Cela peut être classé comme architecture Ad-hoc.

b. La communication de véhicule à infrastructure(V2I) :

Une possibilité est que tous les véhicules communiquent entre eux via une RSU. Cette architecture peut ressembler à des réseaux locaux sans fil (WLAN).

c. La communication hybrides :

Dans la troisième possibilité, certains véhicules peuvent communiquer directement entre eux tandis que d'autres peuvent avoir besoin d'une RSU pour communiquer. Cela peut être appelé un scénario hybride. La figure montre ces trois possibilités.

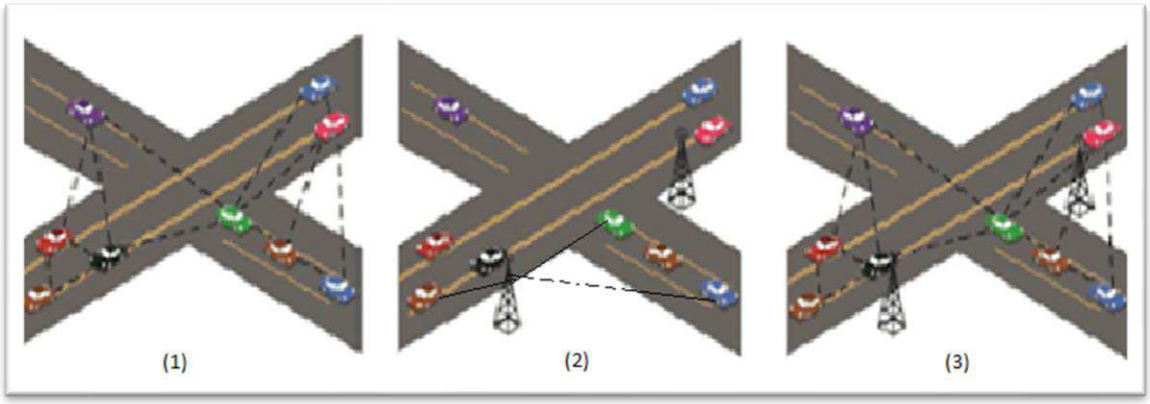

Figure 4 : Vanet Communications Technologies

8. Les applications des Vanets :

Les unités de bord de routes (RSU : Road Side Unit) peuvent être considérées comme des points d'accès ou des routeurs ou même des points tampons qui peuvent stocker des données et les fournir au besoin aux véhicules demandeurs. Toutes les données stockées sur les RSU peuvent être téléchargées par les véhicules. En tant que applications de voiture vers le trafic automobile, les applications vers l'infrastructure, les applications de la voiture vers la maison et les applications au profit du routage.[3][7]

Par ailleurs, une classification des applications pourra être effectuée selon les modes de communication soit V2V ou V2I. Nous organisons les applications liées aux réseaux VANETs dans les classes suivantes :

- Orientée vers la sécurité : Traffic en temps réel, transfert de messages coopératifs, notification post-crash …

- Orientée commercialement : Personnalisation/diagnostic de véhicule à distance, accès à Internet, téléchargement de la carte numérique, publicité à valeur ajoutée …
- Orientée vers la commodité : Routes, systèmes de collecte de péage électronique, disponibilité du stationnement, prévision active.
- Applications productives : Avantages environnementaux, le temps d'utilisation.

9. L'Internet des véhicules

L'Internet des véhicules (IoV) est un réseau distribué qui prend en charge l'utilisation des données créées par les voitures connectées et les réseaux ad hoc de véhicules (VANET). Un objectif important de l'IoV est de permettre aux véhicules de communiquer en temps réel avec leurs conducteurs humains, les piétons, les autres véhicules, les infrastructures routières et les systèmes de gestion de flotte. [10]

10. Les types de communication IoV

L'IoV prend en charge cinq types de communication réseau :

- Systèmes intra-véhicule qui surveillent les performances internes du véhicule via des unités embarquées (OBU).
- Systèmes de véhicule à véhicule (V2V) qui prennent en charge l'échange sans fil d'informations sur la vitesse et la position des véhicules environnants.
- Systèmes de véhicule à infrastructure (V2I) qui prennent en charge l'échange sans fil d'informations entre un véhicule et les unités routières (RSU).
- Systèmes Véhicule to Cloud (V2C) qui permettent au véhicule d'accéder à des informations supplémentaires à partir d'Internet via des interfaces de programme d'application (API).

- Systèmes de véhicule à piéton (V2P) qui prennent en charge la sensibilisation des utilisateurs vulnérables de la route (VRU) tels que les piétons et les cyclistes.

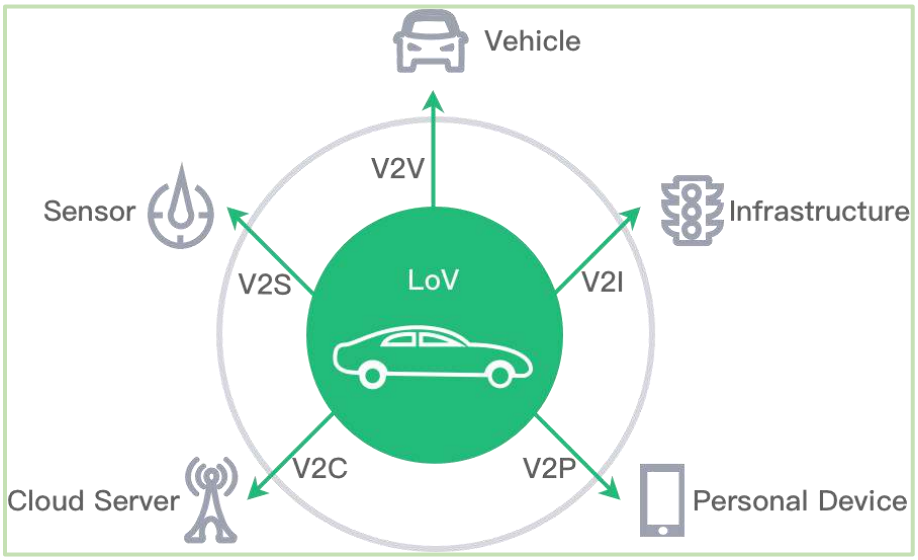

Figure 5 : Types de communication IoV

Selon un récent rapport d'**Allied Market Research**, le marché mondial de l'IoV devrait dépasser les 200 milliards de dollars d'ici 2024 et plusieurs constructeurs automobiles, dont BMW et Daimler, ont annoncé des programmes pour développer une plate-forme qui connectera les services IoV comme la gestion des itinéraires et la smart parking avec centres d'info divertissement à bord.

Les fournisseurs de technologies de l'information (TI) qui travaillent actuellement avec des fabricants et des organisations gouvernantes pour aider à construire l'Internet des véhicules sont Apple, Cisco, Google, IBM, Intel, Microsoft et SAP.

11. Le calcul en ligne (Cloud&Fog computing) :

Le calcul en ligne véhiculaire étend le paradigme du calcul en ligne aux réseaux véhiculaires conventionnels. Cela nous permet de prendre en charge des véhicules plus omniprésents, d'atteindre une meilleure efficacité de communication et de remédier aux limitations des réseaux de véhicules conventionnels en termes de latence, de connaissance de l'emplacement et de réponse en temps réel (généralement requis pour le contrôle intelligent du trafic, les applications de sécurité de conduite, les services de divertissement, et autres applications). De telles exigences sont particulièrement importantes dans les environnements contradictoires (par exemple, la guerre urbaine et les champs de bataille dans l'Internet des objets de champ de bataille impliquant des véhicules militaires). Cependant, il n'y a pas de définition largement acceptée pour le calcul en ligne véhiculaire et les cas d'utilisation. [8]

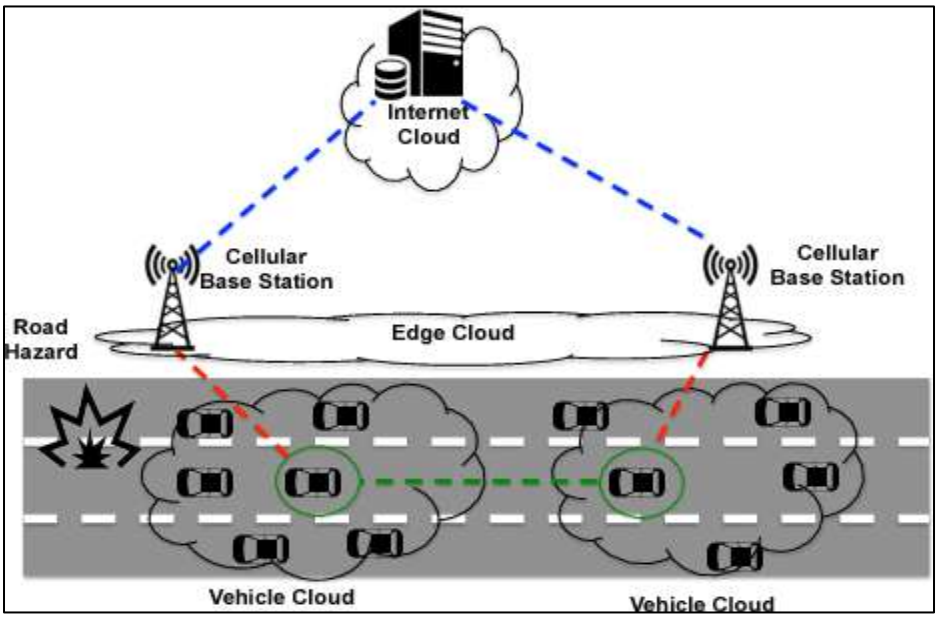

Figure 6 : Le calcul en ligne

12. LTE & DSRC

Le LTE : (Long Term Evolution) est l'évolution la plus récente des normes de téléphonie mobile : GSM/EDGE, UMTS, CDMA2000 et TD-SCDMA.

Le LTE est aussi appelé « 3,9G » car proche de la 4G, mais il ne satisfait pas encore toutes les spécifications imposées par l'Union internationale des télécommunications (UIT) et le consortium 3GPP pour les normes 4G, notamment en termes de bande passante et de débits utilisables

Le LTE sera réellement une norme de 4e génération dans sa version 10 (3GPP release 10) appelée LTE Advanced. [35]

DSRC : Sont des canaux de communication sans fil unidirectionnels ou bidirectionnels de courte portée à moyenne portée spécialement conçus pour l'utilisation automobile, et un ensemble correspondant de protocoles.

Les avantages du DSRC dans trois domaines principaux :

Faible latence - la connexion s'ouvre et se ferme en environ 0,02 seconde.

Interférence limitée - avec ses limites à courte portée ou autour de 1000 m.

Résilience aux intempéries - soleil, nuages ou précipitations, le DSRC est fiable dans toutes les conditions.

13. Conclusion

Ce chapitre a été axé sur le concept des environnements mobiles et l'utilisation de la technologie de communication sans fil. L'évolution rapide qu'a connu la technologie sans fil récemment, a permet l'apparition de nouveaux systèmes de communication qui offrent plus d'avantages par rapport aux systèmes classiques. Les nouveaux systèmes n'astreignent plus l'usager à une localisation fixe, mais lui permet une libre mobilité.

IOV offre de nombreuses fonctionnalités des réseaux véhiculaire ad hoc (VANET) avec des services supplémentaires qui incluent la communication inter-véhicules pour échanger des informations entre les véhicules, les unités routières et les piétons à proximité dans la plage définie. En d'autres termes, il s'agit d'une classe avancée d'applications de système de transport intelligent. [5]

Chapitre

II

Etat de l'art sur les protocoles et les architectures de communication

1. Introduction

La nouvelle ère de l'Internet des objets est à l'origine de l'évolution des réseaux ad hoc de véhicules conventionnels vers l'Internet des véhicules (IoV). Avec le développement rapide des technologies de calcul et de communication, l'IoV promet un intérêt commercial et une valeur de recherche énormes, ce qui nécessite une connexion sécurisée entre les véhicules sur un architecture robuste.

Dans ce chapitre, nous parlerons des différentes architectures de l'IoV qui sont expliquées par certaines recherches open-source. Chaque architecture se compose d'un certain nombre de couches, ainsi que nous aborderons la technologie des protocoles précédemment découverts pour être utilisés pour établir une connexion entre les nœuds de ce réseau véhiculaire.

2. Les architectures d'IOV :

Actuellement, il n'y a pas d'architecture spécifique de l'IoV en raison de la compréhension différente existante de l'IoV. Certains chercheurs proposent l'architecture de l'IoV basée sur l'IoT, qui est similaire à l'IoV et se compose d'une couche de détection, d'une couche réseau et d'une couche d'application. Cependant, l'IoV n'est pas seulement un réseau de services pour la communication de véhicule à véhicule ou les terminaux de véhicule, mais aussi un système complexe qui présente la particularité d'une interaction étroitement coordonnée homme-véhicule-environnement et d'une évolution hautement dynamique. [32]

Une architecture à trois couches a été identifiée dans [28], qui décrivent les différentes interactions des technologies de l'environnement IoV. La première couche se compose de tous les capteurs du véhicule qui collecte données environnementales et de détecter certains événements importants tels que les situations des véhicules, les modèles de conduite, et les conditions de l'environnement environnant. La deuxième couche de communication qui prend en charge différents types de communication sans fil comme le véhicule à piéton (V2P), le

véhicule à l'infrastructure (V2I), le véhicule au véhicule (V2V) et le véhicule à capteur (V2S). Grâce à la couche de communication, une connectivité transparente est assurée à plusieurs réseaux tels que IEEE 802.15.4, IEEE 802.11p, GSM, LTE, Wi-Fi, Bluetooth. La troisième couche dispose des ressources de renseignement IOV et responsable de prendre des décisions dans des situations à risque (par exemple, des conditions routières dangereuses et des accidents). Cette couche contient des outils statistiques ainsi que les ressources collectées de stockage et de traitement de données big data.

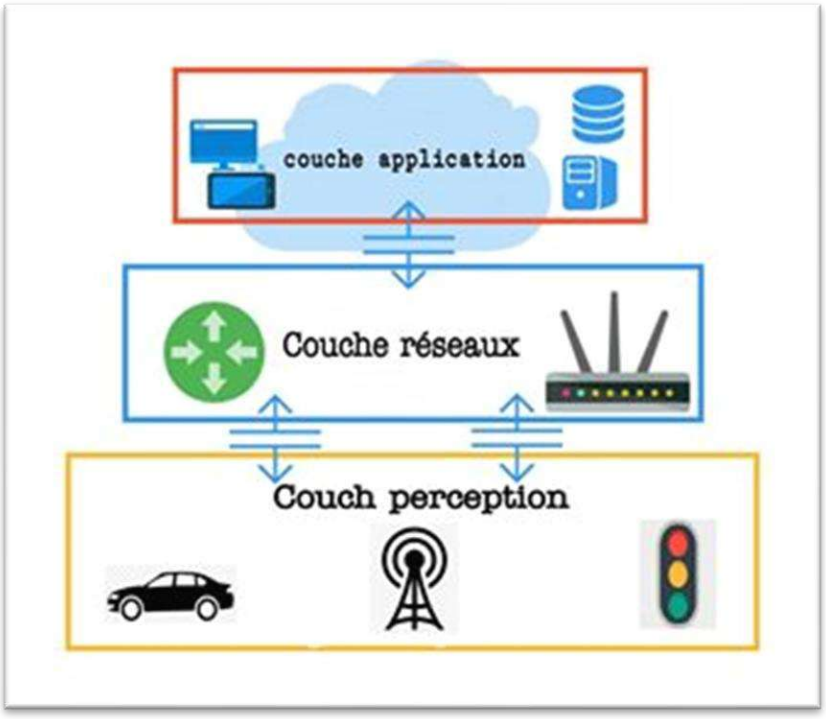

Figure 7 : Architecture iov 3 couche

Dans Bonomi [29], CISCO est proposé une architecture à 4 couches pour IoV. La première couche couvre les véhicules, les logiciels de communication principalement pour V2V et les liaisons de communication utilisant le protocole 802.11p. La deuxième couche, appelée infrastructure, définit les technologies qui permettent la connectivité entre tous les participants de l'écosystème IoV dans lequel le véhicule est situé à tout moment. La couche d'opération est la troisième couche. Il vérifie et assure le respect de toutes les politiques applicables pour réglementer la gestion de l'information et le flux. La quatrième couche qui est la couche cloud indique le type

de nuage (public, privé ou commercial) basé sur un profil défini couplé à la possibilité de recevoir des services (voix, vidéo d'entreprise et données) à la demande.

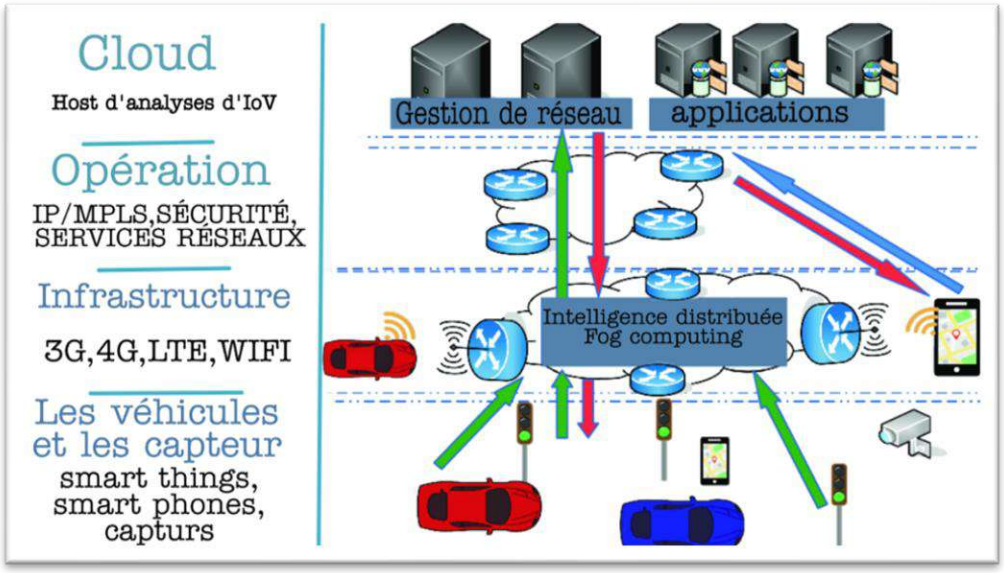

Figure 8 : Architecture iov 4 couche

Dans [31] proposé une architecture à 5 couches, qui est composée des couches suivantes :

- ✓ Perception : Cette couche représente l'interaction entre le véhicule et son environnement. Il utilise des dispositifs à l'intérieur des véhicules tels que des capteurs, des actionneurs, des appareils personnels et ceux installés en travers de la route comme RSU afin de collecter des informations pertinentes à utiliser dans les décisions relatives aux véhicules.
- ✓ Coordination : Cette couche est principalement responsable de l'interopérabilité, du routage et de la sécurité du transport des messages.
- ✓ Intelligence artificielle : Il s'agit de la couche principale où les tâches des composants de décision doivent être exécutées. Cette couche se concentre principalement sur l'analyse de Big Data, l'exploration de données, le cloud computing et la décision basée sur des systèmes experts.
- ✓ Application : cette couche concerne le type de services et d'exigences disponibles dans le système.
- ✓ Business : c'est la partie qui décrit les types d'entreprises que le marché de l'IoV proposera aux utilisateurs.

Une architecture de modèle à sept couches pour IoV est introduite. Les sept couches sont :

Interface utilisateur, Acquisition de données, Filtrage et prétraitement, Communication, Contrôle et gestion, Traitement, Sécurité.

Une interface utilisateur-véhicule est prise en charge par l'architecture à sept couches pour gérer les interactions entre le conducteur et le véhicule. Une interface de communication a également été introduite pour une sélection optimale du réseau de transmission. En analysant les architectures IoV existantes, il est clair qu'il n'y a pas eu de considération pour les exigences en temps réel du processus de big data. En outre, toutes les architectures précédentes supposent que toutes les informations collectées doivent être envoyées aux centres de données (c.-à-d. centres de cloud computing) pour le traitement et l'analyse. Par conséquent, ces architectures ne conviennent pas à de nombreuses applications ITS qui nécessitent une analyse big data en temps réel. En particulier, la latence élevée et la surcharge du réseau de communication sont les résultats attendus du déploiement de l'une des architectures précédentes. [30]

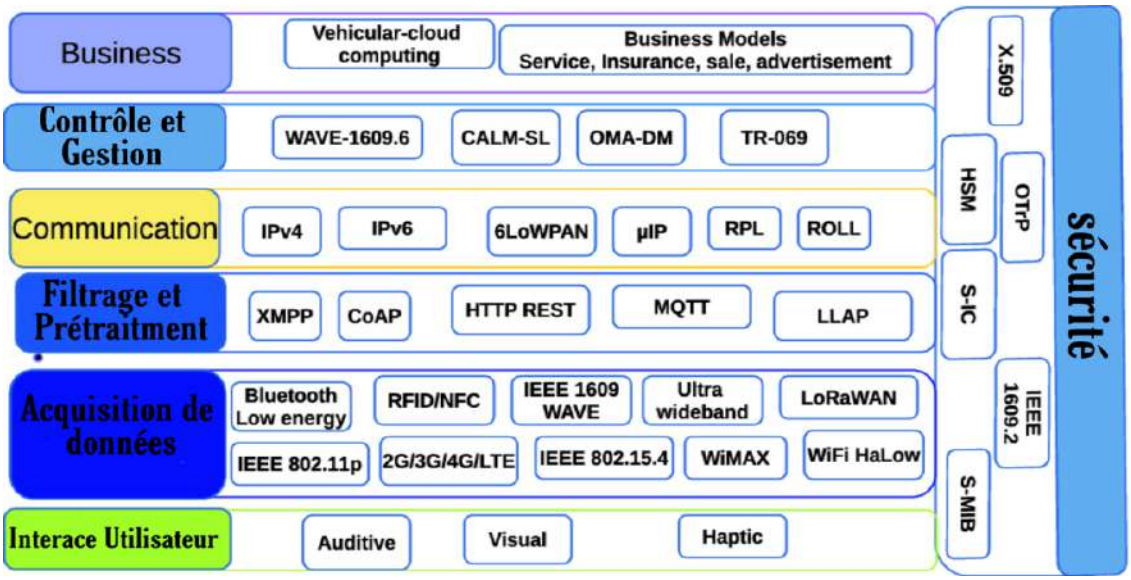

Figure 9 : Architecture iov 7 couche

ARCHITECTURES IOV PROPOSÉES	NOMBRE DE COUCHES	MODÈLES DE COMMUNICATION	SÉCURITÉ
[28]	Trois couches : client, connexion et cloud	V2V, V&R, V&P, V&I	la sécurité en tant que service
[29]	Quatre couches : services, exploitation, infrastructure et points d'extrémité	V2V, V&I	couches croisées
[31]	Cinq couches : perception, coordination, intelligence artificielle, application et entreprise	V&I, V2V, V&S, V&P, V&R	Security plane
[30]	Sept couches : interface utilisateur, acquisition de données, filtrage et prétraitement, communication, contrôle et gestion, traitement, sécurité	V&I, V2V, V&S, V&P, V&R, R&R, R2P, S&A	couches croisées

Tableau 3 : Résumé des caractéristiques des architectures IoV proposées.

3. Les protocoles :

Parmi les nombreux facteurs qui détermineront les performances dans un paradigme combiné IoV, fog et cloud computing, la communication de la couche application, qui dépend elle-même des protocoles de communication sélectionnés, est l'un des principaux. Malgré la popularité et l'utilisation largement répandue de HTTP, les protocoles actuellement utilisés dans divers domaines de l'IoV, du brouillard et des domaines cloud sont de facto fragmentés avec de nombreuses solutions différentes, il n'y a pas d'accord commun sur l'architecture de référence ou les normes adoptées des protocoles de communication. Ainsi, l'un des défis fondamentaux pour les ingénieurs système est de choisir le protocole approprié pour leurs exigences système IoV spécifiques, tout en tirant parti des progrès du brouillard et du cloud computing.

Les problèmes spécifiques des protocoles de communication dans la couche application n'ont pas encore été traités. Cette partie examine les protocoles de communication dans la couche d'application également appelés protocoles de messagerie et machine à machine.

Voici quelques protocoles qui peuvent nous être utiles dans l'étude :

a. MQTT (Message Queue Telemetry Transport) :

MQTT signifie MQ Telemetry Transport. Il s'agit d'un protocole de messagerie de publication / abonnement extrêmement simple et léger, conçu pour les appareils contraints et les réseaux à faible bande passante, à latence élevée ou non fiables. Les principes de conception sont de minimiser la bande passante du réseau et les besoins en ressources des appareils tout en essayant également d'assurer la fiabilité et un certain degré d'assurance de livraison. Ces principes se révèlent également rendre le protocole idéal du monde émergent « machine-to-machine » (M2M) ou « Internet des objets » des appareils connectés, et pour les applications mobiles où la bande passante et la puissance de la batterie sont primordiales. [23] [24]

Du point de vue IoV, les publieurs sont essentiellement les capteurs légers qui se connectent au courtier pour envoyer leurs données et se rendormir chaque fois que possible.

Les abonnés sont des applications qui s'intéressent à un certain sujet ou à des données sensorielles, ils se connectent donc aux courtiers pour être informés chaque fois que de nouvelles données sont reçues.

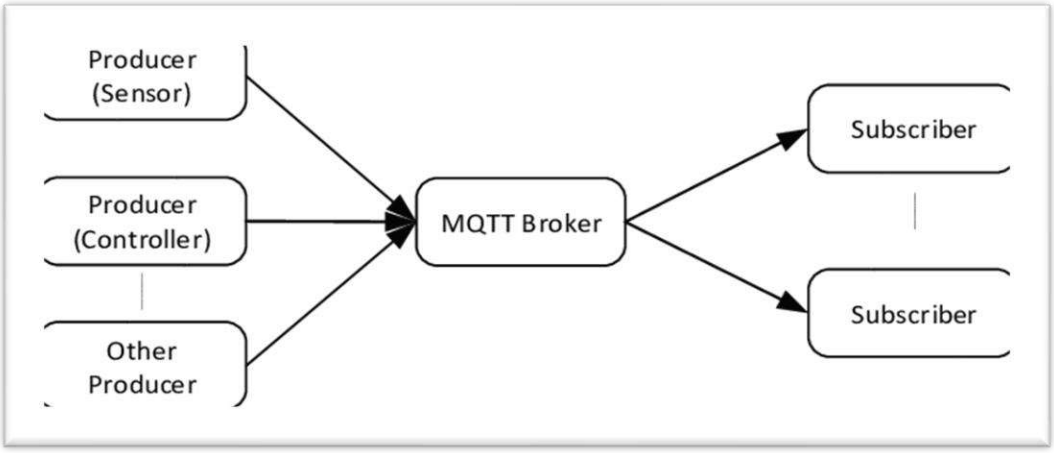

Figure 10 : MQTT Publisher/Subscriber Architecture

b. CoAP (Constrained Application Protocol) :

Il s'agit d'un protocole de couche d'application de request / response synchrone qui a été conçu par l'Internet Engineering Task Force (IETF) pour cibler les dispositifs à recours contraint. Il a été conçu en utilisant un sous-ensemble des méthodes HTTP le rendant interopérable avec HTTP ; il utilise le protocole UDP pour une implémentation légère.

Il utilise également l'architecture RESTful, qui est très similaire au protocole HTTP. Il est utilisé dans les applications mobiles et basées sur les réseaux sociaux et élimine toute ambiguïté en utilisant les méthodes HTTP get, post, put et delete. Outre la communication des données IoT et IoV, CoAP a été développé avec DTLS pour l'échange sécurisé de messages. Il utilise DTLS pour le transfert sécurisé des données dans la couche transport. [19] [20] [21]

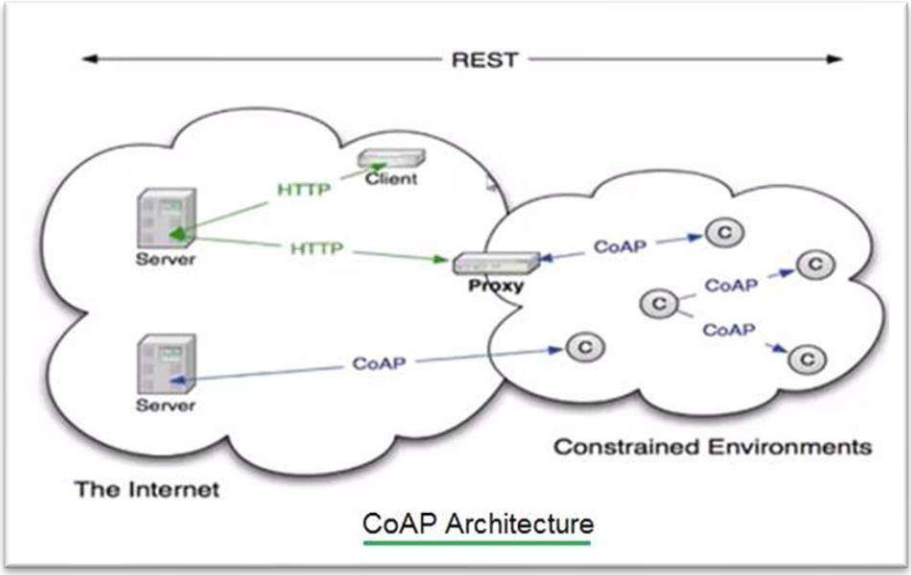

Figure 11 : COAP Architecture

c. AMQP (Message avancé Protocole de mise en file d'attente) :

Le protocole AMQP (Advanced Message Queuing Protocol) est un autre protocole de couche session conçu pour l'industrie financière. Il fonctionne sur TCP et fournit une architecture de Publisher / Subscriber qui est similaire à celle de MQTT. La différence est que le courtier est divisé en deux composants principaux : l'échange et les files d'attente. [22]

Son principal avantage est sa fonction de stockage et de retransmission qui garantit la fiabilité même après des perturbations du réseau [21]. Il garantit la fiabilité avec les garanties de remise de message suivantes :

1. Au plus une fois : signifie qu'un message est envoyé une fois, qu'il soit remis ou non.
2. Au moins une fois : signifie qu'un message sera définitivement délivré une fois, éventuellement plus.
3. Exactement une fois : signifie qu'un message ne sera délivré qu'une seule fois.

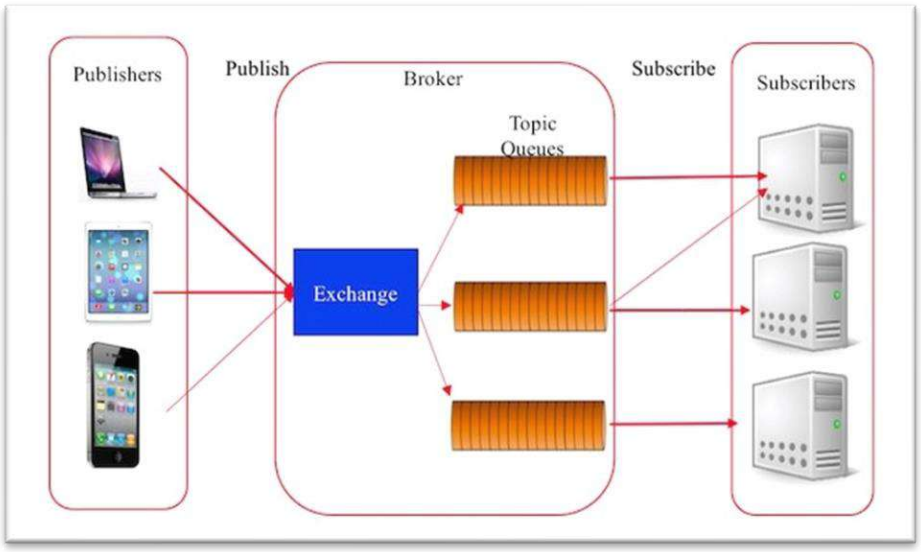

Figure 12 : AMQP Architecture

d. XMPP (Protocole de messagerie et de présence extensible) :

Le protocole de messagerie et de présence extensible (XMPP) est un protocole qui a été initialement conçu pour les applications d'échange de messages et de chats. Il est basé sur le langage XML et a été normalisé par l'IETF il y a plus d'une décennie. Récemment, son utilisation a été étendue pour les applications IoT et SDN en raison de l'utilisation standardisée de XML qui le rend facilement extensible. XMPP prend en charge l'architecture de publication / abonnement et de demande / réponse. Il est conçu pour les applications en temps quasi réel et prend donc en charge efficacement les petits messages à faible latence. [22] [19]

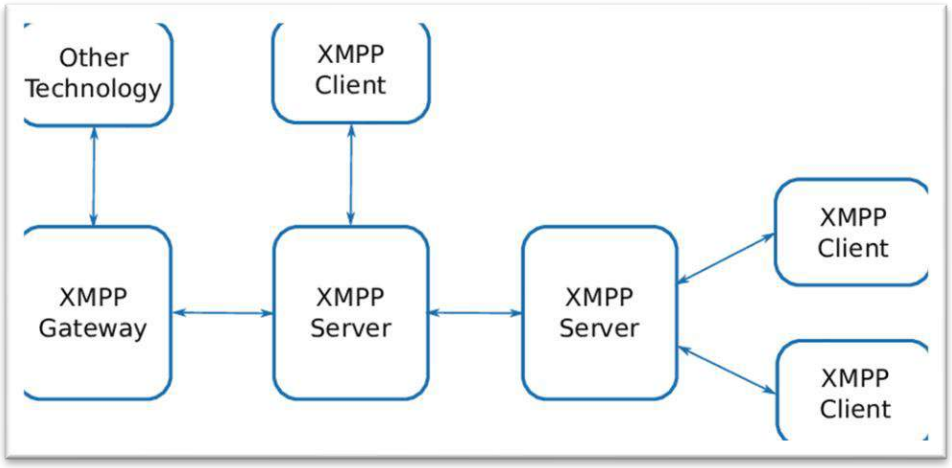

Figure 13 : Architecture de base de XMPP

e. Http (HyperText Transfer Protocol)

Le protocole HTTP (HyperText Transfer Protocol) est le protocole le plus utilisé sur internet, son architecture « request/response » le but du protocole HTTP est de permettre un transfert de fichiers (essentiellement au format HTML) localisés grâce à une chaîne de caractères appelée URL entre un navigateur (le client) et un serveur Web.

Un client et un serveur se produisent via une messagerie de demande / réponse, le client envoyant un message de demande HTTP et le serveur renvoyant ensuite un message de réponse, contenant la ressource qui a été demandée au cas où la demande serait acceptée.

Récemment, HTTP a été associé à REST que nous expliquerons ensuite, une directive pour développer des services Web basés sur un style architectural spécifique afin de définir l'interaction entre les différents composants. En raison du succès des services Web RESTful, il y a eu beaucoup d'efforts pour intégrer cette architecture dans les systèmes basés sur l'IoV en combinant HTTP et REST.

La combinaison du protocole HTTP avec REST est louable, car les appareils peuvent rendre leurs informations d'état facilement disponibles, grâce à un moyen normalisé de créer, lire, mettre à jour et supprimer des données (les opérations dites CRUD). [25] [22]

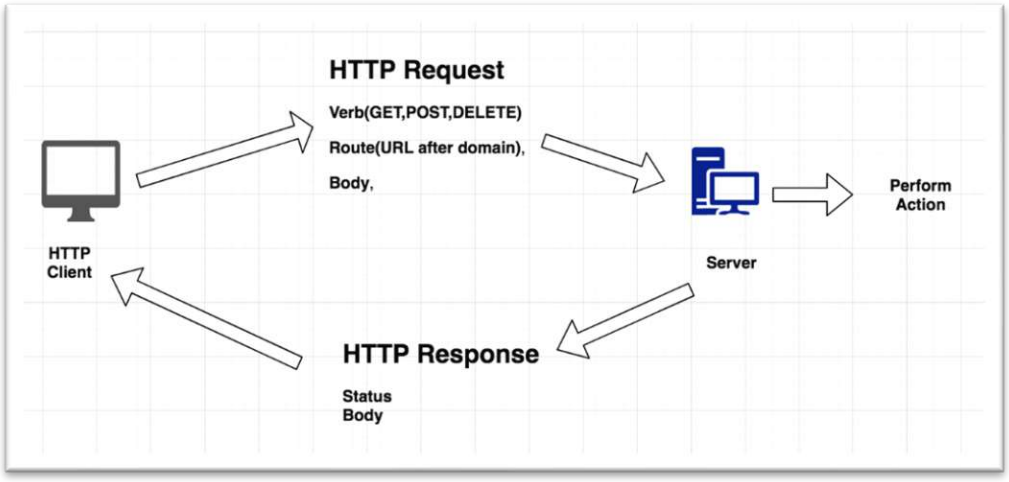

Figure 14 : HTTP Request/Response Architecture

4. RESTfull Services

Le Representational State Transfer (REST) n'est pas vraiment un protocole mais un style architectural. Il a été introduit pour la première fois par Roy Fielding en 2000, et il est largement utilisé depuis. REST utilise les méthodes HTTP GET, POST, PUT et DELETE pour fournir un système de messagerie orienté ressources où toutes les actions peuvent être effectuées simplement en utilisant les commandes synchrones de HTTP (requête / réponse). Il utilise l'en-tête d'acceptation intégré de HTTP pour indiquer le format des données qu'il contient. Les fonctionnalités de HTTP peuvent être entièrement utilisées dans l'architecture REST, y compris l'encaissement, l'authentification et la négociation du type de contenu. Les services RESTful utilisent HTTP sécurisé et fiable qui est le langage Internet éprouvé dans le monde entier. Il peut utiliser TLS / SSL pour la sécurité. [19]

Les Types des méthodes :

Les demandes HTTP RESTful sont classées selon les types de méthode comme suit **:**

- ✓ POST
- ✓ GET
- ✓ PUT
- ✓ DELETE

POST méthode : La méthode RESTful HTTP Request POST est équivalente aux fonctions Create et INSERT SQL. L'exemple suivant consiste à insérer un nouveau partenaire, Partner 1, dans la base de données.

HTTP Message Type	Example
Request	`http://localhost:8090/tpmRest/v1/participants/participant?isHost=false&name=partner1&isActive=true`
Response	• Successful operation response:`{"result":"Operate successfully"}` • Failed operation response:`{"errorMessage":"XXXXXX"}`

Figure 17 : exemple méthode post

GET méthode : La méthode RESTful HTTP Request GET est équivalente aux fonctions Retriever et Select SQL. Cet exemple suivant consiste à récupérer les valeurs du protocole EZComm activé de l'acheteur partenaire.

HTTP Message Type	Example
Request	`http://localhost:8090/tpmRest/v1/participants/transports/all?participantName=partner1&protocolName=EZComm`
Response	`{"result":[{"name":"file","type":"FILE"},{"name":"http","type":"HTTP"}]}`

Figure 18 : exemple méthode get

PUT Méthode : La méthode RESTful HTTP Request PUT est équivalente aux fonctions Update et à l'instruction UPDATE SQL. Voici un exemple pour mettre à jour les informations d'identification de certificat d'un participant dans une base de données.

HTTP Message Type	Example
Request	`http://localhost:8090/rest/participants/certificatecredential?participantName=test &newAlias=new_cert&fileContent=MIIOQQIBAzC+==&fileName=bcpartner1_cert.p7b &shadowCertName=shadow_cert&getby=1389196800000&alias=cert` **Note:** File content data object: *MIIOQQIBAzC+==* is in Base64 format.
Response	• Successful operation response: `{"result":"Operate successfully"}` • Failed operation response: `{"errorMessage":"XXXXXX"}`

Figure 19 : exemple méthode put

DELETE méthode:

La méthode RESTful HTTP Request DELETE est équivalente aux fonctions Delete et à l'instruction SQL DELETE.

Voici un exemple pour supprimer un partenaire dans la base de données.

HTTP Message Type	Example
Request	`http://localhost:8090/tpmRest/v1/participants/participant?name=new_participant`
Response	• Successful operation response: `{"result":"Operate successfully"}` • Failed operation response: `{"errorMessage":"XXXXXX"}`

Figure 20 : exemple méthode delete

Ce paragraphe décrit les protocoles mentionnés ci-dessus en fonction de leurs principales caractéristiques. Le tableau 2 résume l'architecture, protocole de transport et mécanismes de sécurité. MQTT, AMQP, XMPP et REST HTTP, sont conçus pour fonctionner sur des réseaux qui utilisent TCP, tandis que CoAP utilise UDP comme transport sous-jacent. Comme mentionné dans le section précédente, MQTT, AMQP implémentent un modèle de publication / abonnement, tandis que REST http et CoAP implémente un modèle d'interaction demande / réponse. Les protocoles MQTT, AMQP et CoAP fournissent prise en charge QoS très basique pour la livraison des messages. MQTT et AMQP implémentent trois QoS différentes niveaux, tandis que dans CoAP, les messages de demande et de réponse sont limités à deux. La QoS dans REST HTTP et XMPP est fournie par les protocoles de transport sous-jacents. La plupart de ces protocoles choisissent le protocole TLS ou DTLS comme mécanismes de sécurité.

PROTOCOLE	TRANSPORT	ARCHITECTURE	SÉCURITÉ
REST HTTP	TCP/IP	Req/Rep	TLS/SSL
COAP	UDP/IP	Pub/Sub - Req/Rep	DTLS
MQTT	TCP/IP	Pub/Sub	TLS/SSL
AMQP	TCP/IP	Pub/Sub - Req/Rep	TLS/SSL
XMPP	TCP/IP	Pub/Sub - Req/Rep	TLS/SSL

Tableau 2 : Tableau 4 Comparaison entre les 5 protocole

5. Conclusion

Arguments pour le protocole httpREST

- ✓ **Transport :** HTTP utilise TCP. Tout en utilisant TCP fournit fiable la livraison de grandes quantités de données, ce qui est un avantage dans les connexions qui n'ont pas de exigences de latence, il crée des défis dans les environnements à ressources limitées. Un dès le principal problème est que les nœuds contraints envoient la plupart du temps de petites quantités de données sporadiquement et la mise en place d'une connexion TCP prend du temps et produit une surcharge inutile. Pour QoS, HTTP ne fournit pas d'options supplémentaires, mais s'appuie plutôt sur TCP, ce qui garantit livraison réussie tant que la connexion n'est pas interrompue. [26]
- ✓ **Sécurité :** HTTP comme mécanisme de sécurité utilise le TLS très connu pour activer le cryptage sécurisé canal de communication, résultant en une version sécurisée de HTTP, également connu sous le nom de HTTPS. La première une partie de la sécurisation de l'échange de données client /serveur est une prise de contact TLS, implémentée comme échange des messages « bonjour client » et « bonjour serveur » où ils doivent se mettre d'accord sur une suite de chiffrement, qui est une combinaison d'algorithmes qu'ils utiliseront pour garantir des paramètres sécurisés. Après cela, le client et clés d'échange côté serveur basées sur l'algorithme d'échange de clés convenu. Le résultat est un échange des messages chiffrés avec une clé secrète partagée. Les données sont cryptées pour empêcher quiconque d'écouter et comprendre le contenu. [27]

Chapitre

III

Implémentation – Etude de cas

1. Introduction

Afin d'évaluer les performances du protocole spécifique à l'étude, nous devons l'adapter à l'environnement du simulateur, dans ce chapitre, nous présenterons principalement les outils de mise en œuvre des simulations, puis les différentes étapes d'implémentation du protocole à l'aide de l'émulateur OMNET ++.

2. L'environnements de travail :

2.1.Simulateur (OMNET++) :

OMNeT ++ (Objective Modular Network Testbed en C ++) est une bibliothèque et un simulateur d'évènements basé sur le langage C++, principalement pour la construction de simulateurs de réseau et les systèmes distribués. Il est totalement programmable, paramétrable et modulaire. C'est une application open-source et sous licence GNU. OMNeT ++ peut être utilisé gratuitement pour des simulations non commerciales comme dans les institutions académiques et pour l'enseignement. Destiné principalement à la recherche et (Opnet) et (OMNeST) sont des alternatives commerciale d'OMNET++. [12] [13]

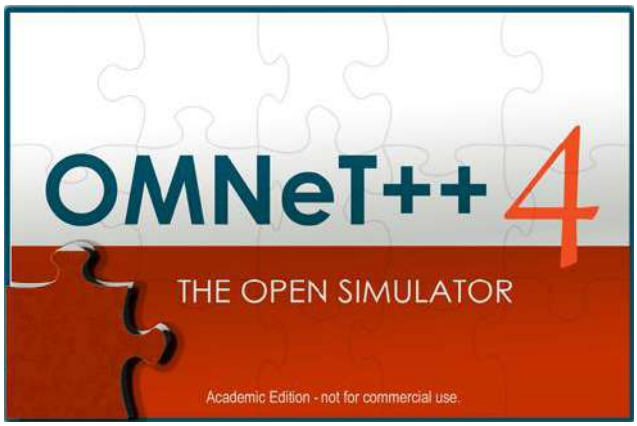

Figure 21 : La version 4 d'Omnet++

Dans la figure ci-dessous, les cases représentent des modules simples (fond gris) et des modules composés. Les flèches reliant les petites boîtes représentent les connexions et les portes.

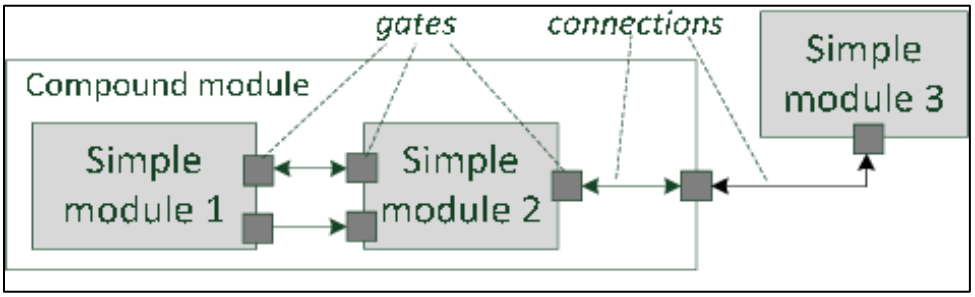

Figure 22 : Modules simples et composés

2.2.Pourquoi OMENT++

Actuellement, il y a de nombreuses plateformes de simulation de réseaux disponibles sur le marché, qu'elles soient commerciales ou non. Nous ferons ici une comparaison entre OMNet ++ et les plus importantes solutions concurrentes.

Dans la topologie de réseau et les modèles hiérarchiques, OMNET ++ est meilleur que NS2 et OPNET, NS2 est insuffisant dans ce type de capacité et OPNET a trop de limites. Comme WSN n'avait pas la reconnaissance d'une topologie d'analyse supérieure et du protocole de réseau, généralement les environnements différents nécessitent une topologie et un protocole différents, de sorte que les inconvénients de l'inflexible ont une sérieuse influence sur la simulation dans WSN. [14]

OMNET ++ est le seul qui prend en charge les deux modèles de programmation.

Dans la bibliothèque de simulation, la bibliothèque de simulation NS2 fournit moins de fonctions. La bibliothèque de simulation OPNET est basée sur C, tandis que celle d'OMNET ++ est une bibliothèque de classes C ++.

Dans cet aspect, les fonctions d'OMNET ++ sont beaucoup plus puissantes que NS2, C ++ est meilleur que C dans la simulation WSN. [14]

L'ouverture du code source est non seulement nécessaire pour incorporer ou modifier le moteur de simulation, mais fournit également une aide importante dans le débogage des modèles de simulation. OMNET ++ et NS2 ne sont pas payés pour la licence,

et les utilisateurs peuvent apprendre ces architectures logiques, donc OMNET ++ et NS2 sont meilleurs que OPNET dans cet aspect.

Logiciel de simulation dans WSN, les performances de NS2 ne sont pas très bonnes, OPNET a de bonnes performances (comme OMNET ++), mais c'est trop cher. Donc, OMNET ++ est le meilleur que les autres simulateurs dans WSN simulation. [14]

2.3.Inet :

Inet Framework est une bibliothèque de modèles open source pour l'environnement de simulation OMNeT ++. Il fournit des protocoles, des agents et d'autres modèles aux chercheurs et aux étudiants travaillant avec des réseaux de communication. INET est particulièrement utile lors de la conception et de la validation de nouveaux protocoles, ou lors de l'exploration de scénarios nouveaux ou exotiques. [18]. Nous avons utilisé la versions (2.5.0).

2.4.Veins :

Veins est un simulateur open source. Il est basé sur deux simulateurs généralement utilisés tels que OMNeT ++ et SUMO. Les modèles de simulateur de veines sont exécutés par un simulateur OMNeT ++ tout en interagissant avec un simulateur SUMO. D'autres composants de Veins prennent en charge d'autres fonctions telles que la configuration, l'exécution et le suivi de la simulation. Les veines comprennent un grand nombre de modèles de simulation qui peuvent être utilisés pour la simulation de réseaux de véhicules. Ils ne sont pas tous nécessaires dans chaque simulation mais, pour certains, il est logique d'instancier au plus un dans une simulation donnée.

Comme mentionné précédemment, pour exécuter les simulations de veines, les autres simulateurs tels que SUMO et OMNET ++ doivent être exécutés en parallèle pour la connexion de socket TCP. Le protocole pour cette communication respective est normalisé en tant qu'Interface de contrôle du trafic (TraCI). La simulation bidirectionnelle couplée du trafic routier et du trafic réseau permet protocole. Le mouvement des véhicules dans le simulateur de trafic routier, SUMO est reflété comme le mouvement des nœuds dans une simulation OMNET ++.[16]

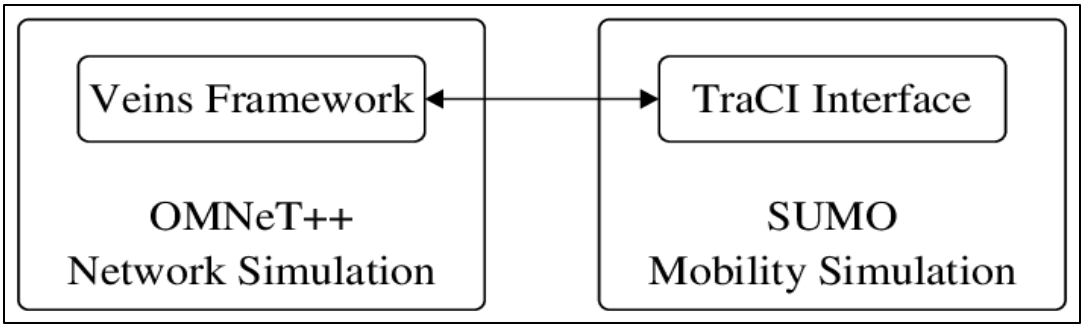

Figure 23 : Structure du Veins

2.5.Sumo :

SUMO est un package de simulation de trafic open source comprenant des composants d'importation nette et de modélisation de la demande. SUMO aide à étudier plusieurs sujets de recherche, par exemple choix d'itinéraire et algorithme de feux tricolores ou simulation de communication véhiculaire. Par conséquent, le cadre est utilisé dans différents projets pour simuler des stratégies de conduite automatique ou de gestion du trafic. L'application probablement la plus populaire de la suite SUMO est la modélisation du trafic dans le cadre de la recherche sur la communication V2X (véhicule-véhicule et véhicule-infrastructure).

Ici, SUMO est généralement couplé à une simulation de communication externe, telle que ns2 ou ns3 utilisant TraCI. Pour obtenir un environnement fonctionnel pour la simulation des communications véhiculaires, une instance d'application qui modélise l'application V2X à simuler est nécessaire. De plus, un mécanisme de synchronisation et d'échange de messages doit être impliqué.

2.6.iCanCloud :

iCanCloud est une plate-forme de simulation visant à modéliser et simuler des systèmes de cloud, qui s'adresse aux utilisateurs qui traitent étroitement avec ces types de systèmes. L'objectif principal d'iCanCloud est de prédire les compromis entre le coût et les performances d'un ensemble donné d'applications exécutées dans un matériel spécifique, puis de fournir aux utilisateurs des informations utiles sur ces coûts.

Cependant, iCanCloud peut être utilisé par un large éventail d'utilisateurs, des utilisateurs actifs de base aux développeurs de grandes applications distribuées. [17]

2.7.Veins lte:

Veins LTE est un simulateur pour les réseaux de véhicules hétérogènes. Il fournit une simulation fine des réseaux de véhicules basés sur IEEE 802.11p et TE

2.8.Language de simulation :

Le facteur déterminant le temps d'exécution de la simulation est le language de programmation utilisée. Les simulations sous OMNet++ peuvent être programmées en C ou en C++. Ce sont ces languages qui fournissent les meilleurs temps d'éxécution.

2.9.Système d'exploitation :

Ubuntu Système d'exploitation OpenSource, Nous avons utilisé la distribution Ubuntu 14.04 dans **PC *Acer*** de processeur Intel Core i3-4005 CPU @ 1.70GHz avec 4GB de RAM.

3. Les principaux fichier d'Omnet++

OMNET++ est composé par différents principaux fichiers sont :

3.1.Fichier (.ned) : Pour décrire la topologie et la structure de simulation On utilise le langage NED .Cela se fait en créant un module simple et complexe avec l'aide de Veins, en utilisant les deux modes. (Texte dans la Figure-24 et mode graphique dans la Figure-25)

Scenario.ned

```
network Scenario
{
    parameters:
        double playgroundSizeX @unit(m); // x size of the area the nodes are in (in meters)
        double playgroundSizeY @unit(m); // y size of the area the nodes are in (in meters)
        double playgroundSizeZ @unit(m); // z size of the area the nodes are in (in meters)
        @display("bgb=$playgroundSizeX,$playgroundSizeY");
    submodules:
        obstacles: ObstacleControl {
            @display("p=240,50");
        }
        annotations: AnnotationManager {
            @display("p=260,50");
        }
        connectionManager: ConnectionManager {
            parameters:
                @display("p=150,0;i=abstract/multicast");
        }
        world: BaseWorldUtility {
            parameters:
                playgroundSizeX = playgroundSizeX;
                playgroundSizeY = playgroundSizeY;
                playgroundSizeZ = playgroundSizeZ;
                @display("p=30,0;i=misc/globe");
        }
        Manager: TraCIScenarioManagerLaunchd {
            parameters:
                @display("p=512,128");
        }
```

Design Source

Figure 24 : Fichier NED en mode texte.

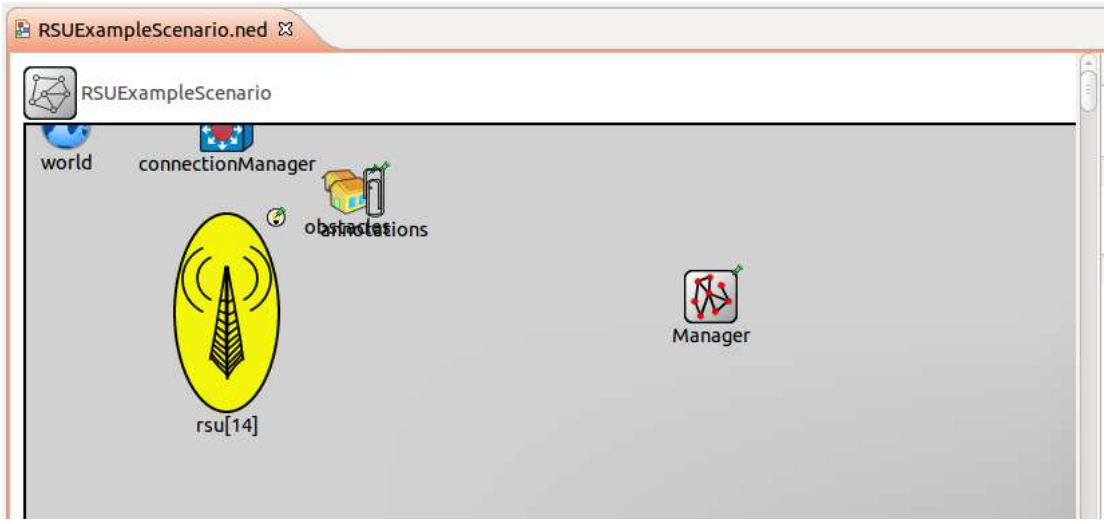

Figure 25 : Fichier NED en mode graphique.

3.2.Fichier (.msg) : Les modules communiquent entre eux en échangeant des messages. Cela peut être déclaré dans un fichier avec une extension (.msg) ou vous pouvez ajouter des champs de données. OMNeT ++ traduira les définitions de message en classes C ++.

3.3.Fichier (.ini) : C'est le fichier principal à partir duquel vous commencez l'exécutions, Il est étroitement lié au fichier NED. Permet à l'utilisateur de configurer divers paramètres de module ainsi que des topologies de réseau. Voici un exemple présenté ci-dessous.

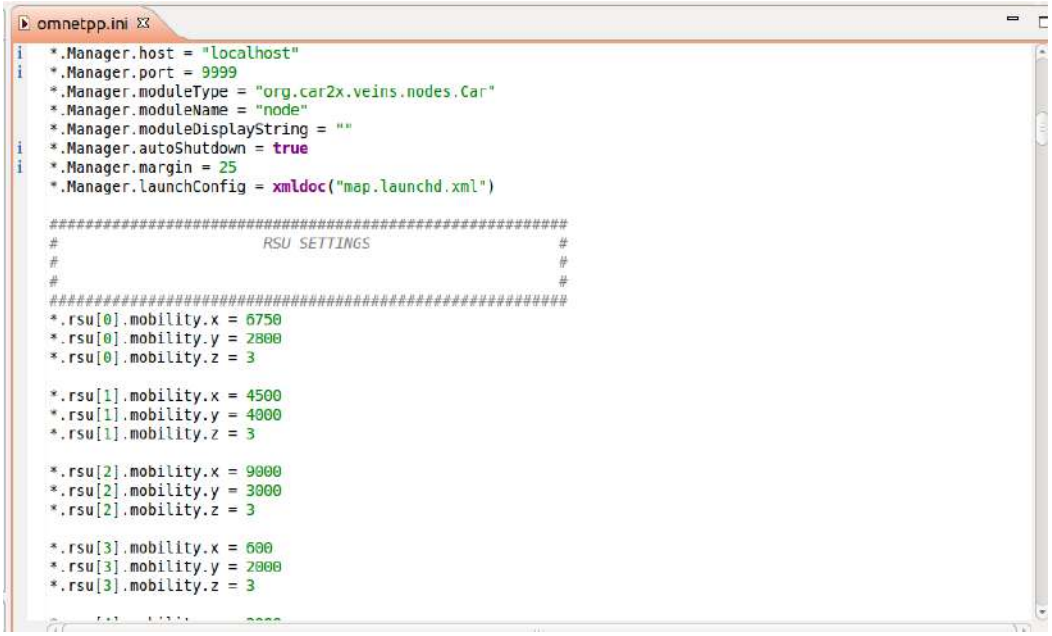

Figure 26 : Exemple d'un Fichier (.ini).

4. Les étapes de simulation

A l'aide de la framework Veins nous avons réalisé une architecture avec le simulateur Omnet++ qui comprend une suite complète des modèles pour rendre les simulations de réseau de véhicules aussi réalistes que possible, sans sacrifier la vitesse, et le package l'Inet pour la simulation des réseaux Informatique .il permet de la simulation des réseaux sans fil et mobiles ainsi les réseaux ad-hoc. Inet contient les différents protocoles de l'architecture tels que 802.11 et des protocoles implémentés, et plusieurs modèles d'application.

4.1.La première étape (Création de plateforme) :

Cette étape représente la plateforme qui nous avons choisi dans notre étude, nous avons importé INET, VEINS, la dans l'espace de travail d'Omnet et construire le projet pour préparer l'environnement de travail avec les versions compatibles Inet-2.5, Veins-4.4. En suit télécharger la platform de **Ouargla** comme un fichier (map.osm) de site http://openstreetmap.org/ comme montre dans la figure-27, et créé les fichier (map.osm.xml), (map.poly) et (map.rou) pour créer les aléatoire mobilité des véhicules.

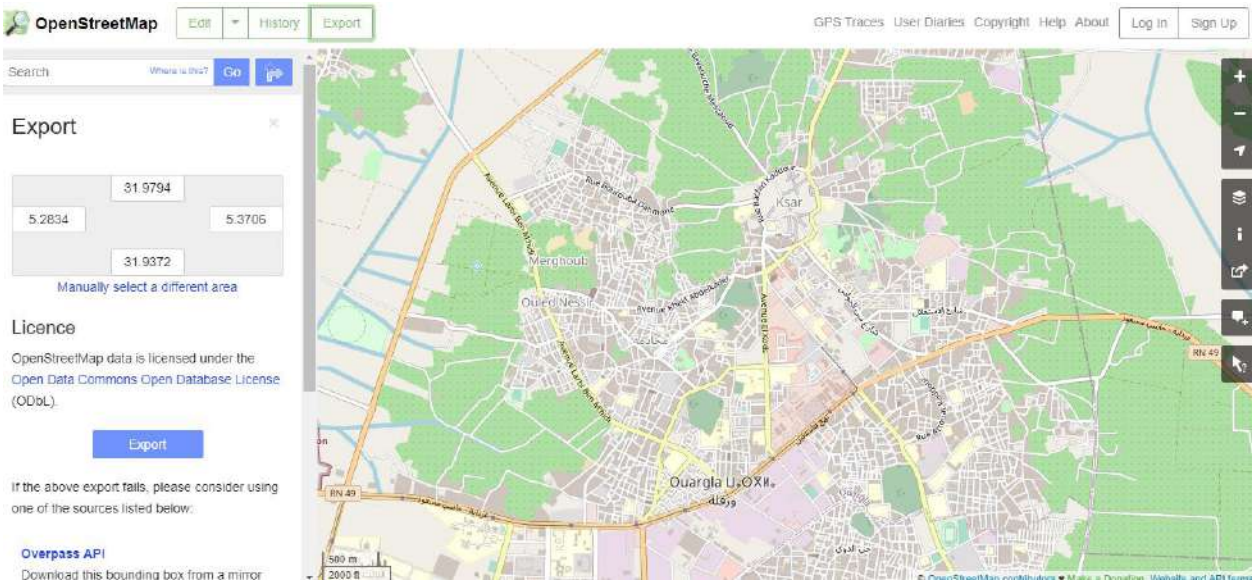

Figure 27 : Plateforme de wilayat Ouargla de Openstreetmap.

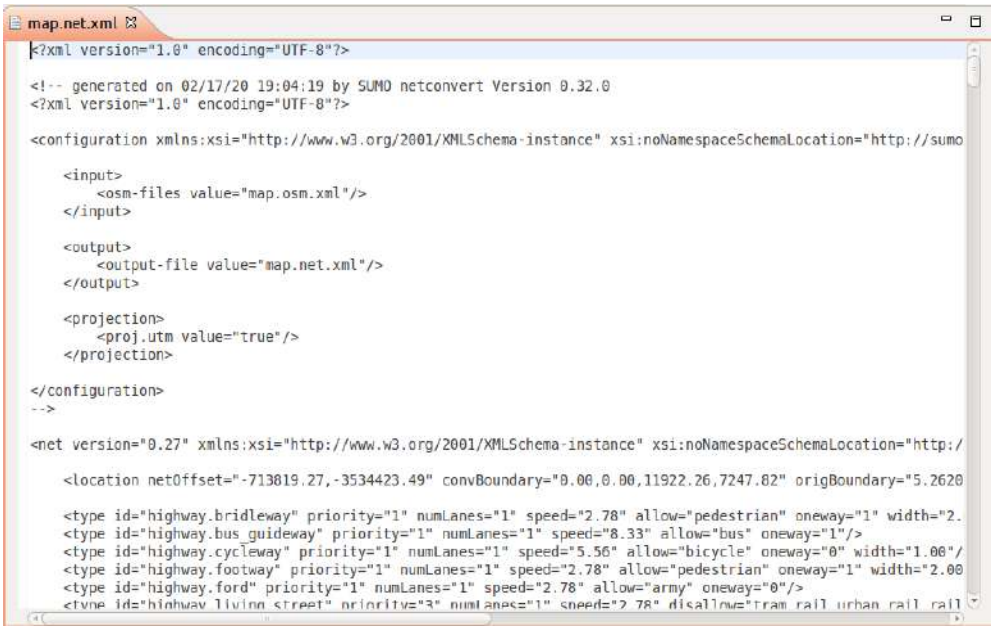

Figure 28 : fichier map.osm.xml

4.2.La deuxième étape (la connexion veins-sumo) :

Pour configurer et exécuter des simulations de manière interactive nous utilisons le simulator sumo qui est utilisé aux fins de recherche telles que la prévision du trafic.

L'appeler le serveur sumo-viens à partir du port 9999 par la commande suivante : "/home/ubuntu/Downloads/veins-veins-4.4/sumo-launchd.py -vv -c /home/ubuntu/Downloads/sumo-0.25.0/bin/sumo", pour créer une connexion entre l'Omnet et SUMO, le serveur attend ici que Omnet connecter. Ci-dessous l'illustration dans la Figure-29.

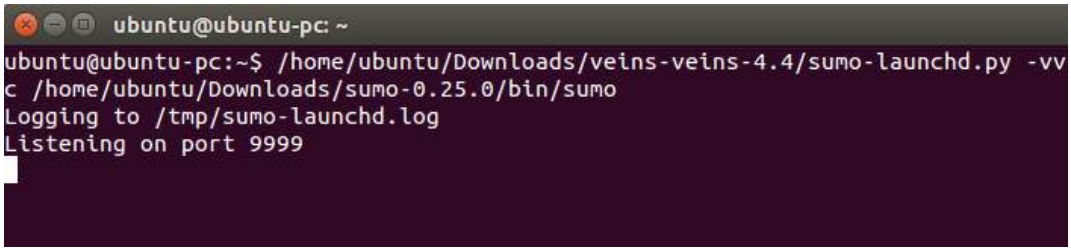

Figure 29 : la connexion de socket TCP du serveur sumo.

Après qu'on lance l'exécution à partir du fichier omnetpp.ini on obtient la topologie qu'on veut la simuler comme le monter dans le Figure-30

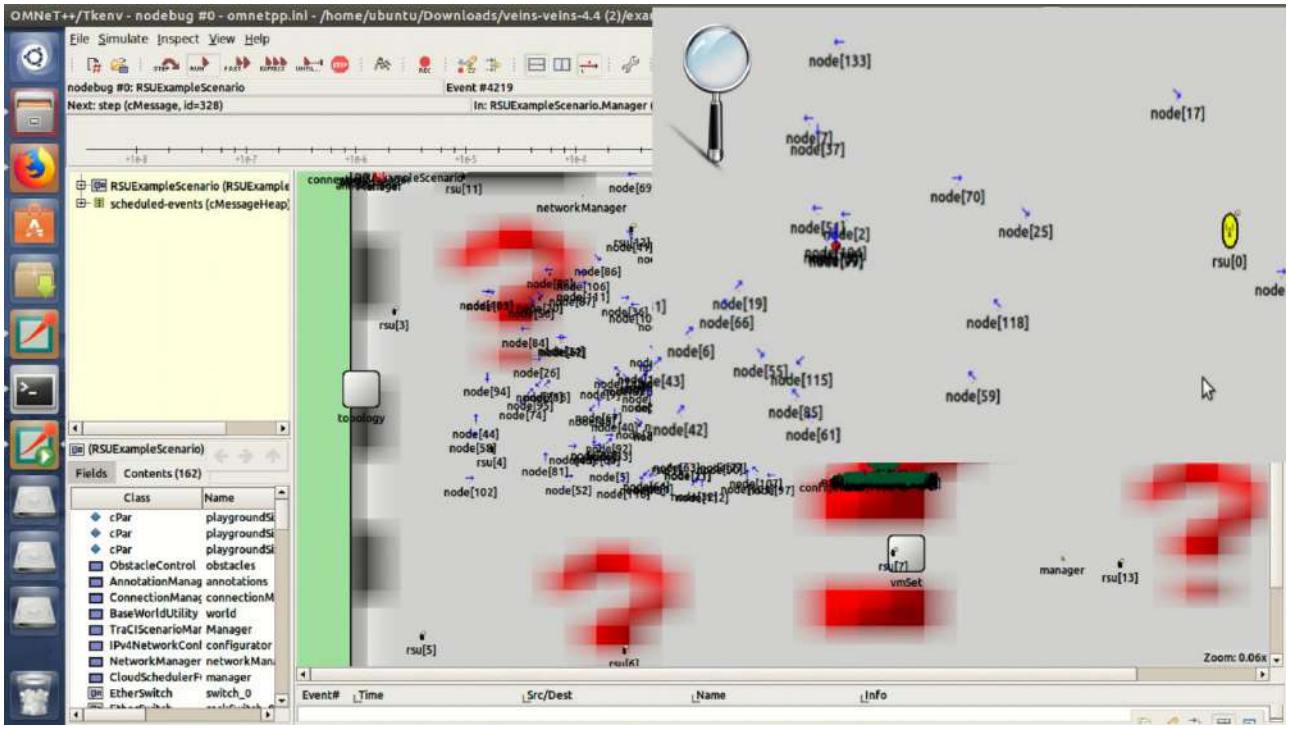

Figure 30 : résultat de simulation du réseaux VANET

Comme une résulta, grâce à l'application de la technologie DSRC et LTE, cette simulation est clarifiée bien la relation entre véhicules et véhicules, véhicule et l'infrastructure (RSU).

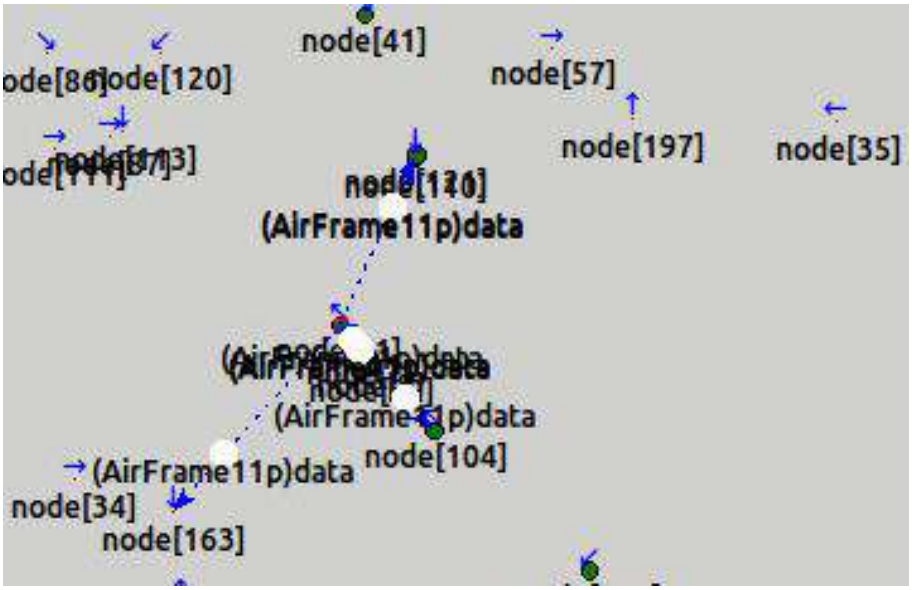

Figure 31 : communication entre les véhicules (V2V) via (DSRC)

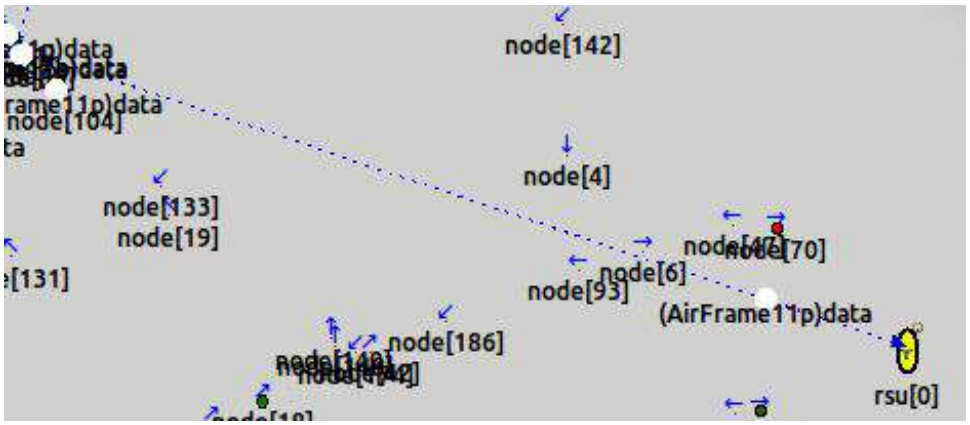

Figure 32 : communication entre les véhicules et les infrastructures (LTE)

4.3.La troisième étape (Cloud et déterminer le Fog dans la plateforme IoV) :

Nous avons utilisé le cloud computing pour enregistrer et stocker les données grâce à iCanCloud qui simuler le stockage des données au niveau de cloud, la liaison de cloud et la plateforme comme montré dans Figure-33.

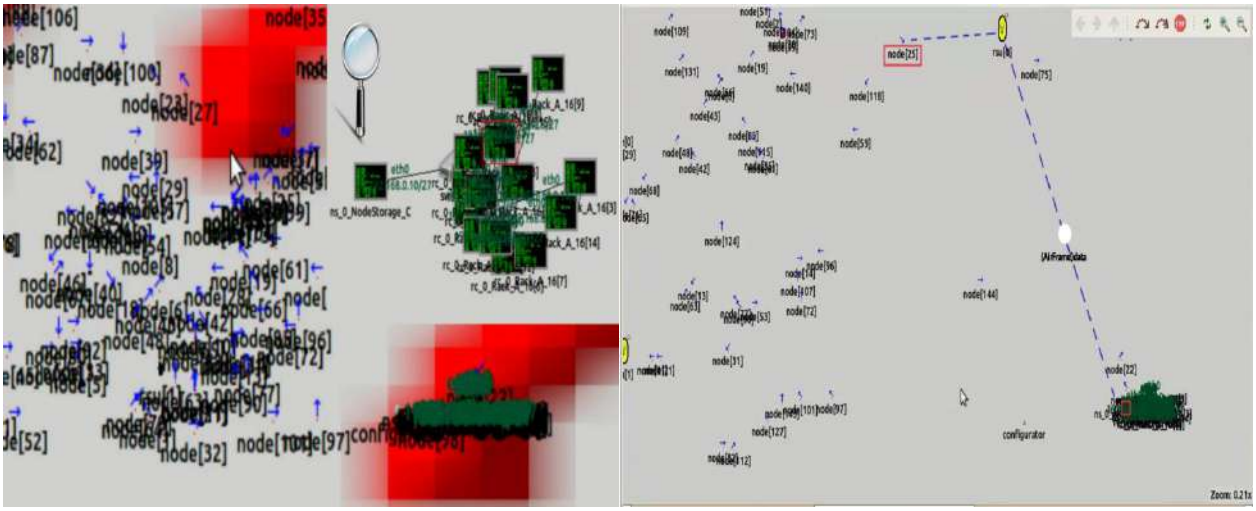

Figure 33 : Communication avec le cloud par la couche Fog.

Et selon l'architecture que nous avons adoptée, qui se compose de trois couches, nous avons choisi les RSUs comme une couche de nœud fog, qui se trouve au milieu des deux couches cloud et IOV. C'est comme un pont ou un médiateur entre les deux couches.

4.4.La quatrième étape (L'importation de protocole httpREST)

Dans cette étape, il s'agit d'importer le protocole httpREST dans la plateforme iov et de le modifier pour l'utiliser dans le transfert des données de la couche Vanet (véhicules) vers le cloud, en passant par la couche Fog, les étapes sont résumées comme suit :

4.4.1. Copiez le dossier contenant les fichiers de code source (* .cc et * .h) dans le répertoire "veins/src/ veins /application/application /".

4.4.2. Copiez le dossier contenant le fichier de configuration pour les paramètres de simulation (* .ini) dans le répertoire "veins/exampels/veins".

Code httpREST en C++ [33]

```
public class Product
{
   public int Id { get; set; }
   public string Name { get; set; }
   public int Qty { get; set; }
   public decimal Price { get; set; }
}
public class ProductsController : ApiController
{
   static Dictionary<long, Product> products = new Dictionary<long, Product>();
[HttpPost]
public IHttpActionResult Create([FromBody] Product item)
{
   if (item == null)
   {
      return BadRequest();
   }
   products[item.Id] = item;
   return Ok();
}
```

```
[HttpGetAll]
public List<Product> GetAll()
{
   List<Product> temp = new List<Product>();
   foreach(var item in products)
   {
      temp.Add(item.Value);
   }
   return temp;
}
[HttpGetid]
public IHttpActionResult GetProduct(long id)
{
   try
   {
      return Ok(products[id]);
   }
   catch
(System.Collections.Generic.KeyNotFoundException)
   {
      return NotFound();
   }
}
```

```
[HttpDelete]
public IHttpActionResult Delete(long id)
{
   var product = products[id];
   if (product == null)
   {
      return NotFound();
   }
   products.Remove(id);
   return Ok();
}
```

```
[HttpPut]
public IHttpActionResult Update(long id, [FromBody] Product item)
{
   if (item == null || item.Id != id)
   {
      return BadRequest();
   }
   if(products.ContainsKey(id)==false)
   {
      return NotFound();
   }
   var product = products[id];
   product.Name = item.Name;
   product.Qty = item.Qty;
   product.Price = item.Price;
   return Ok();
}
```

```
#include <cpr/cpr.h>
auto r = cpr::Post(cpr::Url{ "http://localhost:51654/api/products/create" },
   cpr::Body{ R"({"Id":1, "Name":"ElectricFan","Qty":14,"Price":20.90})" },
   cpr::Header{ { "Content-Type", "application/json" } });
auto r = cpr::Get(cpr::Url{ "http://localhost:51654/api/products/1" });
```

```
int main()
{
   {   std::cout << "Action: Create Product with Id = 1" << std::endl;
       auto r = cpr::Post(cpr::Url{ "http://localhost:51654/api/products/create" },
          cpr::Body{ R"({"Id":1,
          "Name":"ElectricFan","Qty":14,"Price":20.90})" },
          cpr::Header{ { "Content-Type", "application/json" } });
       std::cout << "Returned Status:" << r.status_code << std::endl;   }
   {   std::cout << "Action: Retrieve the product with id = 1" << std::endl;
       auto r = cpr::Get(cpr::Url{ "http://localhost:51654/api/products/1" });
       std::cout << "Returned Text:" << r.text << std::endl;
   }
   {   std::cout << "Action: Update Product with Id = 1" << std::endl;
       auto r = cpr::Post(cpr::Url{ "http://localhost:51654/api/products/1" },
          cpr::Body{ R"({"Id":1,
          "Name":"ElectricFan","Qty":15,"Price":29.80})" },
          cpr::Header{ { "Content-Type", "application/json" } });
       std::cout << "Returned Status:" << r.status_code << std::endl;
   }
   {  std::cout << "Action: Retrieve all products" << std::endl;
       auto r = cpr::Get(cpr::Url{ "http://localhost:51654/api/products" });
       std::cout << "Returned Text:" << r.text << std::endl;
   }
   {  std::cout << "Action: Delete the product with id = 1" << std::endl;
       auto r = cpr::Delete(cpr::Url{ "http://localhost:51654/api/products/1" });
       std::cout << "Returned Status:" << r.status_code << std::endl;
   }
return 0;
}
```

5. Conclusion

Le protocole httpREST est dérivé de l'Internet des objets, et comme l'internet de véhicule est une branche de celui-ci, nous voulions obtenir de meilleurs résultats lors de son utilisation par rapport aux résultats d'autres protocoles. Dans notre tentative de modifier le protocole pour l'implémenter sur la plate-forme Omnet, nous avons obtenu des résultats insatisfaisants. Donc, nous laissons notre humble réalisation comme un point de départ à ceux qui nous succéderont pour continuer.

Conclusion Général et Perspectives

Les VANETs sont une particularité des réseaux MANETs. Ils permettent de faire une communication entre véhicules intelligents équipés de calculateurs, de périphériques réseau et de différents types de capteurs.

Les réseaux véhiculaires sont une projection des systèmes de transports intelligents (Intelligent Transportation System – ITS). Leur objectif principal est d'améliorer la sécurité routière par l'utilisation de la technologie des communications et de l'émergence de dispositif sans fil à faible coût.

Les systèmes de transport intelligents ne sont qu'à leurs balbutiements. A termes, le développement des technologies a favorisé une évolution remarquable des réseaux véhiculaires. Cette évolution a rendu les réseaux plus efficaces, plus fiables, plus sûrs et plus écologiques aussi bien du point de vue de l'industrie automobile que des opérateurs de réseaux et de services.

Par ailleurs, les VANETs commencent à être appréciés et reconnus par les usagers des véhicules et sont de plus en plus utilisés dans différents pays. Ils aident les utilisateurs dans leur conduite et minimisent les risques d'accident et offrent un plus de confort aux usagers et la possibilité d'accès à des loisirs. Pour réduire la concentration et le stockage au niveau du cloud computing dans les réseaux véhiculaires nous suggérons la solution de fog computing car c'est une bonne solution.

Bien qu'en général, httpREST présente l'une des options de protocole les plus stables, il existe encore quelques problèmes qui ont conduit à l'exploration de solutions de protocole alternatives, en raison de leur complexité, des longs champs d'en-tête et de la consommation d'énergie élevée. De plus, la surcharge du protocole TCP peut être trop importante car il a besoin de temp pour connecter par rapport à UDP, en particulier dans le cas de nœuds de calcul simples dans les architectures IoV.

Bibliographie

[1] Définition de réseaux ad-hoc, 14-avril-2020, URL :http://www.tech-faq.com/ad-hoc-network.html/

[2] Toh, C. K. (1997). Wireless ATM & Ad Hoc Networks, 1997, Kluwer Academic Press. ISBN 9780792398226

[3] SAIDI, Abdessamad et MAMEM, Wafa. Amélioration des performances du protocole de routage EGyTAR dans les réseaux VANETs. Thèse de master. 15-01-2018.

[4] LOTFI, Morad et MOUNA, Boussehal. Dissémination de donnée dans un réseau de capteurs véhiculaires (VSN). 2014.

[5] YANG, Fangchun, WANG, Shangguang, LI, Jinglin, et al. An overview of internet of vehicles. China communications, 2014, vol. 11, no 10, p. 1-15.

[6] UR REHMAN, Sabih, KHAN, M. Arif, ZIA, Tanveer A., et al. Vehicular ad-hoc networks (VANETs)-an overview and challenges. Journal of Wireless Networking and Communications, 2013, vol. 3, no 3, p. 29-38.

[7] ASSIA, ABBAS, et al. DETECTION D'INTRUSION DANS LES RESEAUX VANETS. 2016. Thèse de doctorat.

[8] HUANG, Cheng, LU, Rongxing, et CHOO, Kim-Kwang Raymond. Vehicular fog computing: architecture, use case, and security and forensic challenges. IEEE Communications Magazine, 2017, vol. 55, no 11, p. 105-111.

[9] CHAHRA, Adel, OUYAHIA, Samira, FERHI, Kahina, et al. Etude et proposition d'un nouveau protocole de dissémination de données dans les VANETs. 2017. Thèse de doctorat. Université Abderrahmane Mira.

[10] YANG, Fangchun, WANG, Shangguang, LI, Jinglin, et al. An overview of internet of vehicles. China communications, 2014, vol. 11, no 10, p. 1-15.

[12] Définition de omnet++ 18-avril-2020, URL :https://doc.omnetpp.org/omnetpp/manual/

[13] DKHIL, Hassen. Greedy perimeter stateless routing sur omnet++. 2009. PhD Thesis. Master's thesis, Ecole nationale supérieur d'informatique, Tunisie, 2009.(Cité en pages vii et 3.).

[14] XIAN, Xiaodong; SHI, Weiren; HUANG, He. Comparison of OMNET++ and other simulator for WSN simulation. In: 2008 3rd IEEE Conference on Industrial Electronics and Applications. IEEE, 2008. p. 1439-1443.

[15] Extensions du simulateur Omnet++ pour la validation de mécanismes de transmission multimédia dans les réseaux IEEE 802.11 par Ahmed Ayadi, Ecole Nationale des Sciences de l'Informatique - Ingénieur informatique 2007

[16] Définition de Veins, 2-août-2020 URL :http://veins.car2x.org/documentation/.

[17] source site de iCanCloud, 4-août-2020, URL : (https://www.arcos.inf.uc3m.es/old/icancloud/Home.html).

[18] Définition de inet,28-mars-2020, URL : https://inet.omnetpp.org/Introduction.html

[19] SATTAR, Mohd Abdul; ANWARUDDIN, Mohammed; ALI, Mohd Anas. A Review on Internet of Things-Protocols Issues. International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering, 2017, 5.2.

[20] DA SILVA, Aloizio Pereira; BURLEIGH, Scott; OBRACZKA, Katia (ed.). Delay and Disruption Tolerant Networks: Interplanetary and Earth-Bound--Architecture, Protocols, and Applications. CRC Press, 2018.

[21] les protocoles d'IoT, 15-aoùt-2020, URL : https://www.opensourceforu.com/2017/07/internet-things-protocols-landscape/

[22] DIZDAREVIĆ, Jasenka, et al. A survey of communication protocols for internet of things and related challenges of fog and cloud computing integration. ACM Computing Surveys (CSUR), 2019, 51.6: 1-29.

[23] Mqtt,17-mai-2020,URL :https://www.velotio.com/engineering-blog/mqtt-protocol-overview

[24] BANKS, Andrew; GUPTA, Rahul. MQTT Version 3.1. 1. OASIS standard, 2014, 29: 89.

[25] le protocole http,17-mai-2020, URL :https://www.w3schools.com/whatis/whatis_http.asp

[26] W. Shang, Y. Yu, R. E. Droms, and L. Zhang. Challenges in iot networking via tcp/ip architecture. Number 2, page 7,2016.

[27] T. Dierks and E. Rescorla. The transport layer security (tls) protocol version 1.2. RFC 5246, RFC Editor, August 2008.

[28] K. Golestan, R. Soua, F. Karray, and M. S. Kamel, ''Situation awareness within the context of connected cars: A comprehensive review and recent trends,'' Inf. Fusion, vol. 29, pp. 68–83, May 2016.

[29] Bonomi, F.: The smart and connected vehicle and the Internet of Things,WSTS 2013, San Jose.

[30] J. Contreras, S. Zeadally, and J. A. Guerrero-Ibanez, "Internet of vehicles: Architecture, protocols, and security," IEEE Internet of Things Journal, 2017.

[31] TUYISENGE, Livinus, et al. Network Architectures in Internet of Vehicles (IoV): Review, Protocols Analysis, Challenges and Issues. In: *International Conference on Internet of Vehicles*. Springer, Cham, 2018. p. 3-13.

[32] YANG, Fangchun, et al. Architecture and key technologies for Internet of Vehicles: a survey. 2017.

[33]code httpREST en C++,3-septembre -2020 URL : https://www.codeproject.com/Articles/1244632/Making-HTTP-REST-Request-in-Cplusplus

[35] LTE, vendredi octobre 9 2020 09h:20,
http://dictionnaire.sensagent.leparisien.fr/LTE%20(r%C3%A9seaux%20mobiles)/fr-fr/

[36] BRADLEY, James Frederick; COOPER, Paul W. Method and apparatus for providing interactive two-way communications using a single one-way channel in satellite systems. U.S. Patent No 6,243,366, 2001.