UNIVERSITÉ CHEIKH ANTA DIOP DE DAKAR

ÉCOLE DOCTORALE MATHÉMATIQUES ET INFORMATIQUE

ÉCOLE SUPÉRIEURE POLYTECHNIQUE DE DAKAR

Année : 2020 N° d'ordre : 145

THESE DE DOCTORAT UNIQUE

Présentée pour obtenir le grade de Docteur de l'Université Cheikh Anta Diop de Dakar

Mention : Informatique et Télécommunications

Spécialité : Télécommunications

Présentée par

Ousmane SADIO

Titre : Hyperviseur de Passerelles/Interfaces Sans Fil pour un Laboratoire Mobile de Nouvelle Génération

Soutenue le : 08/02/2020 devant le jury composé de :

Président	Hamidou	DATHE	Professeur	UCAD, Dakar
Rapporteurs	Michel	BABRI	Professeur	INPHB, Yamoussoukro
	Karim	KONATE	Professeur	UCAD, Dakar
Examinateurs	Ahmed Dooguy	KORA	Professeur	ESMT, Dakar
	Samuel	OUYA	Maitre de Conférence	UCAD, Dakar
Directeur de Thèse	Claude	LISHOU	Professeur	UCAD, Dakar

REMERCIEMENTS

Je tiens d'abord à remercier :

Pr Claude LISHOU, Ecole Supérieure Polytechnique de Dakar, mon Directeur de Thèse. J'aimerais à présent vous exprimer toute ma gratitude et toute ma reconnaissance. C'est un immense plaisir de vous avoir comme directeur de thèse, avec vos qualités scientifiques, pédagogiques et humaines. Vos conseils, remarques et critiques avisées m'ont permis de mieux mettre en valeur mes travaux de thèse. Merci pour votre confiance, votre encadrement et vos encouragements.

Je tiens également à adresser mes sincères remerciements aux autres membres du jury pour m'avoir fait l'honneur de leur présence et pour l'intérêt qu'ils ont porté à mon travail :

Pr Hamidou DATHE, Université Cheikh Anta Diop de Dakar, Directeur de l'École Doctorale Mathématiques et Informatique, pour m'avoir accepté dans l'école doctorale pour y mener mes travaux de recherches, et pour avoir accepté de présider ce jury. Merci également pour m'avoir enseigné les bases de l'Analyse Mathématiques en deuxième année de Physique Chimie à la FST.

Pr Michel BABRI, Institut National Polytechnique Félix Houphouët-Boigny de Yamoussoukro, qui m'a fait l'honneur de juger ce travail et d'en être rapporteur. Je vous remercie pour avoir consacré du temps à la lecture de mon manuscrit et pour vos remarques constructives qui m'ont permis d'améliorer la qualité de ce manuscrit.

Pr Karim KONATE, Université Cheikh Anta Diop de Dakar, qui m'a fait l'honneur de juger ce travail et d'en être rapporteur. Je vous remercie pour avoir consacré du temps à la lecture de mon manuscrit et pour vos remarques constructives qui m'ont permis d'améliorer la qualité de ce manuscrit.

Pr Ahmed Dooguy KORA, Ecole Supérieure Multinationale Des Télécommunications de Dakar, pour avoir accepté d'examiner ce travail. Je vous témoigne toute ma reconnaissance.

Dr Samuel OUYA, Ecole Supérieure Polytechnique de Dakar, pour avoir accepté d'examiner ce travail. Je tiens aussi à vous remercier pour m'avoir accepté dans votre laboratoire LIRT depuis mon Master de Recherche et pour vos conseils et m'avoir fait profiter de votre expérience. Merci également pour m'avoir transmis, en qualité d'enseignant en Master de Recherche de l'ESP, des connaissances qui ont été d'une grande utilité pour ma thèse.

Je tenais également à remercier en particulier,

Dr Ibrahima NGOM, Ecole Supérieure Polytechnique de Dakar, pour votre confiance et votre grand soutien tout au long de mes travaux de thèse et de Master de Recherche. Merci pour avoir relu mon manuscrit, de m'avoir fait profiter de votre expérience et de toutes idées pertinentes qui ont contribuées qui ont contribuées à améliorer la qualité de ce manuscrit.

Je remercie également,

Dr Gervais Mendy et **Dr Idy DIOP**, Ecole Supérieure Polytechnique de Dakar, pour leurs conseils, soutien et encouragements.

Tous les chercheurs des laboratoires LIRT / LTI / LIMBI de l'Ecole Supérieure Polytechnique de Dakar.

Tout le personnel enseignant et administratif du Département Génie Informatique de l'Ecole Supérieure Polytechnique de Dakar.

DÉDICACE

Je dédie cette thèse :

A toute ma famille

A mon père

A ma mère

A mes frères et sœurs

A mes oncles et tantes

A tous mes amis

TABLE DES MATIÈRES

LISTE DES FIGURES

LISTE DES TABLEAUX

LISTE DES ABRÉVIATIONS, SIGLES ET ACRONYMES

ABC	Always Best Connected
AHP	Analytic Hierarchy Process
AODV	Ad-hoc On-demand Distance Vector
ADAS	Advanced Driver Assistance Systems
AP	Access Point
API	Application Programming Interface
ARP	Address Resolution Protocol
ASTM	American Society for Testing and Materials
AU	Application Units
BEX	Base Exchange
BSSID	Basic Service Set Identifier
CAN	Control Area Network
C2C-CC	CAR-to-CAR Communication Consortium
CPDB	Context Proprieties DataBase
DAD	Duplicate Address Detection
DHCP	Dynamic Host Configuration Protocol
DSRC	Dedicated Short-Range Communications
DTN	Delay-Tolerant Networking
ECC	Electronic Climate Control
ECU	Electronic Control Units
EDCA	Enhanced Distributed Channel Access
ESP	Encapsulating Security Payload
GPS	Global Positioning System
GPSR	Greedy Perimeter Stateless Routing
HI	Host Identifier
HIP	Host Identity Protocol
HIT	Host Identity Tag
HVAC	Heating Ventilation & Air Conditioning
IEEE	Institute of Electrical and Electronics Engineers

IETF	Internet Engineering Task Force
IoT	Internet of Things
IoV	Internet of Vehicles
IRTF	Internet Research Task Force
ISN	Intelligent Sensors Network
ISO	International Organization for Standardization
ITS	Intelligent Transportation Systems
ITU	International Telecommunications Union
IVGW	In-Vehicle Gateway
IVI	In-Vehicle Infotainment
IVSN	Intelligent Vehicular Sensor Network
LM	Laboratoire Mobile
LSI	Local-Scope Identifier
LVAP	Light Virtual Access Point
MAG	Mobile Anchor Gateway
MANET	Mobile Ad Hoc Network
MIPv6	Mobile IPv6
MIS	Media Independent Service
MISF	MIS Functions
MN	Mobile Node
MNN	Mobile Network Nodes
MR	Mobile Router
ND	Neighbor Discovery
NEMO	Network Mobility
NFV	Network Functions Virtualization
OBU	On-Board Units
OLSR	Optimized Link State Routing Protocol
PMIPv6	Proxy Mobile IPv6
PoA	Point of Attachment
PLRVS	Proxy Local Rendezvous Server
RA	Router Advertisement
RAN	Radio Acces Network
RAT	Radio Access Technologies

RSSI	Received Signal Strength Indication
RSU	Roadside Unit
RVS	Rendez-Vous Server
S^2aaS	Sensing as a Service
SBC	Single Board Computer
SDN	Software Defined Networking
SDNRAN	SDN Radio Acces Network
SDNVANET	SDN Vehicular Ad Hoc Network
S-RVS	Subnet Rendezvous Server
SSID	Service Set identifier
SSN	Semantic Sensor Network
V2C	Vehicle-to-Cloud
V2I	Vehicle-to-Infrastructure
V2P	Vehicle-to-Pedestrian
V2V	Vehicle-to-Vehicle
V2X	Vehicle-to-Everywhere
VANET	Vehicular Ad Hoc Network
VSN	Vehicular Sensor Networks
WAVE	Wireless Access in Vehicular Environments
WSA	WAVE Service Advertisement
WSN	Wireless Sensor Network

PUBLICATIONS

I. Articles de journaux

1. IEEE Transactions on Vehicular Technology Special Issue on Vehicle Connectivity and Automation using 5G Networks – 2020
Sadio, Ousmane; Ngom, Ibrahima; Lishou, Claude; "Design and Prototyping of a Software Defined Vehicular Networking," *IEEE Transaction on Vehicular Technology*, vol. 69, no. 1, pp. 842–850, Jan. 2020, DOI: 10.1109/TVT.2019.2950426, ISSN: 1939-9359
https://ieeexplore.ieee.org/document/8887262

2. IEEE Sensors Journal – 2019
Sadio, Ousmane; Ngom, Ibrahima; Lishou, Claude; "A novel Sensing as a Service model based on SSN Ontology and Android Automotive," *IEEE Sensors Journal*, vol. 19, no. 16, pp. 7015–7026, Aug. 2019, DOI: 10.1109/JSEN.2019.2911913, ISSN: 1530-437X
https://ieeexplore.ieee.org/document/8693807

3. IEEE Access – 2020
Sadio, Ousmane; Ngom, Ibrahima; Lishou, Claude; "Controlling WiFi Direct Group Formation for Non-Critical Applications in C-V2X Network," *IEEE Access*, pp. 1–10
(Status – Under review)

II. Articles de conférences

4. IEEE IOTSMS - Granada (Spain) – 2019
O. Sadio, I. Ngom, and C. Lishou, "Lightweight Security Scheme for MQTT/MQTT-SN Protocol," *in 2019 Sixth International Conference on Internet of Things: Systems, Management and Security (IOTSMS)*, Granada (Spain), pp. 119–123, 22-25 Oct. 2019, DOI: 10.1109/IOTSMS48152.2019.8939177
https://ieeexplore.ieee.org/document/8939177

5. IEEE UKSim2018 - Cambridge (United Kingdom) – 2018
Sadio, Ousmane; Ngom, Ibrahima; Lishou, Claude; "SDN Architecture for Intelligent Vehicular Sensors Networks," in *2018 UKSim-AMSS 20th International Conference on Computer Modelling and Simulation (UKSim)*, Cambridge (United Kingdom), pp. 139 144, 27-29 Mar. 2018, DOI: 10.1109/UKSim.2018.00036
https://ieeexplore.ieee.org/document/8588192

6. IEEE ICCC 2017 - Chengdu (China) – 2017
Sadio, Ousmane; Ngom, Ibrahima; Lishou, Claude; "Rethinking Intelligent Transportation Systems with Internet of Vehicles: proposition of Sensing as a Service model," in *2017 3rd IEEE International Conference on Computer and Communications (ICCC)*, Chengdu (China), pp. 1-5, 13-16 Dec. 2017, DOI: 10.1109/CompComm.2017.8323041
http://ieeexplore.ieee.org/document/8323041

7. SPRINGER AECIA 2016 - Marrakech (Morocco) – 2016
Sadio, Ousmane; Ngom, Ibrahima; Lishou, Claude; “Performance Analysis of a Proposed Architecture for Remote Construction Machines,” *in Third International Afro-European Conference for Industrial Advancement (AECIA 2016), Advances in Intelligent Systems and Computing, vol 565. Springer, Cham,* Marrakech (Morocco), pp.1-10, 21-23 Nov. 2016, DOI:10.1007/978-3-319-60834-1_25
https://link.springer.com/chapter/10.1007/978-3-319-60834-1_25

8. IEEE SAI 2015 - London (United Kingdom) – 2015
Sadio, Ousmane; Ngom, Ibrahima; Lishou, Claude; Saliah-Hassane, Hamadou; “Enhanced controller of mobility for a new generation of mobile laboratory,” in *Science and Information Conference (SAI)*, London (United Kingdom), pp. 1018–1027, 28-30 July 2015, DOI: 10.1109/SAI.2015.7237267
http://ieeexplore.ieee.org/document/7237267

9. IEEE WSWAN 2015 - Sousse (Tunisia) – 2015
Ngom, Ibrahima; Sadio, Ousmane; Lishou, Claude; Mboup, Mamadou Lamine; Saliah-Hassane, Hamadou; “Wireless interworking gateways/interfaces hypervisor for mobile laboratory,” in *2015 2nd World Symposium on Web Applications and Networking (WSWAN)*, Sousse (Tunisia), pp. 1–7, 21-23 March 2015, DOI:10.1109/WSWAN.2015.7210359
https://ieeexplore.ieee.org/document/7210359

10. IEEE SPICES 2015 - Kozhikode (India) – 2015
Sadio, Ousmane, Ngom, Ibrahima; Lishou, Claude; Saliah-Hassane, Hamadou; “Improving security and mobility for remote access: A wireless sensor network case,” in *2015 IEEE International Conference on Signal Processing, Informatics, Communication and Energy Systems (SPICES)*, Kozhikode (India), pp. 1–5, 19-21 Feb. 2015, DOI:10.1109/SPICES.2015.7091369
https://ieeexplore.ieee.org/document/7091369

11. IEEE ICPADS 2014 – Hsinchu (Taiwan) – 2014
Ngom, Ibrahima; Sadio, Ousmane; Lishou, Claude; Saliah-Hassane, Hamadou; “Enhanced HIP-based micro-mobility and macro-mobility management by proactive signaling scheme,” in *2014 20th IEEE International Conference on Parallel and Distributed Systems (ICPADS)*, Hsinchu (Taiwan), pp. 780–786, 16-19 Dec. 2014, DOI:10.1109/PADSW.2014.7097882
https://ieeexplore.ieee.org/document/7097882

RÉSUMÉ

La technologie des capteurs automobiles progresse rapidement et constitue actuellement un élément essentiel dans la conception automobile moderne. L'avènement de l'Internet des Véhicules (IoV) et de la 5G va déclencher l'émergence d'un vaste éventail de services, notamment grâce à l'exploitation des données de capteurs. Cependant, compte tenu de la nature très intégrée des technologies automobiles, de l'hétérogénéité des propriétés des capteurs et de la dynamique des réseaux véhiculaires ; le défi majeur des applications/services IoV est de répondre à la *vision 4A* (Anywhere, Anytime, by Anyone and Anything).

Cette thèse a pour objectif de proposer des modèles de Laboratoires Mobiles (LM) et des modèles de communication pour la collecte optimale des données de capteurs automobiles. Cette collecte est essentielle pour la transformation de ces données en connaissances utiles. Ainsi, en premier lieu, un modèle de LM basé sur le concept de « data MULE » et sur le protocole HIP (Host Identity Protocol) est proposé. Ce modèle permet d'améliorer la sécurité et la gestion de la mobilité de HIP à travers des schémas de micromobilité et de macromobilité optimisés pour le LM. Les performances de ce modèle sont étudiées de façon analytique et expérimentale. Par la suite, un second modèle de LM est proposé. Le but est de réduire les délais engendrés par les opérations cryptographiques intensives du protocole HIP et, aussi, d'améliorer le modèle de communication du LM en tenant compte notamment des nouvelles possibilités de communication et de calcul proposées par l'IoV/5G. Ainsi, pour mieux s'approcher de la *vision 4A*, ce second modèle intègre plusieurs types de sensibilité (spatiale, de groupe, de données, de contexte) et s'appuie sur un concept cloud nommé Sensing as a Service (S^2aaS). L'ontologie SSN (Semantic Sensor Network) est employée pour représenter de façon homogène les données et les propriétés des capteurs automobiles, de même que pour optimiser leur découverte, classement et sélection. Pour faire abstraction de la nature très intégrées des technologies automobiles, une extension d'Android Automotive adaptée pour le LM est proposée. En dernier lieu, un modèle de réseau SDN (Software Defined Networking) véhiculaire intégral est proposé afin de rendre le réseau plus flexible et agile. Ainsi, des schémas SDN de handover réactifs/proactifs, intracanal/intercanal et intertechnologie sont expérimentés. De même, des schémas de routage SDN geocast/unicast adaptés pour le S^2aaS sont proposés et testés. L'ensemble des propositions faites dans cette thèse sont validées par le biais de modèles analytiques, de simulations et/ou de prototypes.

Mots clefs: Laboratoire Mobile, Internet des Véhicules, Sensing as a Service, Communication véhiculaire, Host Identity Protocol, Software Defined Networking

ABSTRACT

The automotive sensor technology is advancing rapidly and is actually a vital part of any modern automobile design. The advent of the Internet of Vehicles (IoV) and 5G has led to the emergence of a vast service market, particularly through the exploitation of sensor data. However, given the highly integrated nature of automotive technologies, the heterogeneity of sensor properties and the dynamics of vehicle networks; the major challenge of IoV applications/services is to meet the *4A vision* (Anywhere, Anytime, by Anyone and Anything).

The purpose of this thesis is to propose Mobile Laboratories (LM) models and communication models for optimal collection of automotive sensor data. This collection is essential for the transformation of this data into useful knowledge. First, an LM model based on the concept of "data MULE" and HIP protocol (Host Identity Protocol) is proposed. This model improves the security and the mobility management of HIP through micromobility and macromobility schemes optimized for LM. A HIP security enhancement scheme as well as optimized micromobility and macromobility schemes for LM are also proposed. The performances of this model are studied analytically and experimentally. Subsequently, a second model of LM is proposed. The goal is to reduce the delays caused by the intensive cryptographic operations of the HIP protocol and, also, to improve the LM communication model thanks to the new communication and computing possibilities offered by IoV/5G. To get closer to the 4A vision, this second model integrates several types of awareness (spatial, group, data, context) and is based on a cloud concept called Sensing as a Service (S^2aaS). The SSN (Semantic Sensor Network) ontology is used to uniformly represent the data and properties of automotive sensors, as well as to optimize their discovery, classification and selection. To ignore the highly integrated nature of automotive technologies, an extension of Android Automotive adapted for the LM is proposed. Finally, a Software Defined Vehicular Networking (SDVN) model is proposed to make the network more flexible and agile. Thus, reactive/proactive, intrachannel/interchannel and intertechnology SDN handover schemes are tested. Likewise, SDN-based geocast/unicast routing schemes adapted for S^2aaS are proposed and tested. All of the proposals made in this thesis are validated using analytical models, simulations and / or prototypes.

Keywords: Mobile Laboratory, Internet of Vehicles, Sensing as a Service, Vehicular communication, Host Identity Protocol, Software Defined Networking

Introduction

Grace aux Systèmes de Transports Intelligents (ITS, Intelligent Transportation Systems) et aux véhicules autonomes, la voiture se transforme progressivement en un système informatique intégral. La prochaine frontière de la révolution automobile tend vers les véhicules connectés et l'Internet des Véhicules (IoV, Internet of Vehicles) qui visent à transformer le véhicule en un bien plus qu'un simple « smartphone avec des roues ».

1. Genèse des véhicules connectés

Avec la demande « full option » pour les véhicules, les systèmes de contrôle et de régulation, qui premièrement étaient implémentés de façon mécanique, sont remplacés progressivement par des systèmes électroniques. Ces systèmes deviennent de plus en plus complexes avec l'intégration d'une multiplicité de capteurs, de systèmes de contrôles et d'interfaces exigeant beaucoup plus de bandes passantes. Ainsi, les véhicules modernes emploient généralement un nombre important de ECU (Electronic Control Units) où sont reliés un ensemble distribué et disparate de capteurs et d'actionneurs. Les ECU permettent l'échantillonnage et le traitement des données issues de ces capteurs et en même temps du transfert des commandes vers les actionneurs. Ainsi, on peut retrouver jusqu'à des centaines de capteurs servant à l'acquisition de données relatives à l'état des pièces mécaniques et électroniques, à la perception, à la mesure de grandeurs environnementales… Ces capteurs peuvent être regroupés suivant trois champs d'applications majeurs [1]–[3] : groupe motopropulseur, sureté et confort.

Les ordinateurs de bords (OBU, On-Board Units) ont été, à la base, développés pour des applications de tracking via GPS. Cependant, les capacités de l'OBU ont vite évolué avec notamment la possibilité de recueillir les données de capteurs, et la possibilité de pouvoir communiquer avec des entités externes au véhicule. Ces nouvelles possibilités ont été exploitées pour le développement d'applications [4]–[6] pour les VANET (Vehicular Ad Hoc Network). Ces applications étaient principalement liées à la sécurité/sureté routière, à l'efficacité du trafic et d'assistance à la conduite. La génération suivante de OBU est dédiée aux ITS et introduit le concept de la communication V2X (Vehicle-to-Everywhere) qui intègre beaucoup plus de types d'interactions via les technologies DSRC/WiFi/LTE. L'IoV introduit la « cloudification » des véhicules par la création de nouveaux services cloud. La cinquième génération de réseaux mobiles (5G) est censée répondre au exigences de communications V2X [7], en permettant, entre autres, aux services cloud de s'approcher de la vision 4A (Anywhere, Anytime, by Anyone and Anything).

Un véhicule connecté diffère des autres types de véhicules de part sa capacité à non seulement pas pouvoir communiquer, mais de pouvoir le faire à travers différentes voies technologiques [8]. Un véhicule connecté est ainsi imaginé comme un modèle multicommunicant, appelé communication V2X. Ces véhicules accèdent, consomment, créent, enrichissent, dirigent et partagent l'information entre les individus, organismes, milieux d'affaires, infrastructures, et autres objets connectés.

Selon les prévisions de Gartner [9], d'ici 2020, un véhicule sur cinq sera un véhicule connecté, soit environ un quart de milliard de véhicules. Par conséquent l'IoV occupera une place prédominante dans l'Internet des Objets (IoT, Internet of Things). Cet optimisme est réitéré dans un autre article de presse intitulé « Forget the Internet of Things : Here Comes the 'Internet of Cars' » [10], où T. Koslowski soutient que pour l'industrie automobile, connecter les véhicules est désormais le critère prédominant dans la phase d'innovation. Les consommateurs souhaitent accéder à l'information partout où ils se trouvent, même dans les véhicules. Ainsi, ils affichent un grand intérêt pour les fonctionnalités offertes par les véhicules connectés. On peut d'ailleurs citer la popularité et le succès des applications telles que Android Auto, Apple CarPlay ou encore Waze. En outre, un système d'exploitation dédié au véhicule sera opérationnel est en cours de développement ; il s'agit d'Android Automotive développé par Google.

En somme, avec le concept de l'IoV, les véhicules sont en passe de devenir plus connectés, plus intelligents, et intégreront un riche écosystème de services et d'applications.

2. Laboratoire Mobile de Nouvelle Génération

Le recours aux Laboratoires Mobiles (LM) vient répondre aux besoins exprimés par les acteurs de l'industrie, gouvernement, médecine, armée, organismes de recherche…

La première génération de LM fonctionnait à base d'un ensemble d'équipements de relève mobiles tractés le plus souvent par des véhicules. Ces équipements étaient composés d'ordinateurs, de système de stockage, de systèmes de communication et d'un ensemble d'instruments de mesure embarqués. Le véhicule peut aussi communiquer avec un réseau de capteurs sans fil (WSN, Wireless Sensor Network), dans ce cas précis le terme Data MULE (Mobile Ubiquitous LAN Extension) [11] est associé au véhicule. La transmission des données se fait via un réseau tolérant aux délais (DTN, Delay-Tolerant Networking). Le DTN s'appuie sur des connectivités alternatives (réseaux mobiles, WiMAX, satellites…), et est surtout adapté pour les zones à faible densité réseau. L'avantage de ce type de LM réside sur leurs faibles coûts de déploiement, par contre aucune « intelligence » n'est intégrée au véhicule, qui d'ailleurs est considéré comme une « mule ».

Les réseaux de capteurs véhiculaires (VSN, Vehicular Sensor Networks) [12], [13] sont un nouveau paradigme réseau qui hérite du concept de VANET et de WSN. Cela confère au VSN des propriétés uniques telles que le changement dynamique des zones d'intérêts, la collecte de données issues du véhicule lui-même et de son environnement immédiat. Plus récemment, le VSN s'est dissous dans un autre nouveau concept beaucoup plus large et plus prometteur ; il s'agit de l'IoV. Ce dernier s'appuie sur l'IoT pour faire évoluer le véhicule d'un simple nœud dans les VANET en une plateforme intelligente capable d'apprendre, de penser et de comprendre les systèmes cyberphysiques par lui-même [14]–[16]. Dans [17], les auteurs posent les principaux challenges à relever dans les réseaux de capteurs intelligents (ISN, Intelligent Sensors Network). Ainsi, en se basant sur le concept de l'ISN, l'intelligence des réseaux de capteurs peut être atteinte de quatre façons : la sensibilité spatiale, la sensibilité de donnée, la sensibilité de groupe et la sensibilité de contexte. Dans [18], le concept de sensibilité spatiale collective a été développé pour les systèmes d'information et de communication par le biais de trois entités : *Self, Others* et *Environment*. Le concept de Laboratoire Mobile de Nouvelle Génération peut être considéré comme étant la combinaison des concepts suscités.

Lorsque les véhicules connectés sont transformés en LM, ils pourront tirer parti de leur mobilité pour étendre le domaine d'observation d'un phénomène. La technologie pouvant gérer efficacement les données générées et en même temps orchestrer l'ensemble des services/applications en lien avec l'IoV sera sans doute le Cloud Computing [19], [20]. Le Sensing as a Service (S^2aaS), terme utilisé pour la première fois par Sheng et al. [21], est à l'origine conçu pour offrir des services liés à l'exploitation des capteurs incorporés dans les smartphones. Ce concept pourrait aussi être élargi pour pouvoir aussi gérer les données de capteurs automobiles.

L'usage de véhicules comme capteurs mobiles présente plusieurs avantages comme décrits dans [2] : très grande variété de capteurs, quasi-absence de contraintes d'énergie, possibilité d'embarquer des ordinateurs puissants, capacité de stockage élevée… En outre, la collecte de données s'effectue de façon continue et automatique. En effet, partout où il y a des routes, des véhicules y circulent et pourront y collecter des données. On se retrouve donc avec une diversité spatiotemporelle des données collectées. Comparé au déploiement de capteurs dédiés (fixes), l'usage de véhicules comme capteurs mobiles est beaucoup plus avantageux. En effet, les capteurs fixes nécessitent un lourd investissement et un long temps de déploiement. En outre, ces capteurs sont restreints à couvrir une petite zone d'observation, contrairement aux véhicules qui tireront parti de leur mobilité pour couvrir une plus grande zone. Un autre avantage est le bas coût d'acquisition des données. En effet, tout le dispositif nécessaire (capteurs, réseau interne, unité de traitement…) pour le bon fonctionnement du Laboratoire Mobile de Nouvelle Génération est déjà incorporé dans le véhicule.

Durant cinq ans (2011-2016), les activités de recherches et développement du Département de Transport des États-Unis (USDOT) se sont focalisées sur le prototypage et l'évaluation des applications relatives aux véhicules connectés [22]. L'évaluation se base essentiellement sur les impacts des applications sur la sureté, la mobilité et l'environnement. La *Figure 1* liste l'ensemble des applications sélectionnées et sponsorisées par l'USDOT. Dans [8], une étude de l'applicabilité de ces applications pour l'Afrique du Sud a été proposée. Les applications pour un développement à court terme sont priorisées. La plateforme pour le développement du C-ITS (Cooperative Intelligent Transport Systems) de l'Union Européenne (C-ITS Platform) a créé un groupe de travail (WG6) chargé d'examiner les moyens potentiels d'accès aux données et ressources des véhicules [23]. Le but est que les fournisseurs de services puissent proposer des services à leurs clients via l'exploitation de ces données.

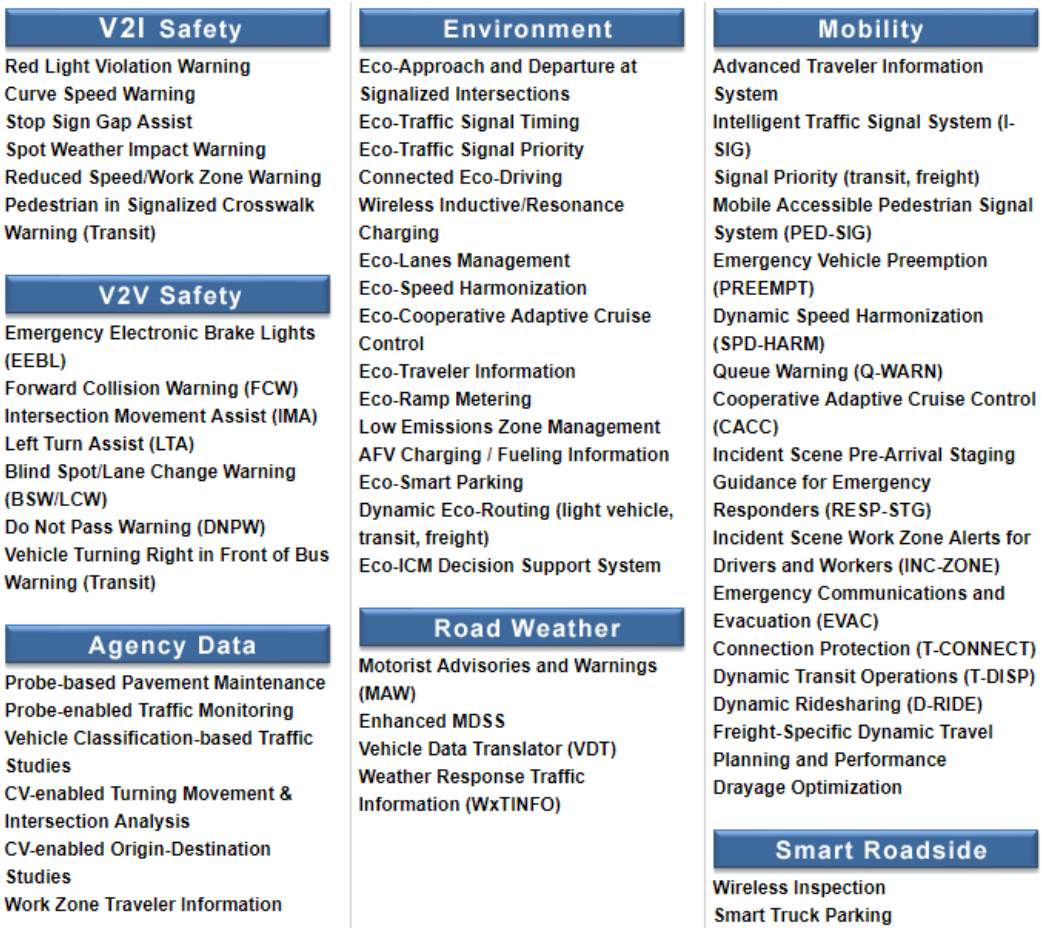


Figure 1 : Application des Véhicules Connectés [22]

Le concept de Laboratoires Mobiles de Nouvelle Génération peut être considéré comme une extension des applications suscitées, et pourra être utilisé pour, en plus, faire :

- *Monitoring de la Qualité de l'Air :* consiste à mesurer la concentration de polluants tels que les gaz oxydants (NO_2, O_3) et réducteurs (CO, hydrocarbures) qui ont un impact négatif sur le système respiratoire, et aussi les composés organiques volatils (CH_2O, C_7H_8) qui perturbent le système nerveux. Les particules fines telles que le PM_{10} et le $PM_{2.5}$ impactent aussi sur les systèmes respiratoires, visuels et cardio-vasculaires.
- *Agrométéorologie, production d'énergie solaire et tourisme :* la température, l'humidité de l'air, l'ensoleillement, la pluviométrie sont des paramètres très importants pour le développement de l'agriculture, de l'énergie renouvelable, et dans certaines mesures, du tourisme.
- *Sécurité nationale :* la fusion des données issues du radar, du lidar et des caméras permet de fournir un « œil artificiel » au véhicule. Cela peut servir à la lutte contre la criminalité comme expliquée dans [13] en renforçant les systèmes de vidéosurveillance déjà mise en place. En outre, la connaissance des conditions météorologiques (surtout la pluie et les tempêtes de sable) peut servir à mieux planifier et à mieux sélectionner l'itinéraire des véhicules de transport de produits dangereux (Hazmat, Hazardous Materials).

Le défi majeur à relever pour le développement des véhicules connectés, et plus particulièrement les Laboratoires Mobiles de Nouvelle Génération, réside sur la manière dont ces véhicules devront communiquer.

3. Problématique et challenges

Les premiers modèles de communication pour les véhicules ont été pensés à l'origine pour supporter des applications de sécurité routière. De ce fait, le VANET est proposé pour permettre aux véhicules de pouvoir communiquer directement entre eux sans passer par l'intermédiaire d'une infrastructure. Le VANET s'appuie essentiellement sur la technologie DSRC (Dedicated Short-Range Communications) et la norme 802.11p. Toutefois, l'adoption de la technologie DSRC a été retardée en raison de sa faible évolutivité et des difficultés de communication imposées par les environnements à forte mobilité et à topologie hautement dynamique [24], [25]. Notons également que le modèle de communication des Laboratoires Mobiles de Nouvelle Génération est un modèle orienté cloud, or les VANET ne peuvent pas fournir des services et applications à portées globales et durables. En effet, les entités impliquées sont temporaires, aléatoires, instables et à usage local [15]. En outre, les exigences des LM peuvent dépasser les limites de ce que les dispositifs et protocoles de communication actuels sont en mesure de fournir [26], [27].

Le déploiement à large échelle des Laboratoires Mobiles de Nouvelle Génération soulève quelques questions, à savoir :

Comment transformer un véhicule connecté en un Laboratoire Mobile de Nouvelle Génération ?

Quel(s) modèle(s) de communication faut-il proposer pour répondre aux besoins de communication des services/applications des LM ?

Quel(s) modèle(s) de communication faut-il proposer pour faciliter la transformation et l'intégration des données issues des capteurs automobiles en connaissances utiles ?

Quel(s) modèle(s) réseau faut-il proposer pour assurer un échange optimal des données entre les LM et les autres entités impliqués ?

La première entrave pour le développement des LM réside sur la nature très intégrée des technologies automobiles. Par exemple, le type exact de bus et les protocoles varient considérablement selon les fabricants, et même entre les différents modèles de véhicule de la même marque. Il est nécessaire de trouver un système générique capable d'interagir avec les différents sous-systèmes de capteurs et de communication.

Un modèle de communication décrit la manière dont les données sont échangées entre les différentes parties, exemple : peer-to-peer, client-serveur, publish-subscribe… Le choix d'un modèle de communication impactera sur les performances des applications, la facilité à effectuer différentes transactions, la fiabilité des services... Quant à un modèle réseau, il s'agit d'un ensemble de techniques de transmission de données via des réseaux de communication et les protocoles associés. Ainsi, les contraintes de la communication véhiculaire devront nécessairement être prises en compte dans la proposition de schémas de mobilité/handover, de techniques de gestion du flux...

Un autre défi à prendre en compte est l'étude de faisabilité des modèles via le prototypage. En effet, dans la littérature, la majorité des travaux se limitent à de nouvelles propositions théoriques et ne prennent pas en compte les problèmes d'implémentation. En fait, la faisabilité de la mise en œuvre des communications V2X et ses défis restent une question de recherche pertinente.

Dans cette thèse, nous nous proposons de répondre à l'enjeu de l'exploitation du potentiel informatique, de communication et de capture des véhicules pour des fins de Laboratoire Mobile de Nouvelle Génération. Ainsi, le premier objectif est de proposer des modèles de LM qui permettront d'exploiter de façon optimale les données de capteurs. Le but est d'étendre les services des véhicules connectés vers des applications liées à l'environnement, au climat, à l'agrométéorologie, à la santé, à la vidéosurveillance… Le second objectif est de proposer des modèles réseaux répondant aux exigences des modèles de communications qui seront proposés. Cela requiert la proposition d'algorithmes capables de fournir à l'ordinateur de bord et aux infrastructures de la route toute l'intelligence nécessaire pour prendre la meilleure décision quant à l'utilisation des ressources réseau et le maintien d'une

bonne connectivité. Cela pourrait se faire grâce un hyperviseur[1] de passerelles/interfaces sans fil. Un hyperviseur permet d'avoir une vue globale du réseau, une connaissance des modèles de communication employés, et enfin avoir un contrôle sur l'ensemble des interfaces de communication du véhicule.

4. Méthodologie

Afin d'évaluer les différentes architectures et modèles proposés, il est important de suivre une méthodologie adaptée. En général, la validation de concept suit un processus à deux étapes : analyse et vérification par simulation, et analyse et vérification par le prototypage [28]. Il faudra aussi tenir compte de certains aspects tels que le modèle de mobilité des véhicules, le modèle de communication et le modèle d'application. Les modèles de mobilité décrivent la dynamique des véhicules et sont très importants lorsque les tests se font en environnement de simulation.

Dans cette thèse, trois approches sont utilisées pour faire l'évaluation de l'ensemble des propositions. Il s'agit de l'évaluation analytique, de la simulation et du prototypage. L'évaluation analytique est utilisée pour principalement faire une étude comparative entre les solutions existantes dans la littérature et celles proposées. Ainsi, les schémas de mobilité/handover et les modèles de communication sont évalués analytiquement en amont. C'est à l'issue de cette évaluation analytique qu'une proposition est réévaluée via des simulations ou de prototypes. La validation de propositions liées à certains aspects de la communication véhiculaire nécessite d'effectuer des tests à grandes échelles qui sont souvent coûteux et nécessite la mobilisation d'une importante logistique. Raison pour laquelle la simulation est utilisée pour l'évaluation des schémas de routage V2V/V2I, l'échange de données impliquant plusieurs véhicules et/ou autres entités. Par contre, l'ensemble des schémas de mobilité/handover sont évalués à partir de prototypes. Il en est de même pour l'évaluation des performances réseau (latence, débit, perte de paquets…) des infrastructures d'expérimentation. Les modèle de services Cloud proposés seront évalués analytiquement et par simulation. Les implémentations en environnement réel ne pourront pas être réalisées en raison de problèmes de logistique.

5. Structure de la thèse et contributions

Cette thèse comprend la présente introduction, cinq chapitres et une conclusion. Les Chapitres 1 et 2 présentent les concepts théoriques et technologiques, et les Chapitres 3, 4 et 5 présentent nos différentes contributions.

Le Chapitre 1 décrit la structure d'un Laboratoire Mobile de Nouvelle Génération, les concepts et les technologies liés à la communication véhiculaire. Il est question, dans un premier temps, de montrer comment pourraient être exploitées

[1] hyper est un préfixe d'origine grecque signifiant au-dessus et indiquant une position supérieure.

les données d'une certaine catégorie de capteurs automobiles à des fins de Laboratoire Mobile. Ensuite, l'évolution du véhicule et des technologies, en allant du VANET jusqu'à l'IoV, est décrit dans ce chapitre.

Le Chapitre 2 est consacré à l'état de l'art des protocoles utilisés pour la gestion de la mobilité et du flux. Ainsi, les techniques de gestion du handover à travers WAVE (Wireless Access in Vehicular Environments), IPv6, HIP (Host Identity Protocol) et SDN (Software Defined Networking) y seront étudiées en détail. La gestion du flux à travers SDN est également étudiée dans ce chapitre.

Le Chapitre 3 constitue la première contribution dans cette thèse. Un modèle de Laboratoire Mobile est proposé et repose sur la transformation du véhicule en un nœud réseau multi-interfaces. Ensuite, une architecture de communication sécurisée et des schémas de mobilité à base du protocole HIP y sont proposés. Un hyperviseur à base du modèle multi-interfaces et HIP est implémenté. Des évaluations analytiques et des expérimentations sont effectuées afin de valider les différentes propositions faites dans ce chapitre.

Le Chapitre 4 consiste principalement à la proposition de modèle de Laboratoire Mobile basé sur le concept des véhicules connectés. Contrairement aux contributions précédentes, ce chapitre se base essentiellement sur les avancées de la technologie automobile, plus particulièrement des technologies de capteurs et d'accès réseau, pour proposer de nouvelles solutions de Laboratoire Mobiles. Ainsi, un modèle de Réseaux de Capteurs Véhiculaires Intelligents (IVSN, Intelligent Vehicular Sensor Network) est proposé en premier lieu. Puis, à partir du modèle IVSN, deux propositions de modèles de services cloud pour l'IoV, basé sur le modèle S^2aaS (Sensing as a Service), sont proposés. Ces modèles décrivent essentiellement les techniques de sélection sémantique de capteurs et des méthodes d'optimisation réseau. L'évaluation des performances réseau, pour la première proposition, s'est faite via un simulateur implémenté en Python. Pour la seconde proposition, une évaluation analytique du modèle est d'abord proposée, ensuite un exemple d'application S^2aaS nommée « Dynamic Best Air Quality Routing (DBAQ-Routing) » est implémenté via un simulateur en Python. Enfin, une extension d'Android Automotive optimisée pour le S^2aaS est proposée.

Le Chapitre 5 est le dernier chapitre de cette thèse et consiste à la proposition d'une architecture réseau adaptée aux modèles de communication proposées dans le chapitre 4. Ainsi, une architecture SDN intégrale pour la communication véhiculaire est proposée. Un prototype composé de switches SDN, de points d'accès WiFi SDN et de contrôleurs SDN est proposé comme banc d'expérimentation. Des techniques d'optimisation de flux et d'équilibrage de charge sont proposées au niveau du backbone. Pour le réseau d'accès, des schémas de mobilité réactifs et proactifs ont été proposés de même que des techniques d'équilibrage de charge. Une partie des fonctionnalités de l'ordinateur de bord sont implémentées sur un ordinateur monocarte. L'étude des performances de

l'architecture SDN ainsi proposée s'est réalisée à l'aide d'outils de mesures classiques tels que *iperf* et *ping*, mais également à l'aide d'applications SDN de benchmarking. Enfin, des schémas de routage V2V et V2I sont proposés puis implémentés via un simulateur en Python. Ce simulateur a permis l'évaluation des performances de ces schémas de routage.

Cette thèse se termine par une conclusion générale où un bilan de l'ensemble des travaux effectués durant notre thèse est proposé. Des perspectives ont également été proposées.

Chapitre 1 : Concept de Laboratoire Mobile de Nouvelle Génération et communication véhiculaire

Ce premier chapitre présente le concept de Laboratoire Mobile de Nouvelle Génération sous deux aspects, à savoir : capteurs et communication. Les véhicules modernes sont équipés d'une variété de capteurs dont l'exploitation se limite essentiellement à l'amélioration de l'efficacité énergétique de transport et au confort. La première partie de ce chapitre vise donc à montrer comment ces capteurs pourraient être utilisés afin de transformer un véhicule en un Laboratoire Mobile. La seconde partie du chapitre traite de l'aspect de la communication véhiculaire. Il sera question de décrire l'évolution de cette communication en allant du VANET jusqu'à l'Internet des Véhicules.

1.1. Structuration d'un Laboratoire Mobile de Nouvelle Génération

Les capteurs sont devenus des composants essentiels dans les systèmes de contrôle électronique des véhicules. Les constructeurs automobiles affichent un très grand intérêt sur leurs usages, car les performances des véhicules en dépendent fortement. Ainsi, plus le véhicule est haut de gamme, plus le nombre de capteurs qui y sont incorporés est élevé. On peut même retrouver jusqu'à plus d'une centaine de capteurs dans certaines classes de véhicules [1]. Dans [2], les auteurs exposent quelques avantages de l'usage des véhicules comme capteurs mobiles : très grande variété de capteurs, quasi-absence de contraintes d'énergie, possibilité d'embarquer des ordinateurs puissants, capacité de stockage élevée. En plus de la multiplicité des capteurs, ces véhicules pourront tirer parti de leur mobilité pour étendre le domaine d'observation d'un phénomène ; ces véhicules pourront ainsi être considérés comme des *Laboratoires Mobiles (LM) de Nouvelles Générations* comme illustrée par la *Figure 2*.

1.1.1. Variété de capteurs pour le LM

Avec la demande « full-option » des clients, on retrouve une très grande variété de capteurs dans les véhicules, chaque capteur étant optimisé pour une application bien particulière. Néanmoins, on peut regrouper ces capteurs suivant trois champs d'applications majeurs [1]–[3].

Figure 2 : Laboratoire Mobile de Nouvelle Génération

- *Groupe motopropulseur :* les capteurs contribuent à la fourniture de données essentielles à un ensemble d'éléments participant à la motricité du véhicule.
- *Sureté/sécurité :* les capteurs contribuent à la fiabilité, à la disponibilité, à la maintenabilité et à la sécurité du véhicule et de ses occupants.
- *Confort/commodité :* les capteurs sont utilisés pour fournir des services de commodités pour les passagers ; des services d'assistance et d'efficacité de conduite pour le conducteur.

En dehors de leurs applications premières, les capteurs automobiles peuvent être utilisés comme des dispositifs de laboratoire.

1.1.1.1. Mesure de grandeurs environnementales

Plusieurs capteurs sont spécialement incorporés dans le véhicule pour le monitoring de l'environnement immédiat du véhicule. Il existe également d'autres capteurs qui sont utilisés à d'autres fins, néanmoins on peut en extraire des informations liées à l'environnement.

- *Mesures directement liées à l'environnement*

Une partie des capteurs automobiles servent à mesurer la température ambiante, la pression atmosphérique, l'humidité de l'air, le taux d'oxygène dans les gaz brûlés… Ces mesures sont directement exploitables par les applications de monitoring de l'environnement.

- *HVAC (Heating Ventilation & Air Conditioning)*

Pour faire face à la pollution de l'air et aux changements climatiques dont sont exposés les conducteurs et les passagers, les constructeurs automobiles utilisent des capteurs pour commander le HVAC. Ainsi, pour réguler la température, certains capteurs fournissent au ECC (Electronic Climate Control) la

température en cabine, la température ambiante, l'humidité de l'air, la pression en cabine, l'ensoleillement… En dehors du HVAC, on peut exploiter ces mesures pour développer des applications pour l'agrométéorologie, la production d'énergie solaire ou le tourisme. L'humidité participe à renforcer la sensibilité des organes olfactifs à l'odeur, notamment les réactifs en poudre inorganiques [29]. Pour améliorer le confort des passagers, le système de ventilation régule l'humidité de la cabine en se basant sur les données des capteurs d'humidité. L'humidité de l'air est également un paramètre environnemental très exploité par les organismes de santé publique, d'où l'intérêt d'enrichir leurs sources de mesures.

- *Monitoring de la Qualité de l'Air (AQM)*

Il est utilisé par le système de ventilation du HVAC et le système d'alarme de gaz toxiques. Dans [30], les auteurs fournissent un ensemble de paramètres IAQ (In-cabin Air Quality) utilisés pour le monitoring de la qualité de l'air. Des polluants tels que les gaz oxydants comme le dioxyde d'azote (NO_2) et l'ozone (O_3), les gaz réducteurs comme le monoxyde de carbone (CO) et les hydrocarbures ont un impact négatif sur le système respiratoire. Tandis que les composés organiques volatils comme le méthanal (CH_2O) et le toluène (C_7H_8) perturbent le système nerveux. Les particules fines telles que le PM_{10} et le $PM_{2.5}$ impactent sur les systèmes respiratoires, visuels et cardio-vasculaires. Les conséquences pour le conducteur sont la nervosité, l'irritation de la peau et des yeux, la fatigue, le manque de concentration et pire, la somnolence. D'où l'utilisation du AQM par le HVAC pour réguler la qualité de l'air dans les cabines de véhicules. Dans [31], trois technologies de capteurs utilisées pour l'AQM sont présentées. Les premiers types de capteurs sont des capteurs de gaz à base de semiconducteur de métal oxyde (SMO) dont la conductibilité est proportionnelle à la concentration de gaz. Les capteurs SMO permettent de mesurer des gaz tels que le NO_2, O_3, CO, hydrocarbures... La seconde technologie est basée sur des capteurs électrochimiques (EC) dont le potentiel électrode est proportionnel à la concentration de gaz. Les capteurs EC permettent de déterminer essentiellement la concentration de gaz de la famille d'oxydes d'azotes (NO_x). La troisième technologie est à base de capteurs infrarouges (IR) caractérisés par leur grande sélectivité. Les gaz sont identifiés via le spectre d'absorption IR où chaque gaz présente une fréquence de résonnance moléculaire bien spécifique. Les capteurs IR peuvent détecter des gaz tels que le SO_2, le CO_2, le CO, le NO et les hydrocarbures. En dehors du HVAC, on pourrait collecter et analyser ces mesures relatives pour des raisons de santé publique.

- *Capteur de pluie*

Ce capteur est placé devant le rétroviseur intérieur du véhicule et mesure l'intensité d'un rayon infrarouge projeté sur le pare-brise. Les mesures effectuées par ce capteur sont utilisées par le contrôleur d'essuie-glace et de fermeture automatique des ouvertures. Lorsqu'il pleut, les gouttes de pluie sur le pare-brise absorbent une partie du rayon infrarouge. Cela permet de déduire l'intensité de la

pluie. D'autres capteurs sont basés sur la variation de la capacitance de deux plaques métalliques causée par la présence d'eau. On trouve aussi des capteurs basés sur des éléments de vibration piézoélectrique qui mesure la fréquence de percussion des gouttes d'eau sur le pare-brise. Les données recueillies à travers ces capteurs peuvent servir à délimiter exactement l'endroit et l'heure où il a plu. Dans une certaine mesure, on peut estimer la quantité de pluie tombée (pluviométrie) grâce à ces capteurs. D'où leur importance pour des applications météorologiques. Les données de pluviométrie peuvent être aussi intégrées dans des modèles d'estimation de risque et de gravité d'accident notamment lors du transport de produits dangereux (Hazmat,Hazardous Materials) [32].

- *Caméras et lidar (Light Detection and Recognition)*

La perception de l'environnement de conduite est très utilisée par les applications d'assistance de conduite telle que le régulateur de vitesse adaptatif. Les caméras et le lidar sont utilisés pour détecter les obstacles, les plaques de signalisation, les feux de trafic… Dans [33], les auteurs proposent de combiner le lidar et la caméra embarquée pour mesurer la pollution de l'air et la détection de poussière, de brume, de brouillard... En effet, le lidar utilisant une longueur d'onde de 900nm, est en mesure de détecter des polluants tels que l'aérosol, les particules nuageuses, le CO, le SO_2 et le CH_2. La fusion des données du lidar et de la caméra permet ainsi de fournir des informations relatives à la pollution de l'air et la qualité de vue avec une bonne fiabilité. Ces mesures pourront être utilisées pour enrichir et diversifier les sources de mesures de grandeurs environnementales.

- *Reconnaissance automatique de la voix (ASR)*

Les systèmes ASR sont devenus un standard pour les véhicules modernes et sont utilisés pour des applications de confort et de commodité tels que la commande vocale du système d'air conditionné, l'assistance à la navigation, la téléphonie.... L'ASR peut aussi être employées pour la commande des phares, des vitres et des rétroviseurs du véhicule [34]. En dehors des applications de commodité, le microphone du système ASR peut être exploité pour mesurer le niveau de pollution sonore et ainsi servir aux organismes de gestion du cadre de vie.

1.1.1.2. Les systèmes de perception : état trafic et vidéosurveillance

Les systèmes d'aide à la conduite ou ADAS (Advanced Driver Assistance Systems) se basent sur de nombreuses technologies, parmi lesquelles, les systèmes de perception. Ces systèmes utilisent la fusion des données issues du radar, du lidar et de la caméra pour fournir un « œil artificiel » au véhicule. Le conducteur bénéficie d'un niveau de sécurité et de confort accru comme le freinage d'urgence, l'aide au stationnement ou le système stop-and-go qui permet l'accélération et le freinage automatique du véhicule dans les zones d'embouteillage. L'exploitation des données issues des systèmes de perception permet également de connaitre le niveau de congestion des routes et peut servir aux applications traitant du trafic

routier. Les caméras des véhicules peuvent également servir à renforcer les systèmes de vidéosurveillance déjà mise en place dans une zone, et ainsi participer à lutte contre la criminalité comme expliquée dans [13].

1.1.2. Sémantique des capteurs

La transformation des véhicules comme Laboratoire Mobile engendrera la production d'un grand volume de données, de même qu'une grande hétérogénéité des sources de données, des formats de données et des procédures de mesures. En outre, la recherche, la réutilisation, l'intégration et l'interprétation de ces données requièrent, plus que, de justes valeurs d'observation. Il est donc nécessaire de résorber le déficit d'informations relatives à la disponibilité des capteurs et des propriétés associées aux mesures effectuées. Une des approches pourrait être d'enrichir la sémantique de ces données par le biais d'ontologies. Les ontologies fournissent des modèles formels, utilisables et extensibles qui sont adaptés pour représenter une information telles que les données de capteurs automobiles. Nous nous intéresserons particulièrement à l'usage de l'ontologie pour la description des capteurs et des propriétés de contexte associées.

1.1.2.1. Contexte et propriétés de contexte

Dans la littérature, il existe plusieurs définitions du terme « contexte ». Dans le cadre du Laboratoire Mobile, nous retiendrons qu'un contexte est un ensemble d'états d'entités ou d'informations décrivant un environnement dans lequel un évènement se produit, et favorisant sa compréhension. Pour le Laboratoire Mobile, un événement est essentiellement lié à une mesure et de son exploitation.

Dans [35], Perera et al. définissent le terme « propriété de contexte » d'un capteur comme étant un ensemble de caractéristiques (précision, sensibilité, plage de mesure, temps de réponse…) pouvant permettre d'évaluer la qualité de ce dernier. Plusieurs travaux de recherche [35]–[37] préconisent l'usage de l'ontologie SSN (Semantic Sensor Network) [38] pour modéliser les descriptions d'un capteur et de ses propriétés de contexte.

1.1.2.2. Modélisation des données de capteurs avec l'ontologie SSN

L'ontologie SSN (Semantic Sensor Network) a été créée par le groupe de travail W3C SSN Incubator Group [39]. Le SSN suit une architecture modulaire qui s'appuie sur une ontologie de base nommée SOSA (Sensor, Observation, Sample, and Actuator) pour représenter ses classes élémentaires et propriétés. L'avantage du SSN réside sur son interopérabilité et aussi sur la tendance actuelle de l'usage de l'ontologie dans l'IoT et dans la gestion des données de capteurs [35]. Le SSN est capable de modéliser un nombre important d'informations relatives aux capteurs telles que leurs aptitudes à pouvoir mesurer des grandeurs physiques, leurs performances ou encore les conditions dans lesquelles ils peuvent être utilisés. Le *Tableau 1* donne l'ensemble des aptitudes, plages de fonctionnement, plages de survie d'un capteur tel que défini par le groupe W3C [39].

Tableau 1 : Spécification des aptitudes d'un capteur

Classes et propriétés
ssn-system:inCondition, ssn-system:Condition, ssn-system:hasSystemCapability, ssn-system:SystemCapability, ssn-system:hasSystemProperty, ssn-system:SystemProperty, ssn-system:MeasurementRange, ssn-system:ActuationRange, ssn-system:Accuracy, ssn-system:DetectionLimit, ssn-system:Drift, ssn-system:Frequency, ssn-system:Latency, ssn-system:Precision, ssn-system:Resolution, ssn-system:ResponseTime, ssn-system:Selectivity, ssn-system:Sensitivity, ssn-system:hasOperatingRange, ssn-system:OperatingRange, ssn-system:hasOperatingProperty, ssn-system:OperatingProperty, ssn-system:MaintenanceSchedule, ssn-system:OperatingPowerRange, ssn-system:hasSurvivalRange, ssn-system:SurvivalRange, ssn-system:hasSurvivalProperty, ssn-system:SurvivalProperty ssn-system:SystemLifetime, ssn-system:BatteryLifetime ssn-system:qualityOfObservation

Cette spécification facilite la mise en place des techniques de recherche, de classement et de sélection de capteurs sur la base de la sémantique associée à ces derniers [35], [36]. La *Figure 3* fournit un exemple de modèle d'un capteur du point de vue de ses propriétés de contexte.

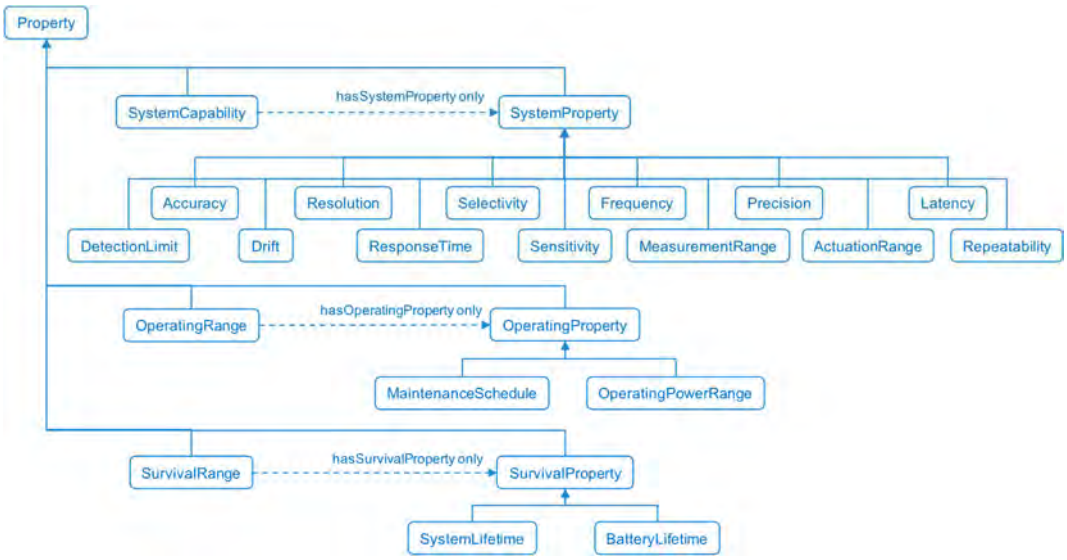


Figure 3 : Modèle de capteur basé sur le SSN [39]

1.1.3. Réseau intravéhiculaire

Grâce l'intégration de systèmes informatiques dans les véhicules, les données de capteurs sont transmises via un réseau intravéhiculaire. Ainsi, les véhicules modernes emploient généralement un nombre important de ECU (Electronic Control Units) où sont reliés un ensemble distribué et disparate de capteurs et d'actionneurs. Les ECU sont responsables de l'échantillonnage et du traitement des données issues de ces capteurs, et aussi du transfert des commandes vers les actionneurs. Ces informations sont partagées via un réseau filaire complexe et multiplexé comme le montre la *Figure 4*.

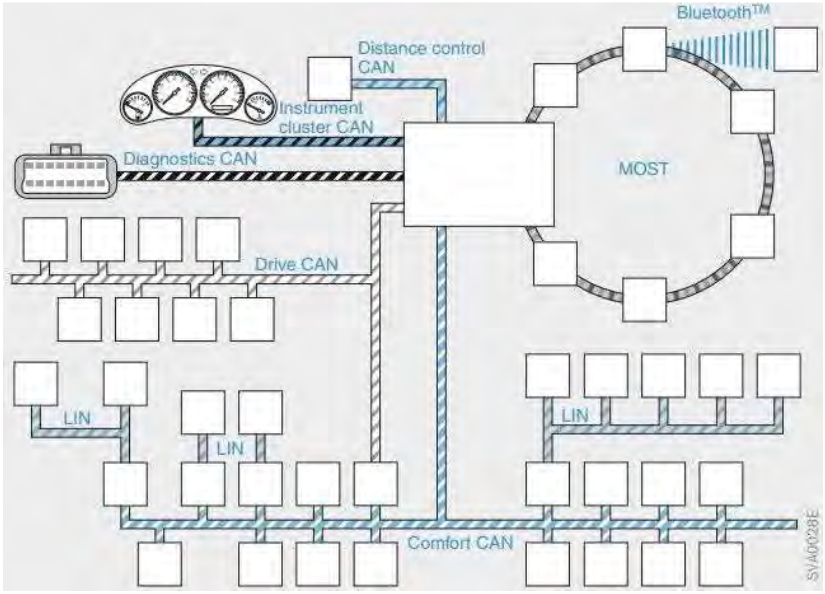


Figure 4 : Topologie réseau intravéhiculaire [3]

1.1.3.1. Communication à base de bus

Pour interconnecter plusieurs ECU, la communication à base de bus est le plus couramment utilisée. Le bus est un système de communication pouvant logiquement connecter plusieurs périphériques via un même ensemble de fils grâce à un système de multiplexage. Les systèmes à base de bus sont connus pour leurs facilités de conception et sont caractérisés par leurs faibles poids, longueurs de câblage et coûts de déploiement. Cependant, comme les nœuds utilisent la même ligne de communication, il est nécessaire de définir des mécanismes d'évitement de collisions ou définir un système maitre/esclave pour gérer l'accès au bus. Le *Tableau 2* regroupe les technologies les plus couramment utilisées [40].

Tableau 2 : Technologies à base de bus pour la communication intravéhiculaire

	CAN	**LIN**	**FlexRay**	**MOST**	**J1850**
Application	Légèrement temps réel, gestion de la priorité	Bas coût, bas débit	Orientée temps réel	Multimédia	Diagnostique
Débit	500 kbps	19,6 kbps	10Mbps	24,8 Mbps	41,6 kbps
Contrôle	Multi-master	Mono-master	Multi-master	Timing-master	Multi-master
Protocole d'accès	CSMA/CA	Polling	TDMA	TDM/CSMA	CSMA/NDA
Support physique	Paire torsadée	Mono filaire	Paire torsadée, fibre optique	Fibre optique	Paire torsadée

1.1.3.2. Ethernet pour l'automobile

L'utilisation d'Ethernet dans le domaine de l'automobile est motivée par son haut débit comparé aux technologies CAN et MOST. Les premières applications de l'Ethernet dans l'industrie automobile étaient dédiées pour le diagnostic sur IP et le flashage des ECU. Puis, la seconde génération d'applications est orientée divertissement (audio/vidéo) et ADAS par l'utilisation de caméras via Ethernet. Enfin, la troisième génération d'applications envisagées vise à changer la topologie réseau intavéhiculaire notamment en mettant en place un backbone Ethernet comme le montre la *Figure 5*.

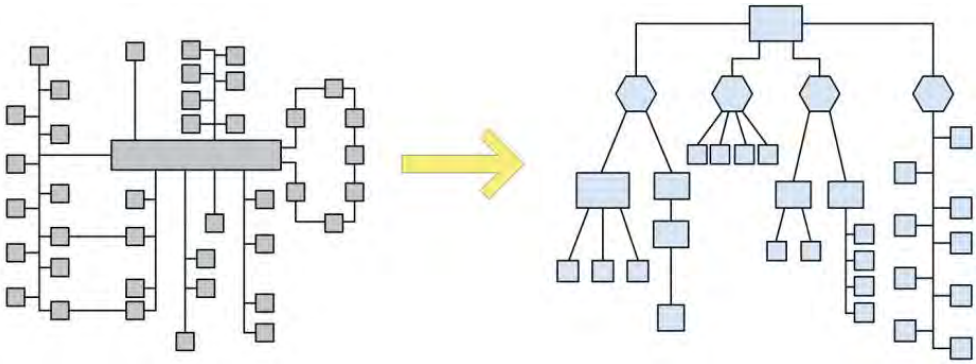


Figure 5 : Migration d'une architecture hétérogène vers le full Ethernet

Pour répondre aux exigences de la communication intravéhiculaire, de nouvelles spécifications et révisions de spécifications ont été effectuées par les groupes IEEE 802.3 et 802.1. On peut citer : IEEE 802.3 pour le diagnostic et le flashage, PoE (Power-over-Ethernet) et EEE (Energy-Efficient Ethernet), IEEE 802.1 Audio Vide Bridging (AVB), TTEthernet (Time-Triggered Ethernet).

1.2. Introduction à la communication véhiculaire

Les avancés dans le domaine de la communication sans fil ont rendues possibles la communication en temps réel entre les véhicules et les infrastructures au bord de la route. Des efforts de standardisation pour la communication véhiculaire sont en cours afin de rendre le transport sûr, écologique et facile.

Les premières applications des réseaux véhiculaires sont majoritairement liées à la sécurité routière. L'objectif principal étant de réduire les accidents grâce à un système d'échange et de diffusion de messages d'alertes. Plus récemment, de nouvelles applications relatives à l'IoV sont de plus en plus proposées. Ces dernières suscitent une grande attractivité économique auprès des opérateurs privés.

1.2.1. Caractéristiques de communication

La communication véhiculaire présente des caractéristiques spécifiques qui la différencie des autres types de communication sans fil. On peut en citer quelques-unes de ces caractéristiques en se basant sur les travaux de Al-Sultan et al. [41] et Sharef et al. [42] :

- *Forte mobilité :* les véhicules sont connus pour être des nœuds pouvant se déplacer à de très grande vitesse.
- *Topologie hautement dynamique :* la topologie des VANET change rapidement du fait de la variabilité de la vitesse des véhicules, du changement fréquent de direction et du comportement des conducteurs. Ce phénomène impacte négativement le routage et l'accès aux services.
- *Densité réseau variable :* la densité réseau dépend fortement de la densité du trafic routier. Dans les milieux urbains, une très forte densité réseau est remarquée. En contraste, une faible densité réseau est notée sur les routes très peu fréquentées (zones rurales) ou durant certaines heures.
- *Déconnexions fréquente :* c'est une conséquence directe de la forte mobilité, de la topologie dynamique et parfois de la faible densité réseau.
- *Mobilité prédictible :* la mobilité des véhicules est principalement déterministe du fait des restrictions géographiques et physiques (dimension des routes, intersections, feux de signalisation…).
- *Absence de contraintes d'énergie et de capacité de calcul :* les véhicules ont de faibles contraintes en énergie, par conséquent, plusieurs capteurs/actionneurs, des ressources informatiques importantes, des technologies d'accès radio performants... sont incorporés en leur sein.
- *Fortes contraintes en délai :* pour certaines applications comme celles de sureté et de sécurité, le délai de transmission est un facteur critique et ne doit pas dépasser un certain seuil (200 ms).

1.2.2. Modèle de mobilité

Dans la littérature, il existe de nombreux modèles de mobilité. Néanmoins, ces modèles peuvent être regroupés. Ainsi, Harri et al. [43] les classifient en quatre différentes classes :

- *Modèles synthétiques :* regroupe l'ensemble des modèles purement mathématiques.
- *Modèles basés sur la simulation de trafic :* le modèle est extrait à partir d'un simulateur de trafic.
- *Modèles basés sur l'investigation :* inclus des données statistiques sur le comportement journalier des citadins.
- *Modèles basés sur les traces :* génère des modèles à partir de données de trafic réelles.

1.2.2.1. Facteurs d'influences de la mobilité

La mobilité des véhicules est affectée par plusieurs facteurs clefs identifiés à travers des travaux de recherche. Ces facteurs peuvent être classés ainsi :

- *Cartes topologiques précises et réalistes :* le mouvement des véhicules dépend fortement de la caractéristique des routes : sens et nombre de voies, feux et signaux de trafic, présence d'intersections...
- *Longueur des tronçons :* le nombre d'intersections dépend de la longueur des tronçons. Cette longueur permet d'estimer la fréquence des accélérations et décélérations.
- *Comportement des conducteurs :* le facteur humain (sexe, âge, catégorie sociale, humeur…) influe également sur les modèles de conduite. Ainsi les dépassements de véhicules, le changement subit d'itinéraire, la prudence de conduite sont des comportements liés à la personnalité du conducteur.
- *Points d'attraction et de répulsion :* souvent, la destination finale des conducteurs demeure identique, comme quitter tous les matins leurs domiciles (zone résidentielle, banlieue) pour aller à leur lieu de travail (zone industrielle, centre-ville). Les lieux de manifestation sont souvent des points de répulsion à l'heure de leur déroulement.

1.2.2.2. Méthodes de prédiction de trajectoire

La détermination de la topologie se fait généralement par le biais d'algorithmes basés sur des modèles de mobilité microscopiques (basés sur le mouvement individuel de chaque véhicule et ses interactions avec les véhicules tout autour) et macroscopiques (basés sur les contraintes de déplacement tels que la topologie des routes, le flux de véhicules, les panneaux et feux de circulation).

La prédiction de trajectoire peut se faire en s'inspirant de modèles de mobilité réalistes tel que celui proposé par Harri et al. [43], voir *Figure 6*.

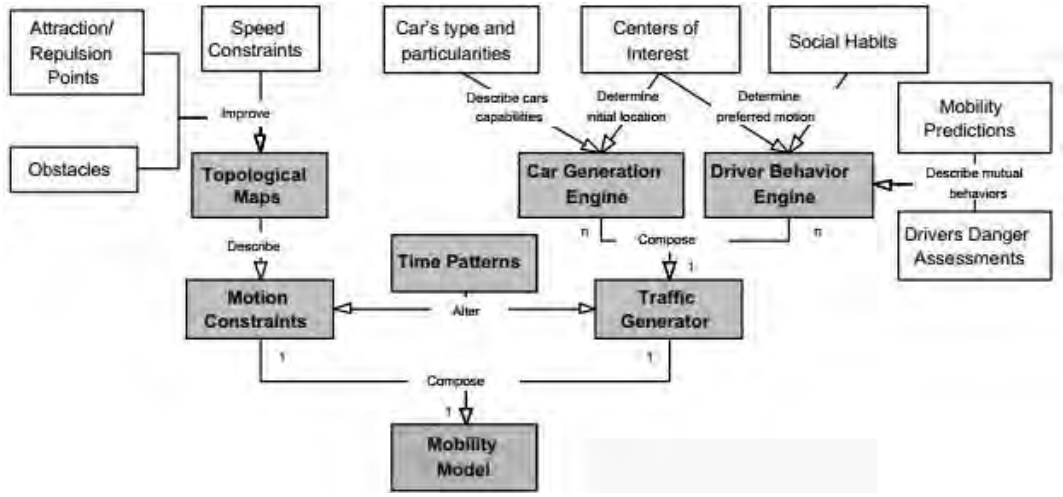


Figure 6 : Framework réaliste de modèle de mobilité [43]

Des méthodes analytiques peuvent également être utilisées pour une modélisation analytique du trafic [44], voir la *Figure 7*.

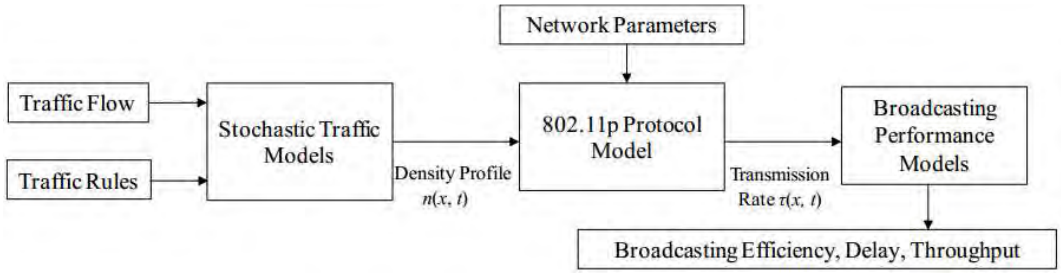


Figure 7 : Framework de modélisation des performances du 802.11p [44]

1.3. Vehicular Ad Hoc Network

Le VANET (Vehicular Ad Hoc Network), fruit de l'application des MANET (Mobile Ad Hoc Network) pour les véhicules, est le socle de la communication véhiculaire. Le concept de base des VANET est de permettre aux véhicules de pouvoir communiquer directement entre eux sans passer par l'intermédiaire d'une infrastructure. Les échanges se font donc en mode ad hoc. Cependant, les véhicules peuvent également échanger avec des nœuds fixes situés au bord de la route pour bénéficier de services supplémentaires. La recherche dans les VANET se focalise essentiellement sur le routage, la diffusion de message, la qualité de service et la sécurité.

1.3.1. Architecture des VANET

L'architecture des VANET peut être décrite à partir des interactions entre ses différents composants et des types de communication supportés. Cette architecture est conçue pour favoriser l'utilisation d'applications coopératives.

1.3.1.1. Les principaux composants

La mise en place des applications pour la communication véhiculaire se fait sur une architecture basée sur trois entités, à savoir :

- *Roadside Unit (RSU)*

C'est un dispositif habituellement situé au bord de la route ou sur des lieux spécifiques tels que les intersections et les parkings. La RSU utilise une technologie sans fil dédiée à la communication véhiculaire et d'autres technologies standards pour l'accès à des réseaux externes. L'emplacement de la RSU est choisi de telle sorte qu'il préserve la couverture et la connectivité d'un maximum de véhicules. La RSU permet d'étendre le champ de communication des réseaux ad hoc via une redistribution des messages ou un transfert des messages vers d'autres RSU. Des applications de sureté routière sont également implémentées au niveau des RSU. Enfin la RSU peut fournir dans certains cas un accès Internet aux véhicules.

- *On Board Units (OBU)*

C'est un équipement embarqué dans le véhicule et permet la communication de ce dernier avec la RSU et d'autres OBU. En outre, la OBU fournit un service de

communication aux Application Units (AU) et relaie des messages pour le compte d'autres OBU. Les principales fonctionnalités de la OBU sont entre autres, la fourniture d'un accès réseau sans fil, le routage, le contrôle de la congestion, la gestion de la mobilité, la collecte et le traitement de données.

- *Application Units (AU)*

Ce sont des entités exploitant les applications à bord du véhicule grâce à la capacité de communication de l'OBU. Ces applications sont principalement liées à la sécurité. Notons que les AU et l'OBU sont souvent intégrés sur le même équipement physique.

1.3.1.2. Architecture de communication

L'architecture de communication des VANET, voir *Figure 8*, est axée sur le véhicule et l'ensemble des communications entre le véhicule et les autres entités. Dans les VANET, on retrouve principalement trois types de communication :

Figure 8 : Architecture réseau des VANET

- *Vehicle-to-Vehicle (V2V) communication*

L'échange de message se fait via un réseau ad hoc ; les véhicules constituant les nœuds de ce réseau. En d'autres termes, les messages sont échangés directement via les différents OBU embarqués dans les véhicules. Le broadcast est utilisé de manière prédominante dans ce type de communication notamment pour la diffusion de messages d'alerte, la dissémination de données, la conduite collaborative…

- *Vehicle-to-Infrastructure (V2I) communication*

Toute la communication se fait en mode infrastructure. Dans cette configuration, les véhicules, via leurs OBU, constituent les stations clientes et la RSU la station de base ou point d'accès. En plus du broadcast pour la diffusion d'informations (météo, assistance à la conduite, place de parkings…), l'unicast est également utilisé pour l'agrégation de données, l'accès à Internet...

1.3.2. Protocoles de routage

La nature très dynamique des topologies, la très grande mobilité des véhicules dans les VANET et les caractéristiques d'un environnement urbain posent un véritable challenge pour la conception de protocoles de routage efficaces [42], [45], [46]. Le but premier de ces protocoles est de minimiser les pertes de paquets, la latence, la charge réseau tout en cherchant à maximiser le débit. Plusieurs scénarios de communications peuvent être définis en se basant sur la façon dont les paquets sont envoyés de la source vers un ou plusieurs destinataires. On distingue ainsi la communication unicast, multicast/geocast et broadcast. Les protocoles de routage V2V se focalisent uniquement sur les véhicules sans tenir compte des infrastructures au bord de la route. Dans la littérature [42], [46], ces protocoles sont classés en plusieurs groupes :

- *Routage basé sur la topologie*

Ce routage utilise les informations des liens existant dans le réseau afin de transmettre un paquet de la source vers le destinataire. Pour le routage proactif, les informations de tous les nœuds sont stockées sous forme de table où est maintenue une liste des destinations et des routes associées. Le routage proactif ne provoque pas un délai lié à la découverte initiale de route, mais néanmoins consomme une bande passante significative lors de la mise à jour périodique des topologies. En outre, une quantité de mémoire et une capacité de calcul considérable est utilisée par chaque véhicule pour le maintien des tables de routage trop volumineuses. Ainsi, le routage proactif n'est pas très adapté pour la communication V2V. Pour le routage réactif, les routes sont établies à la demande en inondant le réseau par des paquets de découverte de route. Ce type de routage règle les problèmes liés à la consommation de la bande passante, en contrepartie, on note également une certaine lenteur de réaction lors de la restructuration des routes et des échecs. Néanmoins, ce protocole est bien adapté pour les VANET.

- *Routage basé sur la position (géographique)*

Pour ce type de routage, tous les nœuds sont capables de connaitre leurs propres localisations et celles de leurs voisins grâce à l'utilisation du GPS ou à l'échange périodique de messages de balisage (beacon). Ainsi le routage de messages se fait directement sans une connaissance de la topologie et sans une découverte préalable de route.

- *Routage basé sur le clustering*

Pour ce type de routage, les véhicules voisins sont organisés sous forme de clusters. Chaque cluster ayant un cluster head responsable de la gestion de la communication intra-cluster et inter-cluster. Dans les VANET, la configuration des clusters et la sélection/élection de leurs clusters heads constitue le principal défis. En effet, la grande mobilité des véhicules fait qu'on se retrouve avec des clusters dynamiques et un changement récurant de leurs clusters heads.

1.3.3. Classification des applications des VANET

Plusieurs consortiums, organismes de standardisation, projets et gouvernements se sont mis au développement d'applications pour les VANET, chacune ayant ses propres exigences en termes de performance et de QoS. En se basant sur les travaux de Papadimitratos et al. [5], quelques exemples d'applications des VANET peuvent être cités, voir *Tableau 3.*

Tableau 3 : Liste des applications du VANET [5]

	Nom de l'application	**Caractéristiques**				
		Communi-cation	Type de message	Période	Latence	Autres exigences
1	Emergency Electronic Brake Lights	V2V	Evènementiel, broadcast à temps limité	100 ms	100 ms	Portée 300 m, haute priorité
2	Slow Vehicle Warning	V2V	Broadcast permanent périodique	500 ms	100 ms	Haute priorité
3	Intersection Collision Warning	V2V, V2I	Broadcast permanent périodique	100 ms	100 ms	Position précise sur la carte, haute priorité
4	Hazardous Location Warning	V2V, V2I	Evènementiel, GeoCast à temps limité	100 ms	100 ms	Haute priorité
5	Traffic Signal Violation Warning	V2I	Evènementiel, broadcast à temps limité	100 ms	100 ms	Portée 250 m, haute priorité
6	Pre-Crash Sensing	V2V	Broadcast périodique, unicast	100 ms	50 ms	Porté 50 m, haute/moyenne priorité pour le balisage/unicast
7	Lane Change Warning	V2V	Broadcast périodique	100 ms	100 ms	Précision de la position relative < 2 m, portée 150 m
8	Cooperative Forward Collision Warning	V2V	Broadcast périodique, broadcast événementiel	100 ms	100 ms	Précision de la position relative < 1 m, portée 150 m
9	Intersection Management	V2V, V2I	Broadcast périodique, unicast	1000 ms	500 ms	Précision de la position < 1 m
10	Limited Access and Detour Warning	V2I, autres réseaux de broadcast	Broadcast périodique	100 ms	500 ms	Moyenne/faible priorité
11	Cooperative Adaptive Cruise Control	V2V	Broadcast, unicast	500 ms	100 ms	Moyenne priorité
12	Electronic Toll Collect	V2I, cellulaire	Broadcast périodique, unicast	1000 ms	200 ms	CEN DSRC
13	Remote Diagnosis/ JIT Repair Warning	V2V, V2I, cellulaire	Broadcast, unicast, événementiel	N/A	500 ms	Accès Internet, accessibilité du service
14	Media Download	Cellulaire, autres réseaux de broadcast	Broadcast, unicast, à la demande	N/A	500 ms	Accès Internet, accessibilité du service
15	Map Download/Update	V2V, V2I, cellulaire, autres réseaux de broadcast	Broadcast, unicast, à la demande	1000 ms	500 ms	Accès Internet, accessibilité du service
16	Ecological Drive Assistance	V2V, V2I, cellulaire	Broadcast, unicast, à la demande	1000 ms	500 ms	Accès Internet, accessibilité du service

Les huit premières lignes constituent des applications de sécurité routière basées sur une communication ad hoc en mode broadcast exigeant une haute priorité et un faible délai. Les quatre lignes suivantes (9−12) présentent des applications d'efficacité du trafic et d'assistance à la conduite. Les échanges se font également en mode ad hoc mais avec des exigences moindres. Par ailleurs, les modes unicast par V2V ou via par réseau cellulaire sont utilisés. Les quatre dernières lignes (13−16) sont des applications de divertissement et de confort. La communication est de type infrastructure avec un accès à Internet.

1.3.4. Standards de communication

Plusieurs consortiums, organismes de recherche et de standardisation tels que IEEE (Institute of Electrical and Electronics Engineers), ASTM (American Society for Testing and Materials), ETSI (European Telecommunications Standards Institute), ISO (International Organization for Standardization), C2C-CC (CAR-to-CAR Communication Consortium), IETF (Internet Engineering Task Force) se sont mis à l'élaboration de standards de communications fiables pour la communication véhiculaire. Même si leurs conceptions sont fortement influencées par des exigences régionales, ces standards visent à assurer l'interopérabilité et l'implémentation rapide de nouvelles technologies radio [6]. Ainsi, l'IEEE développe la norme 802.11p et la pile protocolaire WAVE. L'ISO développe le standard CALM basé sur des réseaux hétérogènes. Le C2C-CC n'expérimente que des protocoles liés aux ITS. L'IETF travaille sur des extensions IP pour les VANET. Enfin, l'ETSI travaille sur l'harmonisation des standards suscités. Dans ce qui suit, nous n'étudierons que les standards DSRC/802.11p/WAVE.

- *DSRC (Dedicated Short-Range Communications)*

Le DSRC utilise le concept de communication multicanal afin d'assurer les communications pour les applications de sécurité et les autres services de l'ITS. Ainsi, aux États-Unis, le FCC (Federal Communication Commission) a attribué la bande 5,9 GHz de largeur 75MHz constitués de 7 canaux pour le DSRC. De même en Europe, l'ETSI et la CEPT (Conférence Européenne des administrations des Postes et Télécommunications) ont attribuées aussi la bande 5.9 GHz de largeur 20 MHz constituée de 4 canaux. Enfin, au Japon, l'ARIB (Association of Radio Industries and Business) a attribué la bande 5.8 GHz de largeur 80 MHz avec 14 canaux. Les détails des caractéristiques du DSRC sont donnés dans le *Tableau 4*.

Les 7 canaux DSRC relatifs à l'ASTM, voir *Figure 9*, ont respectivement le numéro 172, 174, 176, 178, 180, 182 et 184. Le canal 178 est le canal de contrôle CCH (Control CHannel) et les six autres sont les canaux de service SCH (Service CHannels). Le canal 172 est un canal spécial réservé pour fournir une haute disponibilité et une faible latence. De même, le canal 184 est caractérisé par sa forte puissance d'émission et son utilisation pour la sureté publique. L'ETSI par contre utilise seulement les canaux SCH 172, 174, 176, 178 et le canal CCH 180.

Tableau 4 : Caractéristiques du standard DSRC

	États-Unis (ASTM)	Europe (ETSI)	Japon (ARIB)
Communication	Half-duplex (OBU) Full-duplex (RSU)	Half-duplex	Full-duplex
Fréquence	5,9 GHz	5,9 GHz	5,8 GHz
Bande	75 MHz	30 MHz	80 MHz
Canaux	7	5	7 downlink/7 uplink
Largeur canaux	10 Mhz	5 MHz	5 MHz
Débit transmission	3–27 Mbps	500 kbps uplink 250 kbps downlink	1–4 Mbps
Modulation	OFDM	2-ASK (RSU) 2-PSK (OBU)	2-ASK, 4-PSK
Porté	1000 m	15–20 m	30 m

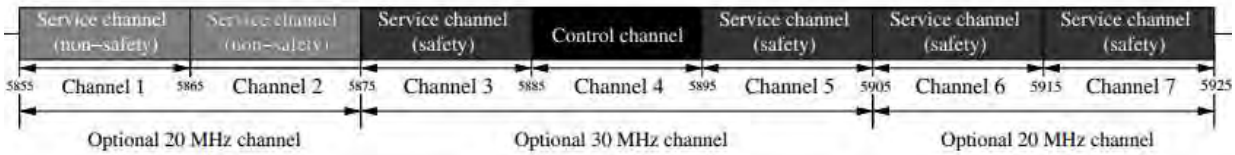


Figure 9 : Disposition des canaux DSRC

- *La norme 802.11p*

La norme 802.11p, évoluant sur la bande DSRC des 5,9 GHz, redéfinit les couches PHY et MAC de la norme 802.11a/e pour les adapter conformément aux caractéristiques et contraintes de la communication véhiculaire. En fonction du schéma de codage de la couche PHY, le débit de transmission peut atteindre des valeurs comprises entre 3 et 27 Mbps pour une portée de 300 à 1000m. La couche MAC est une adaptation de la norme 802.11e utilisant un mécanisme de priorisation d'accès au canal basé sur le EDCA (Enhanced Distributed Channel Access).

- *WAVE (Wireless Access in Vehicular Environments)*

WAVE est un ensemble de piles protocolaires de la famille IEEE 1609 et définit une architecture, un ensemble de services et d'interfaces permettant une communication sécurisée dans les VANET. La famille IEEE 1609 est constituée de standards dont l'intégration dans le modèle OSI est décrite sur la *Figure 10*.

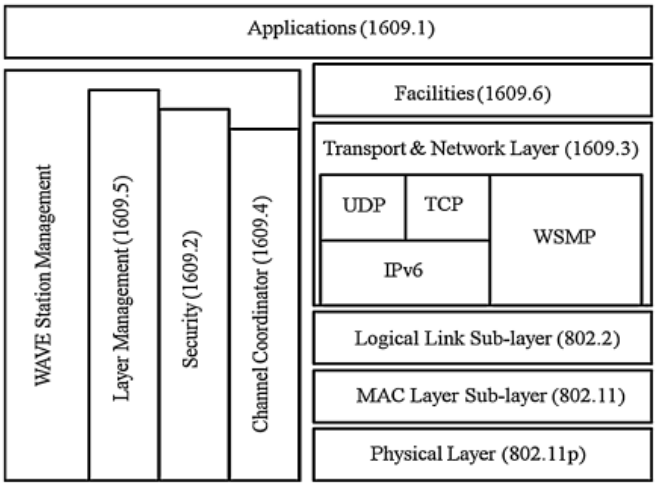


Figure 10 : Architecture WAVE

1.4. Réseaux de Capteurs Véhiculaires

Dans les véhicules modernes, on peut retrouver jusqu'à des centaines de capteurs servant à recueillir de manière continue des informations liées à l'état des pièces mécaniques et électroniques, à la perception, à l'environnement…

1.4.1. Concept

Dans leurs travaux, Lee et al. [12] proposent d'étudier l'usage des véhicules comme des nœuds capteurs pour la collecte d'informations en milieu urbain ; ils ont nommé ce système « Vehicular Sensor Network (VSN) ». Le VSN est donc un nouveau paradigme réseau émergent héritant du concept des VANET et des WSN. Cela confère, par conséquent, au VSN des propriétés uniques comme le changement dynamique des zones d'intérêts, la collecte de données issues du véhicule lui-même (monitoring de l'état du véhicule, du conducteur…) et de son environnement immédiat (niveau de pollution, état du trafic routier, caméra de surveillance…). Enfin les contraintes en termes de puissance de calcul, de capacité de stockage et d'énergie sont relaxées. La recherche sur les VSN se focalise sur les techniques de dissémination, de collecte et d'agrégation de données issues des capteurs embarqués dans le véhicule.

1.4.2. Méthodes de dissémination et d'agrégation de données

Différentes approches pour la dissémination et l'agrégation de données issues des VSN sont proposées dans la littérature [47], [48].

1.4.2.1. Dissémination de données

La dissémination de données est un mécanisme permettant de délivrer des données vers des destinataires ciblés de façon fiable, en un court délai avec une utilisation minimale de ressources. Dans [47], les méthodes de dissémination de données ont été classifiées comme suit :

- *Dissémination opportuniste*

Les messages sont stockés sur des nœuds intermédiaires et transférés à chaque nœud rencontré à l'improviste jusqu'à ce que le nœud destinataire reçoit le message. Cette technique améliore le taux d'acheminement des paquets, mais avec une latence beaucoup plus élevée.

- *Dissémination basée sur le cluster*

Les messages sont envoyés au cluster head qui est chargé de les relayer au cluster head suivant jusqu'à atteindre la destination finale. Cette technique permet d'augmenter le taux d'acheminement des paquets et réduit la charge réseau, car peu de nœuds intermédiaires sont utilisés.

- *Dissémination pair-à-pair*

Les messages sont envoyés à la demande, sinon ils demeurent stockés au niveau du nœud. Cette technique permet de limiter la charge réseau; cependant elle induit un délai de réponse.

- *Dissémination géographique*

Les messages sont envoyés en basant sur la connaissance géographique des nœuds. L'une des techniques les plus utilisées est le geocasting.

1.4.2.2. Agrégation de données

L'agrégation de données est toute forme de dissémination où les données ont été modifiées et traitées par les nœuds intermédiaires en utilisant des techniques de compression, recodage, fusion, ou en mettant à jour et/ou supprimant des informations non valides. Dans les VSN, le but recherché est surtout de réduire l'utilisation de la bande passante et d'améliorer la scalabilité, c'est-à-dire réduire la charge réseau et la taille des paquets, cependant la qualité des données doit être préservée après agrégation. Dans [48], les auteurs classifient les méthodes d'agrégation comme suit :

- *L'agrégation syntaxique*

Cette technique consiste à concaténer les données et de les envoyer en un paquet unique. Ainsi des techniques de compression et recodage sont utilisées afin de contenir toutes les données issues de plusieurs véhicules dans une seule trame. Cela permet par conséquent de réduire considérablement la charge réseau.

- *Agrégation sémantique*

Cette technique consiste à représenter l'information sous une forme sommaire. Ainsi, des informations redondantes sont fusionnées, tandis que les informations sont supprimées lorsqu'elles sont dupliquées, similaires ou expirées.

- *Agrégation hiérarchique*

Cette technique est surtout employée pour les véhicules structurés sous forme d'arbre où il y a existence d'une relation de hiérarchie. Ainsi l'agrégation se fait à base de cluster, et vise surtout à réduire les délais et les redondances tout en préservant la qualité des données.

1.4.3. Applications

Conçus pour la collecte d'informations en milieu urbain, les domaines d'application des VSN ne cessent de croitre du fait qu'ils sont particulièrement adaptés pour le monitoring d'événements à caractère spatio-temporel et granulaire.

L'un des travaux les plus cités et parmi les plus pertinents dans ce domaine est sans doute celui de Lee et al. [13]. Le système proposé, nommé MobEyes, est conçu pour le monitoring proactif en milieu urbain où chaque véhicule est chargé de collecter, de stocker puis de transmettre les données via un protocole opportuniste dans un environnement distribué. MobEyes se focalise sur l'amélioration de la scalabilité et de la réduction de la charge réseau.

Hu et al. [49] proposent une architecture VSN de monitoring du microclimat basé sur le GSM et le GPS. Un ajustement adaptatif de la quantité de données envoyées permet de trouver un équilibre entre la qualité de monitoring et le coût de la communication.

Salhi et al. [50] proposent un protocole de collecte de données basé sur des clusters géographiques où les *cluster head* sont chargés de collecter et de transmettre les données jusqu'à la RSU le plus proche.

Yu et al. [51] se penchent sur l'acquisition de données coopératives et sur l'utilisation d'un algorithme de compression. L'approche proposée permet d'obtenir un meilleur taux de données reconstituées comparées aux approches conventionnelles.

D'autres travaux plus récents, comme celui de He et al. [52], proposent une nouvelle technique de collecte de données basée sur le ciblage de zones densément peuplées par des véhicules. Leur algorithme permet de collecter une grande quantité de données en un temps record.

Du et al. [53] s'intéressent au monitoring du trafic urbain par l'utilisation de véhicules sondes (taxis, bus, voiture de patrouille). Du fait de l'irrégularité de la distribution des véhicules sondes, les auteurs s'appuient sur la technique de matrice de reconstitution afin d'estimer l'état du trafic sur les artères non sondées. Leur technique permet de considérablement réduire les erreurs d'estimation.

Enfin, Huang et al. [54] proposent une technique de collecte de données basée sur l'utilisation d'agents mobiles qui migrent de manière opportuniste parmi les véhicules d'un petit segment routier. Cette technique permet d'atteindre un ratio élevé de données collectées avec une faible charge réseau.

1.5. Internet des Véhicules

Selon les récentes prédictions, environ entre 25 et 50 milliards d'objets seront connectés à Internet d'ici 2020. Les véhicules y occuperont une large portion. En effet, selon Gartner [9], d'ici 2020 un véhicule sur cinq sera un véhicule connecté soit près d'un quart de milliard de véhicules ; par conséquent l'IoV occupera une place prédominante dans l'IoT. Cet optimisme est réitéré dans un autre article de presse intitulé « Forget the Internet of Things : Here Comes the 'Internet of Cars' » [10], où T. Koslowski soutient que pour l'industrie automobile, connecter les véhicules est désormais un critère prédominant dans la phase d'innovation.

1.5.1. Concept

L'IoV est un nouveau concept émergeant s'appuyant sur l'IoT pour faire évoluer le véhicule d'un simple nœud dans les VANET en une plateforme intelligente capable d'apprendre, de penser et de comprendre les systèmes cyberphysiques par lui-même [14]–[16]. L'IoV se focalise ainsi sur l'intégration des humains, des véhicules, des objets et des environnements pour former un système mobile et dynamique de communication. Ce système sera capable de collecter, partager, traiter et comprendre les informations relatives à l'ITS.

L'architecture de l'IoV s'appuie sur les VANET, la télématique et l'Internet mobile. Le véhicule est ainsi imaginé comme un modèle multi-communicant, appelé Vehicle-to-Everything (V2X) communication. L'IoV s'appuie également sur des techniques et concepts informatiques modernes tels que l'intelligence en essaim, le crown-sensing, le cloud computing… Ce dernier est d'ailleurs l'élément moteur de l'IoV, car étant responsable de l'orchestration de l'ensemble des technologies constituant l'écosystème de l'IoV.

1.5.2. Véhicule connecté

Les progrès technologiques dans les domaines des MEMS (Micro-Electro-Mechanical Systems), de la communication et du traitement de l'information ont contribués à la création de nouveaux types de véhicules. Ces derniers, appelés véhicules connectés, sont dotés de capacités de communication, de stockage et d'apprentissage.

1.5.2.1. Véhicule intelligent

Dans le contexte d'un véhicule, voir *Figure 11*, le terme « intelligent » se réfère à l'incorporation d'un certain niveau d'intelligence artificielle dans le fonctionnement du véhicule [55]. Cette intelligence permet de rendre autonome le véhicule dans certaines tâches tout en sollicitant l'intervention du conducteur dans d'autres. Un véhicule intelligent doit être ainsi doté d'une capacité de calcul (CPU), de communication (V2X), de positionnement (GPS), de perception (capteurs) et d'interactions (actionneurs).

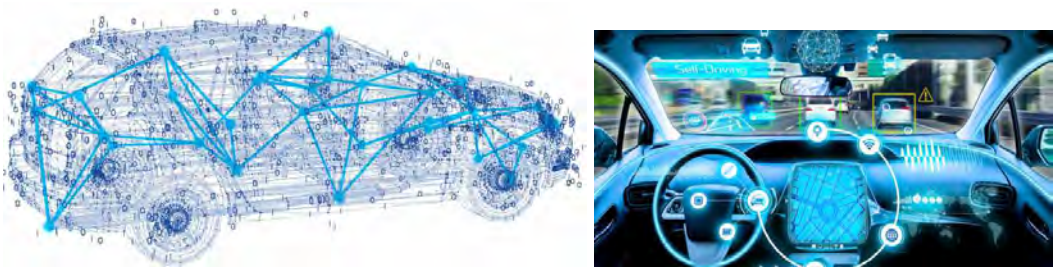
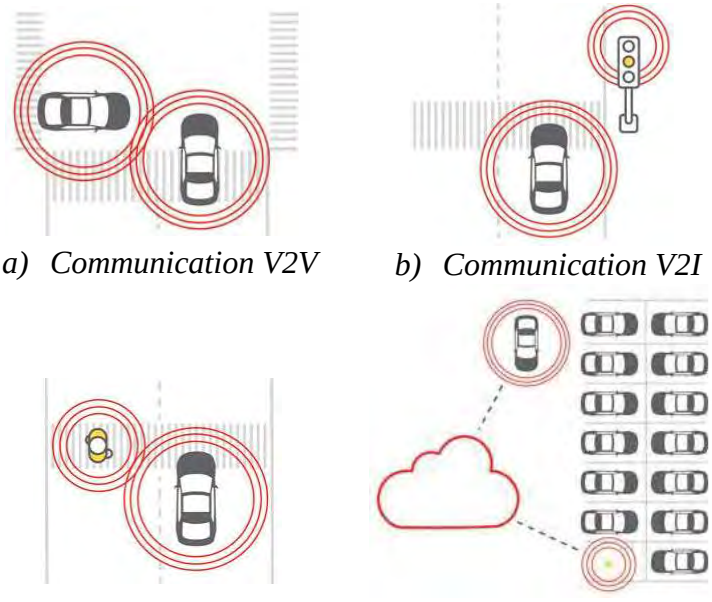


Figure 11 : Véhicule imaginé comme une plateforme numérique intelligente

L'intelligence d'un véhicule signifie aussi sa capacité à inférer de nouvelles connaissances. Ce véhicule est alors amené à exploiter ses capacités de perceptions, via ses propres capteurs et ceux de son environnement immédiat, afin de collecter des données contextuelles. Ensuite, le véhicule usera de ses capacités de modélisation et de raisonnement sur les données de contexte afin d'extraire des informations contextuelles de hautes niveaux appelées connaissances. Cela permet ainsi au véhicule mieux communiquer et interagir avec son environnement extérieur.

1.5.2.2. Communication V2X

Un véhicule connecté se réfère à la connectivité sans fil permettant au véhicule de pouvoir communiquer avec son environnement interne et externe, c'est-à-dire supportant la communication V2X [56], [57]. C'est donc un véhicule qui incorpore plusieurs de types de communications, *Figure 12*, telles que :



a) Communication V2V *b) Communication V2I*

c) Communication V2P *d) Communication V2C*

Figure 12 : Communication V2X

- *V2V (Vehicle-to-Vehicle)*

Héritée du VANET, ce type de communication permet aux véhicules d'échanger directement des messages relatifs à la sécurité routière en mode ad hoc.

- *V2I (Vehicle-to-Infrastructure)*

Héritée également du VANET, et permet aux véhicules d'échanger avec une infrastructure au bord de la route en mode ad hoc. En plus des messages de sécurité routière, des messages liés au trafic routier peuvent être également échangés.

- *V2P (Vehicle-to-Pedestrian)*

C'est une communication entre le véhicule et le piéton via son smartphone. Les messages échangés sont essentiellement des messages d'alertes anticollisions.

- *V2C (Vehicle-to-Cloud)*

C'est une communication entre le véhicule et le Cloud. Le véhicule pourra ainsi bénéficier de services cloud dédiés tels que la recherche de place de parking, le télédiagnostic, l'assistance à la conduite, la météo, le divertissement…

1.5.2.3. Principaux défis

Les véhicules connectés sont des véhicules qui accèdent, consomment, créent, enrichissent, dirigent et partagent l'information entre les individus, les organismes, les milieux d'affaires, les infrastructures et les objets. Comme ces véhicules deviennent de plus en plus connectés, ils ont donc une « conscience de soi », sont sensibles aux contextes et deviennent éventuellement autonomes [10]. La technologie des véhicules connectés [57] se concentre sur le développement et le déploiement d'une plateforme permettant la croissance, l'expansion, l'intégration de nouvelles technologies, le développement d'applications non encore déployées, les interactions complexes avec l'humain et l'environnement. Ces technologies, définissent également de nouveaux défis tels que des standards pour l'interopérabilité, développent de nouveaux schémas pour les systèmes de sécurité, identifient et traitent les défis technologiques tels que le positionnement et la scalabilité. Les défis liés à la connectivité doivent également être traités à travers le clustering des véhicules en mouvement afin d'améliorer la connectivité, la stabilité des clusters et l'efficacité des schémas de handover.

1.5.3. IoV et Cloud Computing

Le nombre important de véhicules connectés engendrera inévitablement l'acquisition et le traitement d'un grand volume de données. Pour gérer cette quantité de données, et en même temps, orchestrer l'ensemble des services/applications en lien avec l'IoV, le Cloud Computing est considéré dans la littérature comme l'approche la plus adaptée.

1.5.3.1. Modèles Cloud pour l'IoV

Parmi les couches constituant le modèle réseau de l'IoV, les problèmes liés à l'élaboration de la couche cloud constituent le plus grand challenge de recherche du fait que l'IoV est un paradigme exclusivement orienté cloud [58]–[60]. La vision ubiquitaire de l'IoV fait qu'un modèle de communication à portée locale ne peut convenir à tous les services/applications. En outre, un modèle de communication à portée globale est parfois inutile, onéreux, non optimal du fait de l'intérêt/pertinence locale d'un service. Un compromis serait alors d'utiliser un modèle où la portée locale ou globale de chaque information est déterminée avant son traitement et sa diffusion. La communication dans l'IoV se fait en pair-à-pair ou en mode client-serveur en s'appuyant sur un conglomérat complexe et hétérogène d'entités sans fil. Dans la littérature [19], [58], trois architectures cloud, désignées sous différents noms, sont principalement utilisées :

- *Cloud-based Architecture, Vehicles using Clouds ou Vehicles to Clouds*

Est basé sur le cloud conventionnel, c'est-à-dire les services/applications sont fournis par des infrastructures distantes accessibles via Internet.

- *Vehicle-based architecture, Vehicular Clouds ou Vehicles as Clouds*

Les véhicules constituent en même temps la couche cliente et la couche cloud. L'accès aux services/applications se fait via la communication V2V ou indirectement via la communication V2I.

- *Hybrid architecture ou Hybrid Vehicular Clouds ou Vehicles with Cloud*

C'est la combinaison des deux modèles suscités. Cette architecture est motivée par la nécessité de maintenir l'accès aux services/applications tout au long du trajet tout en bénéficiant des informations à intérêt/pertinence local.

1.5.3.2. Sensing as a Service : S^2aaS

Le Sensing as a Service (S^2aaS), terme introduit par Sheng et al. [21], se présente aujourd'hui comme une solution permettant de transformer les données brutes issues de ces capteurs en informations de hautes niveaux (connaissance) dont il est impossible d'acquérir par les méthodes traditionnelles. Le S^2aaS va tirer parti de la mobilité des véhicules pour suivre la vision 4A : *Anywhere, Anytime, by Anyone and Anything*. L'intégration du S^2aaS dans l'IoV sera d'un intérêt économique sans précédent. En effet, tous les acteurs évoluant dans le secteur de l'automobile verront leurs services mieux être rentabilisés. Par exemple, une société d'hydrocarbure pourra à partir du S^2aaS mieux évaluer l'efficacité de ses produits, connaitre les habitudes et les préférences des conducteurs, et par conséquent, améliorer sa campagne marketing. De même, une étude environnementale ou climatique à large échelle pourra être menée avec un faible coût d'investissement grâce au S^2aaS. Dans les ITS, le S^2aaS contribuera à la réduction des bouchons et accidents, une meilleure assistance à la conduite, une meilleure gestion des places

de parkings, de la flotte logistique et du transport public. Dans la littérature, on peut retrouver plusieurs travaux relatifs au S^2aaS. Dans [21], [61] les auteurs posent les différents challenges tout en faisant une taxonomie des solutions existantes.

Sheng et al. [21] proposent un modèle exploitant les différents capteurs incorporés dans les smartphones pour fournir un service de perception géré par plusieurs serveurs dans le cloud. Les auteurs insistent sur trois points : la capacité à pouvoir supporter plusieurs plateformes mobiles, la capacité à pouvoir inciter les individus à partager leurs données et la capacité d'économiser l'énergie.

Dans leur revue, Kantarci et al. [61] classent les schémas S^2aaS suivant trois catégories : « service provider search », « GPS-less sensing/scheduling » et « service provider recruitment ».

Perera et al. [62] proposent un modèle S^2aaS découpé en quatre couches : « sensors and sensor owners », « sensor publishers (SPs) », « extended service providers (ESPs) », et « sensor data consumers ». Quelques exemples d'utilisation de leur modèle sont présentés notamment pour la gestion des ordures, le smart agriculture et la gestion environnementale.

Perera et al. [35] proposent un modèle de recherche, de sélection et de classement de capteurs nommé CASSARAM. La sensibilité de contexte est utilisée dans la mise en place de leur système via la combinaison de techniques de raisonnement quantitatif et la recherche sémantique afin d'améliorer les performances du système.

Une amélioration du modèle CASSARAM est proposée par Hsu et al. [36]. Ces derniers se basent sur une architecture distribuée permettant l'évolutivité et la réduction du trafics réseau causés par les changements fréquents de propriétés de contexte des capteurs.

Alam et al. [63] proposent un framework virtuel de gestion des capteurs en s'appuyant sur le paradigme e-SOA (event-driven Service Oriented Architecture). Une interface facilitant l'interrogation des objets connectés et leurs services associés est proposée, de même qu'un mécanisme de contrôle d'accès sémantique.

Dans la même lancée, Abdelwahab et al. [64] proposent des algorithmes de découverte de ressources et de virtualisation. Leur modèle permet de déployer un réseau de capteurs virtuels correspondant à la tâche de capture.

1.5.3.3. Du Cloud Computing au Fog Computing

Avec l'IoT, des milliards d'objets connectés vont communiquer entre eux et vont évoluer dans un tissu de réseaux hétérogènes. Ces objets sont caractérisés par une forte mobilité et requièrent souvent une analyse et un traitement à temps réel des données générées. Le modèle de réseau du Cloud Computing étant centralisé, les ressources se situent donc dans le cœur du réseau, c'est-à-dire éloignées des objets et des terminaux. Par conséquent, le Cloud Computing ne peut couvrir toutes

les exigences relatives à l'IoT telles que la réduction de la latence, la géodistribution des données, la forte mobilité des objets, les systèmes distribués à large échelle… En plus, les objets connectés produisant un grand volume de données non structurées, alors, leur traitement et analyse dans le cœur du réseau risque de surcharger considérablement toute la chaine de communication.

Le Fog Computing, terme introduit par Bonomi et al. [65], est défini comme une extension du Cloud Computing dans laquelle les ressources sont déplacées du cœur vers la bordure du réseau, c'est-à-dire près des objets et terminaux. Cette proximité avec les objets offre d'intéressantes caractéristiques au Fog Computing comme une faible latence, une interaction à temps réel, une sensibilité spatiale, une distribution géographique, un déploiement de réseaux de capteurs à grande échelle, une amélioration de l'hétérogénéité/interopérabilité, un support à la mobilité… Les nœuds Fog sont hétérogènes par nature. Ainsi les nœuds concernés sont les serveurs haut de gamme, les routeurs de bordure, les points d'accès et les terminaux (véhicules, capteurs, smartphones…) [66]. Les domaines d'applications du Fog Computing tournent autour des véhicules connectés, gestion de données médicales, Smart Grid, Smart Cities, des réseaux de capteurs sans fils et autres.

Kai et al. [67] présentent l'état de la recherche sur l'intégration du Fog Computing dans les VANET. La discussion est principalement axée sur les applications, les challenges et les futures tendances de cette intégration.

Zhang et al. [68] discutent également sur les défis et les opportunités de l'intégration du fog computing dans l'IoV. Ces auteurs proposent une architecture fog coopérative pour l'IoV, voir *Figure 13*. Ces derniers montrent l'importance de la mise en place d'une architecture coopérative afin d'optimiser les performances de la communication IoV.

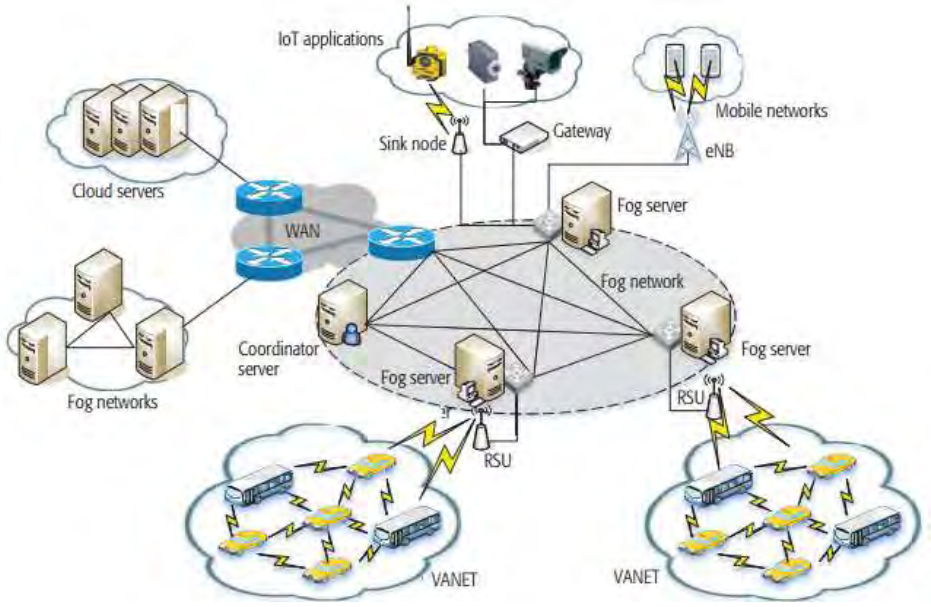


Figure 13 : Architecture fog computing pour l'IoV [68]

Il existe de nettes différences le cloud computing et le fog computing, et on peut se demander laquelle choisir. Cependant, le fog et le cloud se complètent, en les combinant, les services fournis peuvent être encore mieux optimisés.

1.6. Conclusion

La présentation du concept de Laboratoire Mobile de Nouvelle Génération nous a permis de montrer une partie du potentiel applicatif des véhicules connectés. La sémantique des données de capteurs par le biais d'ontologie telle que le SSN apparait comme une option crédible pour permettre la mise en œuvre des mécanismes de raisonnement. Il en ressort également que les modèles cloud hybrides (Hybrid Vehicular Clouds) semblent être les plus adaptés pour les Laboratoires Mobiles de Nouvelles Générations. En effet, certains services des LM auront une portée et pertinence locale, tandis que d'autres devront nécessairement avoir une portée globale par le biais d'Internet et du cloud conventionnel. Pour permettre un bon accès et une bonne fourniture de ces services cloud, il est clair que cela ne peut pas se faire sans une grande robustesse des architectures de communication, notamment la gestion de la mobilité et du flux.

Dans le prochain chapitre, nous nous intéresserons aux techniques utilisées pour la gestion de la mobilité et du flux dans le cadre de la communication véhiculaire.

Chapitre 2 : Gestion de la mobilité et du flux dans le contexte des réseaux véhiculaires

Dans ce chapitre nous présentons l'état de l'art des différentes approches de gestion de la mobilité. Après avoir défini le concept de la mobilité dans un environnement véhiculaire, nous montrerons ensuite les techniques de gestion handover pour les véhicules, basées sur les protocoles 802.11p, IPv6/NEMO et HIP. La dernière partie sera réservée à l'étude de la technologie SDN et de ses applications dans la communication véhiculaire notamment le routage et l'optimisation du flux.

2.1. La mobilité dans le contexte des réseaux véhiculaires

Les réseaux sans fil tendent vers des systèmes de communication totalement persuasives et ubiquitaires qui permettront à terme d'assurer une connectivité *« Any where, Any time, Any place »*. La mobilité peut être définie comme étant une situation où un nœud se déplace tout en gardant ses contextes de communications actives ; et tout cela, de manière transparente. Dans les réseaux véhiculaires, deux types de communications sont principalement utilisées : V2V et V2I. Dans [69], les auteurs proposent une taxonomie des solutions de gestion de la mobilité pour les réseaux véhiculaires, voir *Figure 14*.

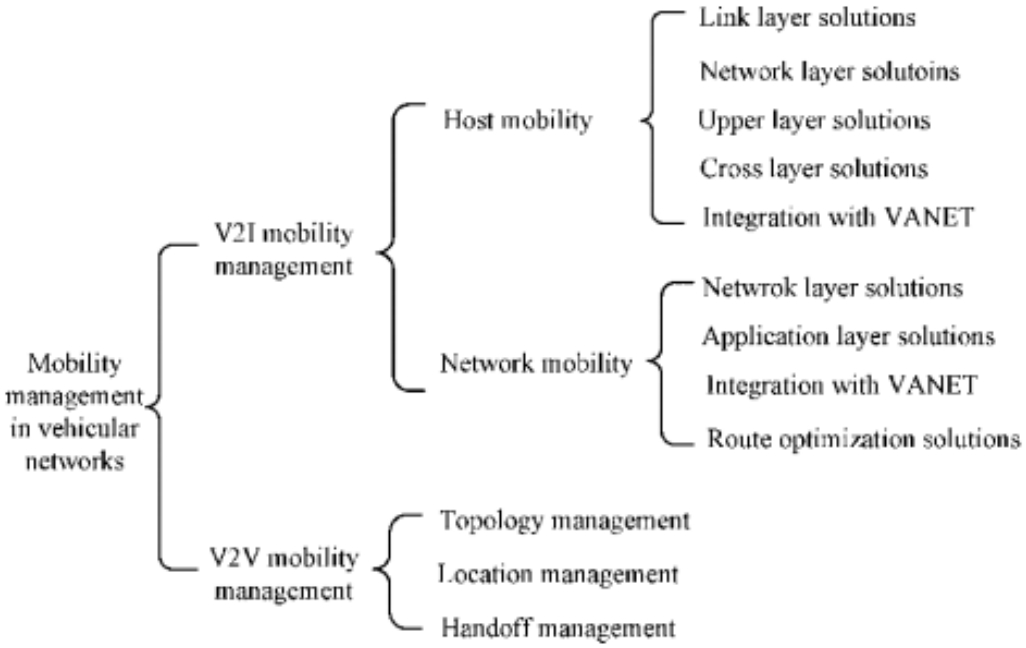


Figure 14 : Taxonomies des solutions de gestion de la mobilité des VANET [69]

2.1.1. Mobilité et communication V2I

La communication V2I permet aux véhicules d'accéder à un certain nombre de services proposés par les infrastructures au bord de la route. La communication peut se faire en multisaut. Dans les réseaux VAN (Vehicular Access Network), l'OBU joue également le rôle de passerelle pour les dispositifs internes du véhicule (réseau de capteurs et certaines applications ITS) et les dispositifs sans fils des passagers (Smartphones, tablettes, ordinateurs portables). Dans ce scénario, le véhicule est considéré comme un routeur mobile dont les fonctionnalités sont définies par le protocole NEMO BS (Network Mobility Basic Support) [69], [70]. Parmi toutes les exigences de la communication V2I, figurent celles des schémas de mobilité, à savoir :

- *Un handover rapide (fast handover) :* les applications VANET sensibles aux délais exigent un handover de faible latence.
- *Communication multisaut :* les VANET utilisent la communication multisaut pour étendre la portée de la transmission.
- *Scalabilité :* les VANET étant un réseau très large et en plus les véhicules étant caractérisés par une forte mobilité, on note alors une fréquence de handover très élevée.

2.1.2. Mobilité et communication V2V

La communication V2V permet aux véhicules de communiquer entre eux en mode ad hoc. Les schémas de mobilité devront nécessairement répondre à certaines exigences telles que le changement dynamique de la topologie, la connaissance de la localisation des véhicules et le maintien de l'efficacité des protocoles de routages. Ainsi, le modèle de cluster est le plus souvent utilisé pour gérer la mobilité d'un groupe de véhicules. Dans ce cas, on considère un Handover lorsqu'un véhicule quitte un cluster pour entrer dans un autre.

2.2. Schémas de handover basés sur le standard 802.11p

Le concept de handover dans 802.11p diffère à plusieurs niveaux de celui des autres standards de la famille IEEE 802.11. Dans le standard 802.11p, la procédure de handover est de niveau 2 et demeure très basique. En effet, lorsque le véhicule entre dans la zone de couverture de plusieurs RSU, il reçoit toutes les annonces WSA (WAVE Service Advertisement) diffusées par ces derniers et choisit d'accéder à l'un de ces services. Lorsque le véhicule accède consécutivement à moins deux services fournis par deux RSU distinctes, on parle alors de handover. Le mécanisme de scan sur plusieurs canaux n'est pas requis dans le sens où dans WAVE, toutes les annonces sont diffusées via un unique canal de contrôle CCH sur la bande DSRC. Cela réduit considérablement la complexité pour la découverte d'une nouvelle RSU. Par ailleurs, aucun processus d'authentification ou de réassociation n'est requis dans WAVE.

La simplification du mécanisme de handover dans WAVE soulève plusieurs problèmes liés à la latence et aux pertes de paquets comme indiqués dans [71], [72]. Le premier problème réside sur la phase d'écoute des messages WSA qui peut prendre un certain temps lorsqu'il y a des collisions sur le canal CCH. Le second problème survient lorsque l'OBU reçoit plusieurs messages WSA. En effet, ce processus peut générer une certaine latence lorsque le nombre de critères de sélection du service RSU devient important. En troisième lieu, des problèmes liés aux pertes de paquets surviennent lorsque l'ancienne RSU ne parvient plus à joindre l'OBU pour lui délivrer ses paquets mis en mémoire tampon. Enfin, étant donné que les RSU opèrent sur les mêmes fréquences, on note une redondance de données. En effet, une même trame peut être reçue ou envoyée par plusieurs RSU comme, par exemple, répondre à une même requête d'un OBU.

2.2.1. Revue des schémas de handover pour le standard 802.11p

Dans la littérature, plusieurs approches ont été proposées afin de pallier aux problèmes suscités.

Bohm et al. [73] proposent un schéma de handover déterministe basé sur un processus rapide de connexion et sur un handoff proactif. Leur schéma nécessite que les OBU soient identifiées via leurs adresses MAC. Et, en fonction de la position, de la vitesse et de la direction du véhicule, la RSU courante informe la prochaine RSU de l'approche du véhicule. Cette prochaine RSU pourra ainsi proactivement connaitre les besoins des futures OBU (véhicules) qui entreront dans sa zone de couverture. Les résultats proposés donnent seulement une indication sur le nombre total de véhicules qui pourront bénéficier du service de handover, cependant aucune garanti n'est fournie sur la possibilité de réduction du délai.

Choi et al [74], quant à eux, se basent sur l'identification de la prochaine RSU par les OBU afin de les informer d'un handover imminent. A la réception d'un message de dissociation, la RSU courante redirige tout le trafic vers la prochaine RSU. Celle-ci délivre les données, mises en mémoire tampon, aux OBU dès qu'elles entrent dans sa zone de couverture. Ce schéma permet d'éviter l'envoi redondant d'une même trame, mais ne résout pas les problèmes d'interférences. Des questions sur la gestion de la mémoire tampon peuvent être soulevées.

Chung et al. [75], proposent un nouveau schéma de priorisation d'accès au canal qui permet aux OBU étant sur le point de quitter la RSU d'avoir une priorité d'accès beaucoup plus élevée que le reste des OBU. Les avantages de leur schéma de priorisation sont exploités dans leur schéma de handover afin de réduire la latence et la congestion du canal de contrôle CCH.

Contrairement aux travaux suscités qui ne se focalisent que sur la réduction de la latence et des pertes de paquets, Cho et al. [76], proposent une approche basée sur l'utilisation, par les RSU, de plusieurs canaux différents pour envoyer les trames WSA. A la réception des trames WSA, la OBU classifie les RSU en fonction du

RSSI et de l'importance des informations incluses dans chaque trame. Après classement, la meilleure RSU est choisie. Des tests de performance ont été effectués en testant l'utilisation de plusieurs applications. Les résultats confirment une réduction de la latence du handover, cependant le processus de classement et de sélection peut être dans certaines situations (complexité de calcul) un facteur dégradant pour les performances de leur schéma.

2.2.2. Discussion sur les schémas de handover 802.11p/WAVE

En se basant sur les travaux de recherche sur la gestion du handover dans le standard 802.11p/WAVE, on constante qu'une certaine frange d'applications peut fonctionner avec les propositions déjà faites. L'accès à Internet des OBU via le standard 802.11p natif demeure un véritable défi, surtout que ces dernières n'ont pas une identité dans le réseau WAVE. Si l'on souhaite offrir un accès Internet aux véhicules, il est nécessaire d'adapter le modèle de communication du standard 802.11p/WAVE vers celui du TCP/IP.

2.3. Schémas de mobilité basés sur IP pour les réseaux véhiculaires

Plusieurs travaux de recherche visent donc à améliorer le standard WAVE. Dans le récent draft du groupe de travail IEEE Network Working Group [70], on peut retrouver une étude complète et détaillée des réseaux véhiculaires basés sur IP.

2.3.1. Le protocole IPv6 dans WAVE

Le protocole IPv6 ne se limite pas seulement à la définition d'un format de trame au niveau de la couche réseau, mais intègre également un certain nombre d'hypothèses liées à la définition d'un modèle de lien, d'adressage et d'opérations réseau (ND : Neighbor Discovery, DAD : Duplicate Address Detection). Baccelli et al. [77] ont fourni dans leurs travaux une analyse complète de l'intégration de IPv6 dans WAVE. L'autoconfiguration d'adresse IP est l'une des questions principales traitées dans les travaux d'adressage réseau pour les véhicules.

Fazio et al. [78] proposent un modèle de cluster où chaque cluster head est assimilé à un serveur DHCP. En changeant de cluster, le véhicule se voit assigner une nouvelle adresse par le biais de son nouveau cluster head. Ce schéma offre l'avantage de réduire la fréquence de changement d'adresse IP au prix d'une charge réseau non négligeable pour s'assurer de l'unicité des adresses IP assignées.

Dans les travaux de Kato et al. [79], l'adresse IP est générée à partir d'un préfixe réseau associé à chaque voie de l'autoroute et de l'adresse MAC de l'interface réseau du véhicule. Cependant ce schéma présente une charge réseau assez élevé lorsque le tronçon présente plusieurs voies et que les véhicules soient susceptibles de changer très fréquemment de voie.

Baldessari et al. [80] proposent un nouveau schéma d'autoconfiguration d'adresse IP appelé GeoSAC. Le concept de lien décrit dans IPv6 est redéfini dans GeoSAC comme un domaine géographique associée avec un Point d'Attachement. La RSU est assimilée à un router, et est chargée de diffuser des RA (Router Advertisement). Le mécanisme DAD doit nécessairement être exécuté par le véhicule pour s'assurer de l'unicité de l'adresse IP. Ce mécanisme s'exécute en mode ad hoc grâce au routage géographique défini sur les couches inférieures IP. Une nouvelle méthode de correspondance basée sur le préfixe IP et la zone géographique permet d'étendre les fonctionnalités du DNS.

En dehors de l'autoconfiguration, d'autres travaux de recherche visent à proposer des architectures réseau beaucoup plus adaptées pour WAVE. Cespedes et al. [81] proposent une solution IP pour la communication V2I nommée VIP-WAVE. Des améliorations ont ainsi été apportées à WAVE parmi lesquelles : un mécanisme d'assignation d'adresse IPv6 et DAD efficace, une gestion de la mobilité basée sur PMIPv6 (Proxy MIPv6), une communication à un et deux sauts pour la V2I, où la RSU joue le rôle de MAG (Mobile Anchor Gateway).

2.3.2. Revue des schémas de mobilité IP pour WAVE

Dans la littérature, les approches et architectures utilisées pour la gestion de la mobilité sont gérées par le terminal ou par le réseau. Dans tous ces schémas, le véhicule peut être considéré comme un terminal ou comme un Routeur Mobile. Dans IPv6, les facteurs affectant négativement les performances du handover sont quasiment les mêmes :

- Le handover L3 est totalement séparé du handover L2, cela n'aide pas, par conséquent, à améliorer les performances du handover L2.
- Le scan de canaux prend un certain temps et occupe une large proportion du handover L2.
- La configuration d'adresse (ND, DAD…) prend un certain temps et occupe une large proportion du handover L3.

2.3.2.1. Véhicule considéré comme un terminal

Dans plusieurs applications pour les véhicules connectés, ceux-ci sont considérés comme des terminaux mobiles, c'est-à-dire que tous les messages sont destinés à l'OBU et aux AU.

Peng et al. [82] proposent un schéma de mobilité où les stations de bases sont positionnées à l'entrée de chaque cellule et la gestion de la mobilité des véhicules se fait en utilisant la disposition des rues plutôt que sur la distance séparant le véhicule de la station de base. Cela permet ainsi de réduire la charge réseau et d'augmenter débit d'acheminement de données.

Mouton et al. [83] proposent de gérer la mobilité des véhicules via un schéma de handover prédictif. Des données de contexte telles que la trajectoire des

véhicules, la topologie des routes et des informations réseau sont exploitées pour décider du moment du handover et du choix du prochain point d'accès. Ce mécanisme permet de réduire la durée du handover notamment en éliminant la phase de scan des points d'accès tout en améliorant la connectivité des véhicules.

Bechler et al. [84] proposent un schéma de mobilité, nommé MMIP6 (Multihop MIPv6), optimisé pour les VANET. MMIP6 apporte certaines modifications au MIPv6 comme la réintroduction du Foreing Agent (FA). Ce dernier matérialise la localisation du véhicule sur Internet masquant ainsi le caractère ad hoc des VANET. L'adressage IPv6 se fait en statique et les adresses de lien local sont évitées, car le mécanisme d'autoconfiguration d'adresse consomme une certaine quantité de bande passante. MMIP6 propose un protocole de découverte de service proactif pour les passerelles Internet (FA). Ainsi, MMIP6 est optimisé pour la scalabilité et adapté pour les véhicules.

Xiaonan et al. [85] proposent de hiérarchiser le réseau véhiculaire en le divisant en domaines routiers constitués de segments de rue et de clusters. A partir de cette architecture, les auteurs combinent le handover L3 et L2 afin d'améliorer les performances globales du handover. Ainsi le handover L3 fournit les informations de canal afin d'aboutir à un handover L2 rapide sans scan de canaux, de même le véhicule n'a plus besoin de configurer un CoA (Care-of Address).

Nguyen et al. [86] introduisent une architecture hybride centralisée-distribuée de gestion de la mobilité appelée H-DMM (Hybrid Distributed-Mobility Management). En effet la gestion décentralisée de la mobilité (DMM) présente des lacunes dans environnement de communication véhiculaire où les véhicules sont caractérisés par une mobilité très forte. Ainsi H-DMM intègre un nœud central dans l'architecture du DMM afin de jouer le rôle d'un LMA (Local Mobility Anchor) comme dans PMIPv6. Par conséquent le véhicule reçoit deux préfixes, un issu du LMA et un autre du MAR (Mobile Access Router). Le véhicule a ainsi le choix d'utiliser le préfixe approprié pour communiquer avec ses CN (Correspondant Node). Toutefois, il faut noter que durant un handover, le préfixe du MAR change. Les performances du H-DMM s'égalent à ceux du DMM sans pour autant hériter de ses lacunes. Cependant, l'exécution du mécanisme DAD pour la validation de l'adresse IPv6 générée à partir du préfixe MAR risque d'induire des délais.

2.3.2.2. Véhicule considéré comme un réseau mobile

Il existe des applications VANET et ITS où le véhicule est considéré comme un réseau mobile. Cespedes et al. [87] indexent les problèmes (adressage, optimisation des routes, scalabilité...) liées à ce genre de situation, notamment avec l'usage de NEMO-BS. Ainsi, des améliorations possibles sont proposées afin de tenir compte de la spécificité des réseaux véhiculaires comme l'utilisation des informations géographiques pour le routage en dessous de la couche IP, ou encore l'utilisation de préfixe dynamique (MNP, Mobile Network Prefix).

Soto et al. [88] proposent une extension du PMIPv6 pour NEMO (N-PMIPv6) en transformant un Routeur Mobile en un mMAG (moving MAG). Ainsi les fonctionnalités des MNN (Mobile Network Nodes) demeurent inchangées, mais ont la possibilité de recevoir des préfixes de la part du LMA grâce à la nouvelle entité mMAG. Cette dernière est connectée au MAG et bénéficie de toutes les fonctionnalités d'un nœud mobile dans un domaine PMIPv6. Afin de gérer l'accessibilité des MNN, les fonctionnalités du LMA ont été étendues en lui permettant de détecter les mouvements des MNN via le mMAG et de localiser le MAG gérant le mMAG. Le LMA a ainsi toutes informations pour construire des routes entre les CN et les MNN, et effectuer les encapsulations nécessaires. Les résultats de simulations confirment les bonnes performances du schéma N-PMIPv6, par contre, la communication multisauts dans les VANET n'est pas prise en compte.

Nguyen et al. [89] proposent un schéma de mobilité hybride centralisé/distribué H-NEMO fonctionnant de manière similaire à H-DMM [86], à la différence que H-NEMO est dédié au NEMO. Comme dans H-DMM, les MNN reçoivent deux types de préfixes, un venant du LMA et l'autre du MAR. De ce fait les MNN auront la possibilité de choisir le préfixe adéquat pour les correspondances d'une grande longévité (via le LMA, centralisé), et les correspondances éphémères (via le MAR, distribué). L'évaluation analytique montre une amélioration de la latence de handoff pour H-NEMO, en contrepartie d'une signalisation beaucoup plus importante.

Lee et al. [90] proposent une extension des solutions PMIPv6 pour NEMO (P-NEMO), nommée FP-NEMO, en y intégrant les fonctionnalités de Fast Handoff (FPMIPv6) afin de résoudre les problèmes de pertes de paquets dont souffre P-NEMO. Ainsi, dans FP-NEMO, les évènements réseau de niveau 2 sont utilisés pour anticiper la procédure de handover. De ce fait, un tunnel est établi entre l'actuel et le prochain MAG afin d'éviter les pertes de paquets durant le handover grâce au mécanisme de forwarding entre les deux MAG. Afin de supporter la mobilité des MNN, des informations telles que les préfixes (MNP) sont également inclus lors du transfert de contexte entre l'actuel et le prochain MAG. Cependant, l'établissement de tunnels entre les MAG provoque une charge supplémentaire aux MAG.

Ainsi, Ryu et al. [91] proposent une amélioration de FP-NEMO, nommée Improved FP-NEMO (IFP-NEMO), par l'élimination de la surcharge des MAG. En effet, IFP-NEMO adopte le schéma TBU (Tentative Binding Update) qui effectue un enregistrement au niveau du LMA juste avant d'exécuter un handover L2. Ainsi un tunnel est créé entre le LMA et le prochain MAG, évitant ainsi la création de tunnels supplémentaires entre les MAG. En outre, IFP-MAG supporte le handover en mode prédictif ou réactif contrairement à FP-MAG. L'évaluation des performances du IFP-NEMO, tout comme celui du FP-NEMO, s'est faite dans un cadre analytique et montre les bonnes performances du schéma IFP-NEMO comparées aux schémas FP-NEMO et P-NEMO.

2.3.3. Discussion sur la solution IP pour WAVE

Cespedes et al. [87] font partie des premiers à exposer les challenges de l'applicabilité de la mobilité IP et NEMO BS pour les VANET. Malgré l'abondance des solutions IP pour la gestion efficace de la mobilité pour WAVE, le rôle dual de l'adresse IP pose problème. En effet, l'adresse IP joue le rôle de localisateur d'un point de vue réseau, et d'un point de vue applicatif, l'adresse IP identifie un hôte en jouant également le rôle d'identificateur. Ainsi, lorsqu'on est dans un environnement de communication véhiculaire, les hôtes changent rapidement de localisation, ce qui induit un changement d'adresse IP. En outre, le mutihoming fait qu'un seul hôte se verra attribuer plusieurs identificateurs, chose non souhaitable pour le bon fonctionnement des applications. Niveau sécurité, le changement d'adresse IP provoque la rupture des sessions de communication et des contextes de sécurité, il sera nécessaire alors de renégocier de nouvelles associations de sécurité. Cette préoccupation a fait l'objet de plusieurs débats au sein du milieu de la recherche et des recommandations et alternatives ont été proposées.

2.4. Schémas de mobilité basés sur HIP

Le protocole HIP (Host Identity Protocol) [92] utilise les avantages de la séparation du localisateur/identificateur pour répondre aux problèmes liés à la mobilité, au multihoming et à la sécurité. HIP repose sur la séparation entre un nouvel identificateur, et un localisateur, plus concret, qui sera simplement l'adresse IP, réduite à un rôle moins visible, sans exigence de stabilité.

2.4.1. Présentation du protocole HIP

Le protocole HIP est basé sur deux principes : l'introduction d'un nouvel espace de nom et l'introduction d'une nouvelle couche protocolaire.

2.4.1.1. Un nouvel espace de nom

HIP fournit à chaque hôte un unique identifiant global, le Host Identifier (HI). Celui-ci représente la partie publique d'une paire asymétrique de clefs public/privé. Le HI est généré via l'algorithme RSA ou DSA, la longueur de la clef publique peut être de 512, 1024 ou 2048 bits. La taille du HI étant de grande et en plus variable, alors l'insérer dans un paquet HIP pose problème. Ainsi, deux autres formes d'identification sont créées via un condensat cryptographique du HI.

- *Host Identity Tag (HIT) :* est un hash résultant du passage à sens unique de la fonction de hachage SHA-1 sur le HI. Le HIT est au même format qu'une adresse IPv6 et résulte de la concaténation des 28 bits du préfixe ORCHID (`2001:0010::/28`) et des 100 bits résultant du hachage du HI.
- *Local-Scope Identifier (LSI) :* est un identifiant à 32 bits, il est au format 1.x.y.z. Les 24 bits d'ordres inférieurs du LSI représentent les 24 bits d'ordres inférieurs du HIT.

2.4.1.2. *Une nouvelle couche de protocole*

Le protocole HIP introduit une nouvelle couche, Host Identity Layer, entre la couche IP et la couche de transport. Cette nouvelle couche fait le mappage entre le HIT, utilisé par les couches de transport et d'application, et l'adresse IP. Par conséquent, le TCP et l'UDP gardent leurs propriétés même si l'hôte change de point d'attachement réseau. Ainsi, le changement d'adresse IP est transparent aux sockets (couche transport). La couche HIP met à jour le mappage entre HI et adresse IP lorsque l'hôte change de localisation ; la mobilité est ainsi assurée.

2.4.1.3. *HIP Base Exchange*

Le Base Exchange (BEX), défini dans le RFC5201 [93], consiste à l'échange de quatre messages entres deux hôtes (handshake), voir *Figure 15*. Cette association englobe toute les propriétés et aspects que constitue le contexte de communication HIP entre deux hôtes. Ce handshake a pour finalité, l'authentification des hôtes, l'établissement d'un mappage HI-to-IP, l'élaboration d'un secret partagé qui sera utilisé pour le chiffrement symétrique et l'authentification. On appelle Initiateur l'hôte qui amorce le Base Exchange, l'autre est appelé Répondeur.

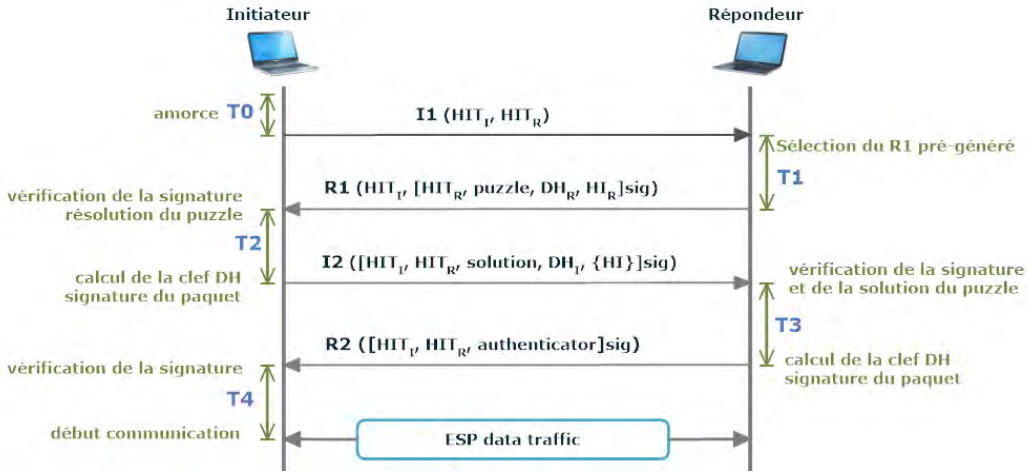


Figure 15 : Le handshake HIP Base Exchange

2.4.1.4. *L'extension RVS*

L'architecture HIP introduit un nouveau mécanisme de services de Rendezvous géré par un serveur de rendezvous (RVS). L'avantage majeur du RVS est de mettre à jour rapidement les changements liés aux informations de localisation et d'identification. Le client RVS est un nœud HIP qui utilise l'extension d'enregistrement HIP [94] pour créer une correspondance HIT/IP au niveau du RVS. Ainsi, d'autres hôtes HIP pourront utiliser l'adresse du RVS pour initier le Base Exchange avec ce nœud. Le RVS sera alors chargé de relayer le paquet I1 à l'hôte concerné, comme le montre la *Figure 16*.

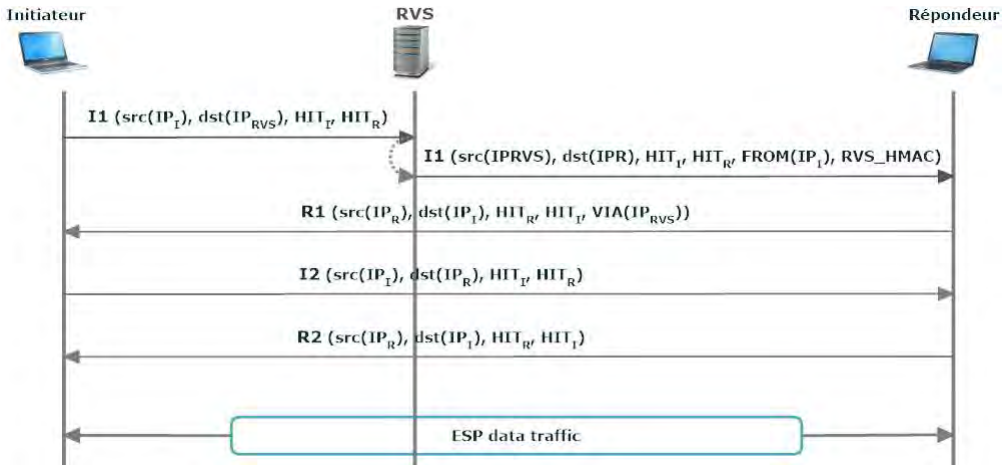


Figure 16 : Paquet I1 relayé via un RVS

2.4.2. La mobilité dans HIP

Le RFC5206 [95] définit l'extension de mobilité pour HIP. La procédure de mobilité, voir *Figure 17*, s'effectue par l'échange de paquets UPDATE servant à :

- La notification à l'autre nœud du changement d'adresse IP
- L'association de la nouvelle adresse IP avec le HIT dans la couche HIP
- La réponse avec un ECHO_REQUEST pour valider la nouvelle localisation

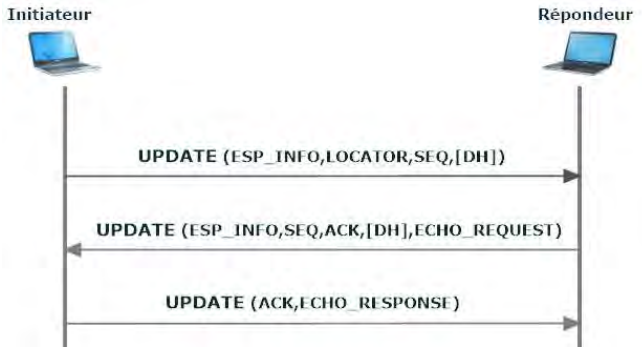


Figure 17 : Procédure de mobilité HIP

Lorsque l'un des hôtes change d'adresse IP, il envoie un paquet UPDATE, à son pair, contenant la nouvelle adresse IP dans le paramètre LOCATOR. Le paquet inclut également le paramètre ESP_INFO contenant les valeurs de l'ancien SPI (Security Parameter Index) et du nouveau SPI. Le paramètre SEQ est utilisé si le paquet a besoin d'un acquittement. Une fois qu'il a reçu le paquet UPDATE, l'autre nœud répond à son tour par un paquet UPDATE. Ce paquet contient, entre autres, le paramètre ACK pour acquitter le paquet reçu, le paramètre ECHO_REQUEST afin de vérifier l'accessibilité de la nouvelle adresse. Lorsque l'hôte mobile reçoit le paquet UPDATE, il réplique par l'envoi d'un troisième paquet UPDATE contenant le paramètre ECHO_RESPONSE. Ce paquet clôture la procédure de mobilité et n'a pas besoin d'être acquitté.

2.4.3. Revue des schémas de mobilité HIP pour WAVE

Dans la littérature, on note un intérêt manifeste par la communauté de recherche à expérimenter HIP dans les réseaux véhiculaires.

Les premières solutions HIP pour les réseaux véhiculaires sont basées sur NEMO. Melen et al. [96], proposent un schéma permettant aux nœuds HIP de bénéficier de la mobilité réseau. La délégation de signalisation entre les nœuds (MNN) et le Routeur Mobile (MR) est utilisée pour des raisons d'optimisation de route. L'accessibilité des MNN est assurée grâce à un schéma de translation d'adresse privée/publique gérée par le MR. Quant à la gestion de la mobilité, elle est également gérée par le MR qui sera chargé d'envoyer des messages UPDATE aux CN pour le compte des MNN.

Ylitalo et al. [97] proposent une extension du schéma de Melen en minimisant l'envoi de la signalisation via les liens sans fil. Ainsi, le MR exécute une seule procédure UPDATE avec une entité filaire, considérée comme son proxy de signalisation. Ce proxy sera chargé d'effectuer la procédure UPDATE avec les CN par le biais des proxys de signalisation de ces derniers.

Un autre schéma HIP pour NEMO a été défini par Novaczki et al. [98] en introduisant une nouvelle entité, le mRVS. Ce dernier est co-localisé avec le MR et s'enregistre auprès du RVS pour le compte des MNN. Contrairement aux deux schémas précédents, le mRVS permet aux CN d'être également les Initiateurs des échanges HIP. Notons qu'en dépit de leurs adresses routables et accessibles, les MNN s'appuient sur la translation d'adresse pour communiquer avec leurs CN. De ce fait, en cas de handover, seul le MR exécutera la procédure UPDATE avec le RVS et tous les CN afin de maintenir l'accessibilité des MNN.

Toledo et al. [99], proposent un modèle mathématique des schémas de mobilité suscités [96]–[98] afin d'y effectuer une analyse de la charge réseau. Les résultats obtenus révèlent des problèmes de charge réseau significatif lors de la procédure d'enregistrement au niveau du RVS. Les auteurs recommandent alors de faire un compromis entre l'accessibilité directe des MNN et la charge réseau sur la base des scénarios NEMO et des besoins des utilisateurs.

Toledo et al. [100] proposent un nouveau schéma de sécurité et de mobilité nommé NeMHIP. Ce schéma répond aux exigences de sécurité dans le contexte ITS, plus précisément au niveau de la gestion de la mobilité et des services de données qui seront déployés. Ainsi, le MR est chargé d'enregistrer les LMN (Locally Mobile Nodes) auprès du RVS via une unique procédure de Base Exchange. En effet, l'ensemble des HIT des LMN sont inclus dans les échanges du BEX entre le MR et le RVS. Les auteurs apportent toutes les détails sur le mécanisme permettant d'assurer l'intégrité et la confidentialité de bout en bout lors des échanges entre les LMN et leurs CN, en dépit des procédures de mobilité.

En plus des solutions basées sur NEMO, d'autres auteurs se sont penchés sur la combinaison du PMIPv6 et de HIP. Ainsi, Cepedes et al. [101] proposent un schéma hybride basé sur HIP et PMIPv6 afin de pouvoir exploiter les avantages de ces deux protocoles. Ce schéma permet aux nœuds supportant HIP et ceux ne supportant pas HIP de pouvoir bénéficier d'un service de mobilité sécurisé. Ainsi pour les nœuds supportant HIP, les auteurs s'appuient sur les fonctionnalités du mMAG définit dans [88] afin d'assurer leurs accessibilités. Une fois que ces nœuds obtiennent un préfixe réseau via le LMA, ils peuvent dès lors initier une session de communication HIP. Afin de gérer la communication des nœuds ne supportant pas HIP, les auteurs s'appuient cette fois-ci, sur les fonctionnalités du proxy HIP [102]. Ainsi, le MR intercepte les paquets des nœuds non-HIP puis les encode en paquets HIP avant de les transférer aux CNs. Notons que tous les deux types nœuds devront enregistrer leur localisation auprès du RVS afin d'assurer leurs accessibilités. Cela est géré par le MR pour le compte des nœuds non HIP, et pour les nœuds HIP, cela est exécuté par eux-mêmes. Lors d'un handover intradomaine (changement de MAG uniquement), la signalisation est gérée au niveau des MAG et du LMA, par conséquent transparent aux nœuds. Par contre lors d'un handover interdomaine (changement de MAG et de LMA), les préfixes réseau du MR changent. Dès lors le MR initie la procédure UPDATE HIP avec tous les CN pour le compte des nœuds ne supportant pas HIP. Quant aux nœuds HIP, la procédure UPDATE est gérée par eux-mêmes. Notons que des messages UPDATE devront également être envoyés au RVS afin de maintenir l'accessibilité des nœuds. Les auteurs terminent par effectuer une analyse de performance de leur schéma sur une base analytique.

Dans [103], Cepedes et al. apportent des clarifications sur la signalisation utilisée et étendent aussi l'analyse des performances de leur schéma précédent, notamment en y intégrant des résultats de simulations effectués dans un scénario réaliste d'environnement urbain.

2.4.4. Discussion sur la solution HIP pour WAVE

L'utilisation de HIP dans WAVE nécessite de faire un compromis entre la sécurité, la mobilité et la latence. En effet, certaines applications des VANET exigent une latence très faible, tandis que certaines applications de l'IoV exigent une garantie en termes de sécurité et de continuité de service. En outre, la délégation de signalisation au MR est problématique dans le sens où si plusieurs associations HIP parallèles venaient à être établies par les MNN, le MR se verra vite saturé par le grand nombre d'opérations cryptographiques à effectuer. En somme, HIP ne pourra pas être déployé sur l'ensemble du réseau véhiculaire, notamment au niveau de la communication V2V caractérisée par des applications critiques et à très fortes sensibilités temporelles.

2.5. Schémas de mobilité et d'optimisation réseau basés sur le SDN

Les réseaux logiciels sont un ensemble d'approches qui visent à concevoir, construire et exploiter autrement les réseaux. Les éléments qui militent pour cette évolution des réseaux sont, entre autres, la virtualisation et l'automatisation. La virtualisation permet de s'affranchir des contraintes des équipements génériques pour l'implémentation de fonctions réseau. Cette approche est connue sous le nom de NFV (Network Functions Virtualization) [104], [105]. L'autre approche est l'automatisation de la mise en œuvre et de la gestion des réseaux par le biais du paradigme SDN (Software-Defined Networking) [106], [107].

2.5.1. Software-Defined Networking

Le terme Software-Defined Networking se réfère, à l'origine, à une architecture réseau où un point central gère le plan de contrôle (la partie qui détermine comment les paquets sont transmis), tandis que les commutateurs/routeurs physiques n'ont plus à prendre en charge que le plan de données (la partie qui est responsable de la transmission des paquets). SDN repose sur l'utilisation des interfaces standardisées basées sur des API ou protocoles réseaux. Depuis 2014, des efforts de standardisation sont menés, portés notamment par les organismes de standardisation ONF (Open Networking Foundation) [108], IRTF (Internet Research Task Force) [109] et ITU (International Telecommunications Union) [110]. Malgré des approches et focus différents, ces organismes ont un objectif en commun, permettre la programmabilité des réseaux. Ainsi, un certain nombre de principes définissent le SDN, voir *Figure 18*, à savoir :

- Séparer le plan de contrôle du plan de transmission.
- Centraliser et mutualiser le plan de contrôle entre tous les équipements.
- Réduire les équipements réseau classiques, à leur fonction la plus élémentaire ; faire de la commutation.
- Les serveurs chargés de fournir le plan de contrôle sont généralement appelés *contrôleurs SDN*.

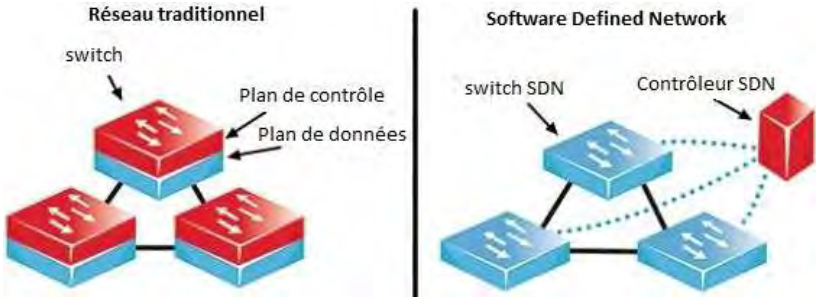


Figure 18 : Principe du SDN : découplage plan de donnée du plan de contrôle

2.5.1.1. *Architecture SDN*

Le fondement du SDN réside sur le découplage du plan de données du plan de contrôle. Un plan est une collection de ressources (matériel et logiciel) qui sont responsables d'une activité réseau donnée. Les fournisseurs SDN offrent un large choix d'architectures concurrentes, néanmoins le modèle représenté par la *Figure 19* reste le plus répandu.

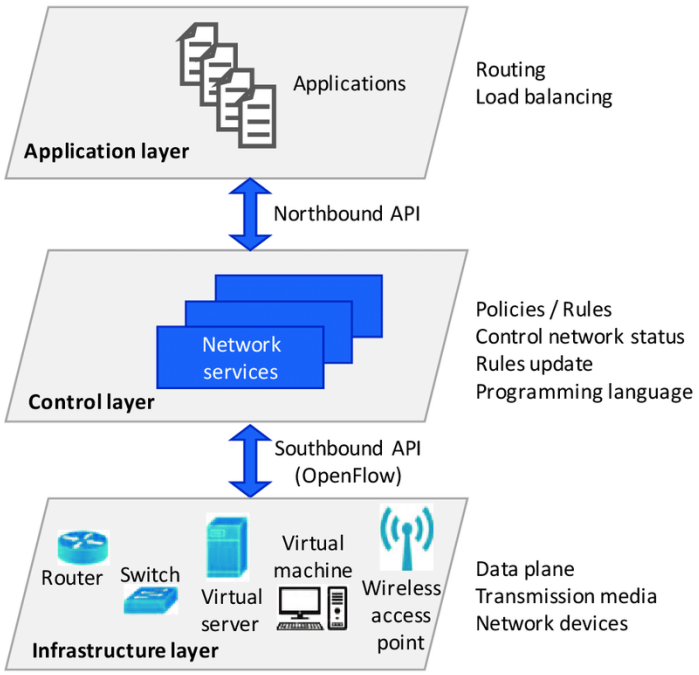


Figure 19 : Architecture de base du SDN [111]

- *Plan de données (data plane)*

Désigne les composants à hautes performances (switches) dédiés à l'acheminement de données sur le réseau, c'est-à-dire transférer des paquets d'une interface à une autre en fonction des règles établies par le contrôleur.

- *Plan de contrôle (control plane)*

Désigne l'intelligence de l'équipement (serveur) responsable de remplir la table de transmission. Ce plan connaît le moindre détail se rapportant au réseau et décide comment diriger le flux du trafic réseau.

En outre, le RFC 7426 [109] modifie l'architecture de base du SDN en y définissant trois autres plans supplémentaires, à savoir :

- *Plan des opérations (operational plane)*

Désigne la gestion globale d'un composant de l'architecture de SDN. Ce plan exécuté par le composant réseau tel que le switch.

- *Plan d'administration (management plane) :*

Fournis les instructions de base indiquant comment le composant réseau doit interagir avec le reste du réseau et gère aussi la supervision de ce composant.

- *Plan des applications (application plane)*

Désigne les applications et services développés pour piloter le réseau. Ce plan construit une vue abstraite du réseau en se basant des informations reçues du contrôleur.

L'architecture SDN est composée de trois couches, chacune étant composée d'une ou de plusieurs entités responsables de l'exécution des fonctions d'un plan spécifique.

- *Contrôleur SDN*

Cette une entité centralisée et est considérée comme le « cerveau » du réseau, car disposant d'une maîtrise globale de l'infrastructure et est capable de s'informer en temps réel sur l'état et l'activité des équipements (physiques ou virtuels) qu'ils pilotent. Le contrôleur est responsable de l'exécution des fonctions du plan de contrôle. Le contrôleur reçoit des instructions et des exigences de la part de la couche des applications et les relaie aux dispositifs réseau. Ainsi, un NIB (Network Information Base) est construit au niveau du contrôleur et permet à ce dernier de savoir comment implémenter chaque ordre abstrait, trouver les équipements qui doivent être reconfigurés, s'assurer de la capacité de ces derniers à implémenter une directive, les API supportées par l'équipement... [112].

- *Dispositifs réseau*

C'est l'ensemble des entités commandées par le contrôleur SDN et capables d'effectuer des opérations de manipulation et de transfert de paquets. Ces dispositifs sont responsables de l'exécution des fonctions du plan de données. Dans le modèle SDN du RFC 7426 [109], aucune distinction n'est faite entre un dispositif réseau physique ou virtuel. Néanmoins, l'essentiel des dispositifs est composé de commutateurs à hautes performances à base d'ASIC (Application Specific Integrated Circuit).

- *Applications SDN*

C'est un ensemble d'applications qui permet à l'administrateur de réseau de très rapidement implémenter des services via des programmes dynamiques automatisés. Ces services peuvent être le routage de flux, la mise en œuvre de politiques de QoS pour les flux, la sécurité de bout en bout des flux, le filtrage de flux… Une application peut construire un modèle abstrait du réseau et prendre des décisions en s'aidant des informations (événements et statistiques) collectées par le contrôleur. Une application donne un ordre abstrait au contrôleur et ce dernier se chargera d'implémenter cet ordre au niveau du réseau.

Afin d'assurer la communication entre les différentes couches, le SDN propose l'usage d'interfaces. Cela peut être fait avec un protocole réseau, ou une API. On distingue essentiellement deux types d'API :

- *API Southbound*

C'est une interface de communication utilisée par le contrôleur et les dispositifs réseau. Ces APIs permettent au contrôleur d'appliquer dynamiquement des changements suivant les besoins et demandes en temps réels. Plusieurs API Southbound sont déjà standardisées telles que Openflow, Netconf, OVSDB, SNMP, SSH, BGP, XMPP...

- *API Northbound*

C'est l'interface de communication partagée par les applications SDN et le contrôleur. Ces APIs facilitent l'orchestration efficace et l'automatisation du réseau telle que définie par les applications. Il n'existe pas encore une standardisation de l'API Northbound. Néanmoins, les éditeurs logiciels de contrôleurs publient la documentation de leur API afin de permettre d'interfacer des applications.

2.5.1.2. OpenFlow

OpenFlow [113], [114], publié par l'ONF, est actuellement le protocole le plus abouti et celui suscitant le plus d'intérêt dans le domaine de la recherche et de l'industrie. Un contrôleur OpenFlow peut utiliser un canal sécurisé SSL/TLS pour envoyer des commandes afin de modifier la table de flux des switches. La fonction de base de ces switches est de faire de la commutation tout en réalisant toutes les modifications nécessaires sur les paquets. La fonction de correspondance de paquet (packet-matching function) est très importante pour les switches OpenFlow. Elle est basée sur la table de flux, voir *Figure 20*.

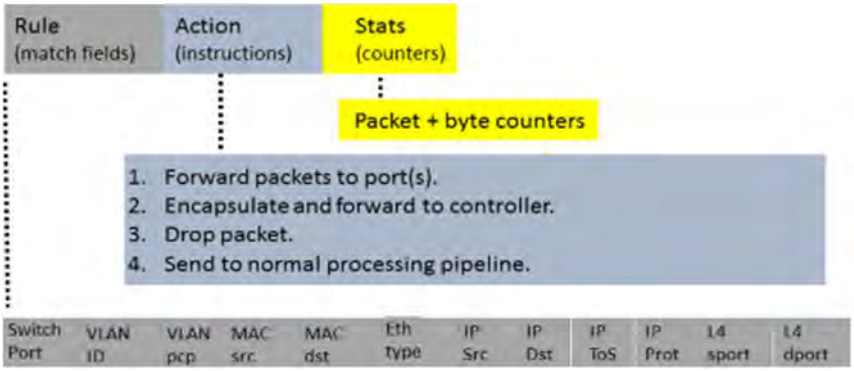


Figure 20 : Structure d'une table de flux

La table de flux est constituée de plusieurs entrées, et chaque entrée contient des champs d'entêtes, des compteurs et des actions. Les champs d'entêtes sont utilisés pour faire la correspondance entre un paquet entrant et une entrée de la table de flux. S'il y a correspondance, le paquet est traité suivant les actions définies pour

cette entrée. Sinon, le paquet est envoyé au contrôleur OpenFlow. Ce dernier pourra alors créer une nouvelle entrée pour ce nouveau paquet ou simplement l'ignorer. Le compteur est utilisé pour des buts de statistique sur le flux. Le contrôleur OpenFlow ayant une vue globale sur le réseau peut également agir de manière proactive afin de modifier certains règles et paramètres réseaux.

2.5.2. Revue des architectures SDN pour les réseaux véhiculaires

Le SDN a été pensé et conçu pour les réseaux filaires notamment le cœur du réseau et les réseaux d'accès. Cependant, la transformation de l'écosystème réseau par le biais du SDN s'est vite répandue vers les environnements sans fil. L'étude des architectures VANET basées sur le SDN est un nouveau domaine de recherche, on ne trouve d'ailleurs dans la littérature que quelques travaux traitant de ce sujet. Ces travaux sont motivé par le modèle de communication de l'IoV qui doit être nécessairement gérable, contrôlable, opérationnelle et fiable [115].

2.5.2.1. Idée de base : Software-Defined VANET

Ku et al. [116] sont les pionniers de l'introduction du SDN dans les VANET. Leur solution, nommée Software-Defined VANET (SDV), utilise deux types de canaux : un canal à longue portée (LTE/WiMAX) pour le plan de contrôle, et un canal à haut débit (WiFi) pour le plan de donnée. Trois entités constituent le SDV : le SDN controller, le SDN wireless node et le SDN RSU. Le SDN wireless node intègre un agent SDN local chargé de la réception des plans de données, et dans certains cas, assurer le transfert de ces plans de données vers d'autres nœuds. Trois modes de contrôle possibles sont proposés pour le SDV, il s'agit du mode centralisé, distribué et hybride. L'architecture des VANET ne garantissant pas un maintien de la connectivité/accessibilité des nœuds tout au long de leur trajet, le mode de contrôle hybride se présente alors comme la solution la plus robuste. En effet, en cas de rupture de communication avec le SDN controller, le mode hybride permet aux SDN wireless node et aux SDN RSU de maintenir, au prix d'une intelligence réduite, le bon fonctionnement du système grâce à un mécanisme de secours (utilisation de protocoles de routage traditionnels des VANET). L'avantage du SDV repose sur la possibilité d'améliorer la sélection des routes, des fréquences/canaux et de la puissance d'émissions des systèmes sans fil. Les résultats de simulations confirment la meilleure efficacité du SDV par rapport aux protocoles de routage traditionnels des VANET (GPSR, OLSR, AODV…). L'architecture SDV a apporté certes des améliorations en termes de flexibilité et de scalabilité, cependant certains problèmes des VANET demeurent inhérents.

2.5.2.2. Architecture distribuée grâce au Fog Computiing

Afin d'améliorer la scalabilité dans les VANET, Kazmi et al. [117] proposent une architecture SDN distribuée. Leur stratégie consiste à décentraliser l'intelligence sur des domaines composés de contrôleurs (DC). Les contrôleurs SDN sont organisés de manière hiérarchique où un contrôleur root (RC) est chargé

d'orchestrer l'ensemble des DC. Ces derniers sont chargés du traitement des requêtes des nœuds (plan de données) et de l'exécution de services réseau. La pertinence de leur approche est validée par des tests de simulation qui confirment un gain de la scalabilité et une réduction du délai d'accès au service.

Dans la même lancée, Truong et al. [24] proposent une amélioration du SDV par le bais du Fog Computing. En plus des trois entités du SDV, leur architecture nommée FSDN VANET intègre deux nouvelles autres entités : le SDN Road-Side-Unit Controller (RSUC) et le Cellular Base Station (BS). Les nœuds SDN jouent ainsi un rôle dual. En effet, le SDN Controller devient également l'orchestrateur et le gestionnaire de ressources Fog tandis que le reste des entités deviennent des nœuds Fog. L'intégration d'entités Fog impose logiquement un mode de contrôle hybride dans le sens où le RSUC et le BS gèrent une partie du système en se basant sur leur propre intelligence et sur une connaissance locale de l'état du système. Le RSUC et le BS supportent des technologies de virtualisation tels que l'implémentation d'hyperviseurs, ce qui permet au besoin d'instancier, de répliquer ou de faire migrer des services dans le Fog. Cependant ces opérations impactent négativement sur la latence et la Qualité d'Expérience (QoE).

2.5.2.3. Algorithmes d'optimisation du trafic

Salahuddin et al. [118] proposent une solution d'optimisation nommée RSU Cloud Resource Manager (CRM). Le CRM vise à minimiser le délai et le coût opérationnel, en d'autres termes, minimiser le nombre d'instanciations/réplication de services et les reconfigurations réseau. Le CRM est modélisé sous forme de problème de programmation linéaire entière multiobjectif dont la résolution, basée sur une méthode heuristique, permet d'aboutir à des solutions POF (Pareto Optimal Frontier). Les résultats de simulation confirment bien l'efficacité de leur CRM heuristique. Cependant, leur solution se limite à la partie filaire composée exclusivement de RSU sans prendre en compte la partie sans fil du VANET et l'optimisation du plan de contrôle.

Li et al. [119] se basent alors sur la théorie du jeu afin d'optimiser le plan de contrôle d'un réseau 5G couplé avec un VANET. La modélisation des interactions entre le contrôleur SDN et les véhicules est formulée sous forme de jeu de Stackelberg (meneur-suiveur). Le but recherché est de trouver un compromis entre la latence et le coût d'accès au contrôleur SDN. Les résultats des tests de performance prouvent bien le bon fonctionnement de leur stratégie.

Zheng et al. [120] proposent un schéma d'optimisation du délai d'allocation des ressources pour un réseau véhiculaire hétérogène (HetVNET) nommé SERVICE [121]. Ce dernier est le résultat de l'intégration du SDN dans les HetVNET dont l'architecture est basée sur le Cloud-RAN. SERVICE utilise un plan de contrôle hiérarchique pour garantir une QoS en dépit de la nature très dynamique des véhicules. Leur schéma permet d'obtenir des gains de performance significatifs. Cependant l'utilisation d'une architecture hiérarchique ne résout complètement pas

le problème de surcharge de réseau. En effet, dû à leur forte mobilité, les véhicules doivent très fréquemment envoyer des informations sur leurs statuts (localisation, vitesse, connectivité…) au contrôleur SDN.

2.5.2.4. Schémas de routage

Zhu et al. [122] proposent un framework de routage SDN pour les VANET afin de réduire le délai de transmission et la surcharge réseau. Les véhicules envoient périodiquement leur statut au contrôleur SDN afin de faciliter la mise à jour des tables de routage. Cependant lorsque la connexion est momentanément rompue avec le contrôleur SDN, une nouvelle métrique nommée MOT (Minimum Optimistic Time) est utilisée pour construire une nouvelle route. Durant le MOT, on suppose que les mêmes paramètres de routage demeurent valides. Malgré des résultats de simulation confirmant les bonnes performances du framework, leur solution ne pourra pratiquement pas être implémentée dans les VANET. En effet l'architecture utilisée ne reflète réellement pas celle des VANET, car étant basée sur une solution réseau purement IP : le WiFi (pour la communication V2V) et le WiMAX (pour la communication V2I).

Contrairement à Zhu et al. [122], Liu et al. [123] se basent sur l'architecture des VANET et du SDN pour concevoir un algorithme de GeoBroadcasting. Ainsi lorsque la RSU reçoit un premier de message d'alerte, ce dernier est envoyé au contrôleur SDN sous forme de *packet-in* message. Le contrôleur SDN se base sur les informations topologiques et géographiques des RSU environnants pour construire les chemins de routages des RSU de destination. De nouvelles entrées OpenFlow sont envoyées aux switches SDN, de ce fait les messages d'alertes similaires emprunteront le même chemin sans nécessiter l'intervention du contrôleur SDN. Par rapport aux solutions traditionnelles de GeoBroadcasting, comme le C2CNET, leur solution a l'avantage de considérablement réduire la charge réseau et la latence des messages. La limite de leur solution est qu'il ne gère que le GeoBroadcasting et ne traite pas les autres types de communication.

He et. al [25] proposent une solution généralisée de routage basée sur l'architecture des VANET et du V2-Cloud. La prédiction de la trajectoire des véhicules est utilisée afin de mettre à jour leur statut (localisation, vitesse, connectivité…). Cette approche a l'avantage de réduire considérablement la charge réseau et en même temps éviter l'utilisation de mécanisme de secours (protocoles de routage traditionnels des VANET) comme proposé dans [116]. Une personnalisation des API OpenFlow est également fourni pour tenir compte de la nature des VANET. Ainsi, un plan de données mobiles (véhicules) et un plan de données stationnaires (RSU) sont définis dans le SDN. Aussi, en fonction des applications (fortes contraintes temporelles, sensibilité aux pertes de paquets), des topologies et protocoles de routage spécifiques pourront dynamiquement être choisis. Les auteurs projettent d'étendre les possibilités d'OpenFlow afin de supporter la gestion d'interfaces réseau multiples.

2.6. Conclusion

La gestion de la mobilité dans le contexte de la communication véhiculaire est une opération complexe du fait des caractéristiques mobiles assez particulières des véhicules (vitesse, structure des routes, comportement du conducteur…). Dans la littérature, plusieurs solutions sont proposées pour tenir compte de ces particularités. Les premières solutions sont basées sur le standard 802.11p/WAVES. Cependant seulement une certaine frange d'applications peut fonctionner avec cette approche. Pour pallier à cela, des solutions basées sur IP ont été proposées, il s'agit en effet des solutions NEMO BS, de Mobile IPv6 et dérivées. Cependant, le rôle dual de l'adresse IP pose problème. Ainsi, le protocole HIP propose la séparation de l'identificateur et du localisateur (adresse IP) afin d'éviter que le changement d'adresse IP ait un impact sur les couches supérieures. En dehors de la gestion mobilité, HIP apporte également des solutions de sécurité robustes.

Il est également important de gérer le flux de données pour les réseaux véhiculaires. L'approche SDN semble être aujourd'hui la plus prometteuse, car permet une gestion plus flexible des réseaux. Dans la littérature, des solutions SDN pour la communication véhiculaire visent essentiellement à optimiser le trafic, à gérer la mobilité ou encore à gérer le routage.

Dans les chapitres qui vont suivre, des propositions de modèle de LM seront faites. Les standards de communication tels que HIP, SDN seront utilisés et permettront d'assurer le bon acheminement des données suivant les modèles de communication proposés.

Chapitre 3 : Proposition d'un modèle de LM à base d'un véhicule vu comme un terminal communicant simple

Ce troisième chapitre est consacré à la proposition de schémas de mobilité basés sur le protocole HIP. La gestion du handover et la sécurisation de la communication sont les deux aspects développés dans ces schémas de mobilité. Le Laboratoire Mobile (LM) est modélisé, ici, comme un nœud réseau multi-interfaces communiquant avec des entités internes et externes. Une évaluation des performances des schémas HIP est proposée à la fin de ce chapitre.

3.1. Modèle de terminal communicant mobile

Le recours aux Laboratoires Mobiles vient répondre à plusieurs besoins exprimés par les acteurs de l'industrie, gouvernement, médecine, armée, organisme de recherche… Le point commun de ces laboratoires, en dépit leurs domaines d'applications très divers, est qu'ils sont mobiles et tractés le plus souvent par des véhicules.

3.1.1. Caractéristiques et défis des Laboratoires Mobiles

L'approche la plus utilisée est souvent d'embarquer des équipements de relève dans le véhicule. Ainsi, un LM peut être caractérisé comme :

- *Une entité satellite d'un laboratoire fixe ou central*

L'entité est utilisée en soutien d'un laboratoire central. Sa mission est alors l'acquisition de données et le stockage pour de futures analyses. Néanmoins, un premier prétraitement de ces données peut être effectué.

- *Une unité pleinement opérationnelle et autonome*

L'acquisition des données et leur traitement se font sur place. Ce type de laboratoire requiert plus de personnels et d'équipements.

Plusieurs travaux [124]–[127] ont été menés quant à l'utilisation des véhicules comme LM. Les domaines d'applications traditionnels des LM sont principalement le monitoring de l'environnement, la recherche scientifique, la biosécurité, les sciences de l'éducation… Plusieurs défis devront être relevés par les LM afin qu'ils puissent s'intégrer dans l'ère des technologies de communication actuelle :

- *Optimisation de la connectivité, la mobilité, la transparence*

Cela passe par la mise en place d'interfaces multiples de réseaux sans fil dans l'écosystème des réseaux véhiculaires, la sécurisation de la mobilité par le passage à l'échelle (LIPS, MIPv6, HIP), l'amélioration de l'expérience utilisateur (fiabilité de la connexion, handover transparent, persistance des sessions). Ainsi, une stratégie de sélection d'interfaces réseau (TOPSIS, DiA, MIH, ABC…) suivant un certain nombre de critères (exigences de l'application, coût, sécurité, l'énergie…) devra être mise en place.

- *Sécurisation de la connectivité*

Les contraintes de sécurité devront être pleinement prises en compte dans la communication véhiculaire. Par conséquent, une sécurisation de la communication (chiffrement, technologies VPN…) entre le réseau interne du LM et les réseaux étendus devra être mise en place. Sécuriser l'accès (pare-feu, prévention d'intrusion, AAA…) au réseau du véhicule devra être également pris en compte.

3.1.2. Proposition de modèle de Laboratoire Mobile

Le modèle de LM proposé définit une partie interne constitué d'un système de partage et de collecte d'informations via des dispositifs transportés par le véhicule ou à proximité, et une partie externe assurant l'accessibilité et la connectivité avec le monde extérieur, c'est-à-dire Internet.

- *Partie interne*

Le dispositif transporté ou à proximité est un nœud polyvalent, en fonction des besoins, plusieurs tâches peuvent lui être assignées. Dans un WSN, ce dispositif peut agir comme capteur mobile embarqué, une mule à données (data mule) ou point d'accès. L'unité de traitement de l'information du dispositif effectue un premier traitement des données pour assurer un échange d'informations beaucoup plus intelligent entre les différentes entités de la partie interne. Ses capacités de calcul étant limitées, ce dispositif ne peut traiter qu'un nombre restreint de requêtes. Ainsi, pour le traitement et l'exploitation de grande quantité de données, les données sont transmises vers l'unité de calcul et de traitement embarqué dans le véhicule. Ce dernier est doté d'un ensemble de technologies d'accès radio (ZigBee, Bluetooth, WiFi...) conçu assurer la compatibilité entre les différents dispositifs et applications de la partie interne. La sécurité est un autre atout qui sera développée dans la conception du LM.

- *Partie externe*

La partie externe fournit une multiplication et une optimisation des moyens d'accès Internet par l'installation de systèmes sans fil hétérogènes et omniprésents (WiFi, WiMAX, 2G/3G/4G, DVB-S.…). Un système central est implémenté pour jouer le rôle d'hyperviseur entre la partie interne et la partie externe du LM. Le but est de faire bénéficier aux autres composants internes du LM de la multiplicité et

de la diversité de moyen d'accès à Internet. Le paradigme Always Best Connected (ABC) [128] permettra d'exploiter de manière optimale les RATs (Radio Access Technologies) mises à la disposition du LM. En effet, ce concept permet aux utilisateurs d'applications communicantes de pouvoir disposer n'importe quand et n'importe où de la solution d'accès répondant au mieux à leur besoin.

3.2. Architecture de communication basée sur HIP

Etant dans un contexte de mobilité, les hôtes ont tendance à changer de point d'attachement. Le protocole HIP gère d'une seule manière la mobilité. HIP ne fait pas la différence entre la micromobilité et la macromobilité. Cette procédure accroit la latence du handover augmentant ainsi la probabilité de perte de paquets. Nous proposons dans ce qui suit, un schéma de mobilité proactif, nommé HIP_{DISASS}, permettant une gestion efficace de la mobilité des hôtes. Nous proposons également une architecture réseau permettant l'authentification et la confidentialité indépendamment du type de réseau d'accès utilisé.

3.2.1. Description de l'architecture du domaine

Le domaine du LM est constitué d'un serveur RVS central, le pLRVS (proxy Local Rendezvous Server), et d'un ensemble distribué de serveurs RVS nommés S-RVS (Subnet Rendezvous Server). Les LM peuvent, lors de leurs déplacements, quitter un sous-réseau pour entrer dans un autre. Une gestion mutuelle de la mobilité est assurée par le terminal (*LM*) et le réseau (S-RVS et pLRVS). La *Figure 21* montre les détails de l'architecture du domaine et de la signalisation.

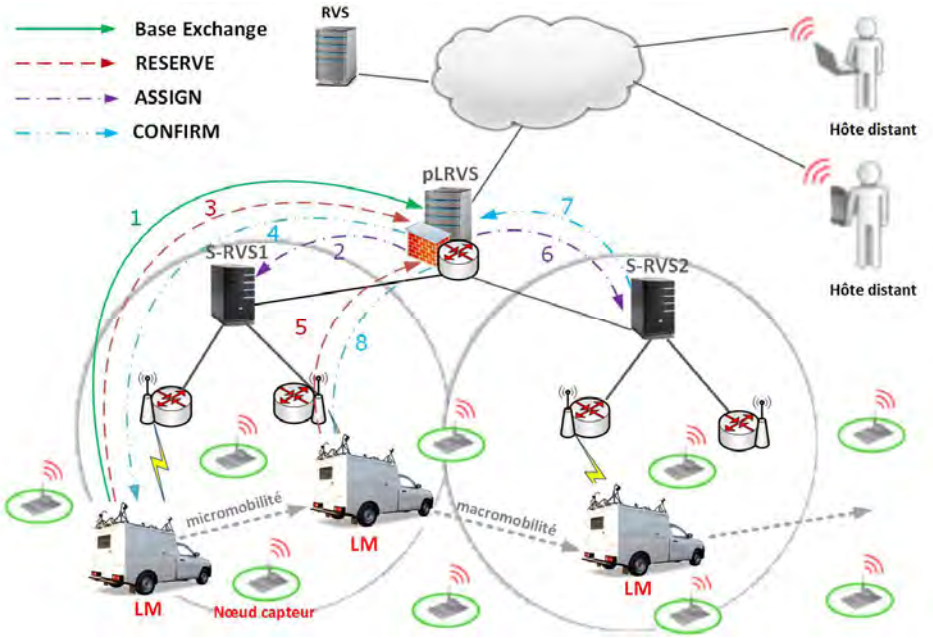


Figure 21 : Architecture du domaine et signalisation

Le pLRVS est le responsable de l'accessibilité des nœuds dans le domaine. Il participe d'une part au processus du handover en héritant de certaines fonctionnalités du LRVS introduit par Novaczki et al. [129]. D'autre part, le pLRVS joue le rôle de proxy en mettant en cache les commandes à distance destinées au LM lors d'un handover. Le pLRVS est également l'interface entre le LM et l'extérieur du domaine en jouant le rôle de passerelle et en gérant l'authentification et les droits d'accès. Chaque domaine contient un seul pLRVS co-localisé avec le routeur/modem Internet de ce domaine.

Le S-RVS, introduit par Muslam et al. [130], assure la signalisation d'un attachement pour le compte d'un nœud mobile. Dans notre cas, le S-RVS sera le responsable de la connectivité des nœuds dans le domaine en assignant les préfixes réseaux aux LMs. Le S-RVS est co-localisé avec les Points d'Attachement (PoA) et couvre une zone géographique bien définie du domaine. Le nombre de S-RVS dépend du nombre de sous réseaux désirés pour le domaine.

3.2.2. Authentification et droit d'accès

Dans cette partie, nous proposons d'intégrer des fonctionnalités de sécurité sur l'extension RVS [131]. Le rôle du pLRVS est primordial dans ce mécanisme. En effet, il est le premier rempart face aux menaces externes, et est la seule entité qui puisse être directement accessible de l'extérieure. Le mécanisme de sécurité proposé agit essentiellement durant la phase d'enregistrement et du Base Exchange.

Le pLRVS joue le rôle de serveur de rendezvous en permettant d'une part de localiser les nœuds (LMs) du domaine, et d'autre part, de relayer les paquets I1 aux nœuds destinataires. Pour bénéficier de ce service, l'hôte distant doit au préalable s'enregistrer auprès du pLRVS. Ce dernier vérifie systématiquement si le HIT de l'hôte distant est autorisé à bénéficier d'un tel service, sinon l'enregistrement est rejeté, voir *Figure 22*.

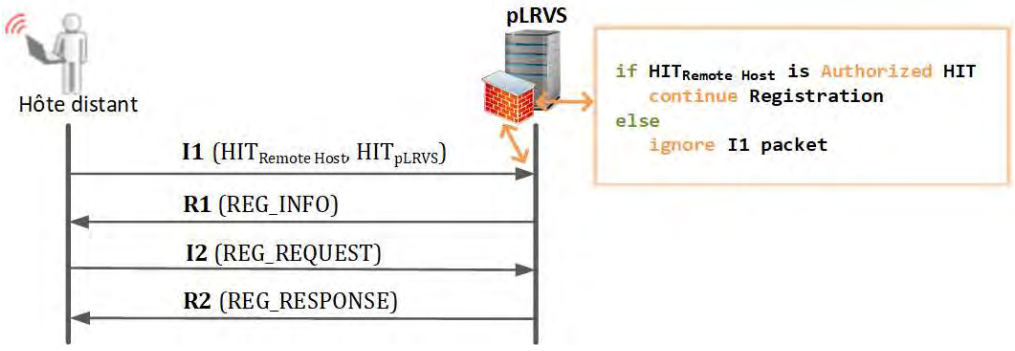


Figure 22 : Mécanisme d'enregistrement sécurisé proposé

Une fois l'enregistrement effectué avec succès, une seconde modification est apportée dans le mécanisme de relai du paquet I1. Dans le nouveau schéma RVS proposé, voir *Figure 23*, le relai du paquet I1 vers un LM, se fait si seulement si l'hôte distant s'est enregistré au préalable, et est autorisé à communiquer avec le

LM. Sinon le paquet I1 est rejeté. A la réception du paquet I1, le LM répond directement à l'Initiateur de ce paquet sans passer par le pLRVS. Une fois le Base Exchange établi entre le LM et l'hôte distant, leur communication se fait via un tunnel ESP (Encapsulating Security Payload).

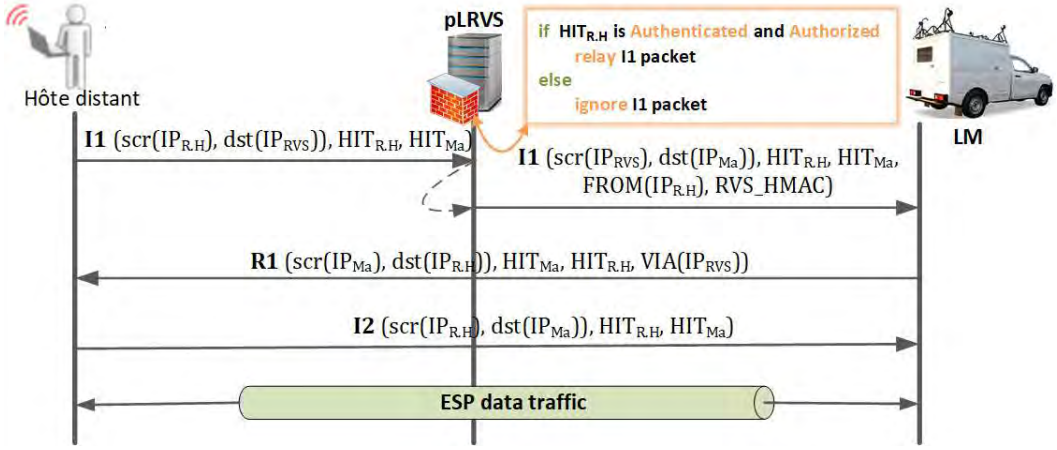


Figure 23 : Mécanisme de relai du paquet I1 proposé

La robustesse du mécanisme de sécurité proposé réside sur le fait qu'aucune information sur la localisation topologique des nœuds (LM) n'est fournie avant l'authentification et l'autorisation. Tout se joue au niveau du pLRVS, qui grâce à la robustesse du mécanisme de sécurité HIP [132] demeure une solution sûre pour la protection des données et des ressources du LM. L'adresse IP étant utilisée comme un simple localisateur, alors la mobilité des LM ne constitue pas une faiblesse dans le mécanisme de sécurité HIP. En effet, ce mécanisme repose sur le HIT qui demeure inchangé lors d'un changement de localisation. Néanmoins, pour garantir une continuité de service, des paquets UPDATE sont échangés entre le LM et ses pairs (hôtes distants). Ce processus entraine une latence qui peut engendrer des pertes de paquets. Ainsi, nous proposons un schéma de mobilité, nommé HIP$_{DISASS}$, capable à la fois de réduire la latence et les pertes de paquets lors d'un handover.

3.3. Schéma de mobilité proposé : HIP$_{DISSAS}$

Le HIP$_{DISASS}$ est un nouveau schéma de mobilité proactif supportant à la fois la micromobilité et la macromobilité.

3.3.1. Présentation de schéma HIP$_{DISASS}$

3.3.1.1. Description globale

Dès qu'un nœud mobile (dans notre cas le LM) est en situation de handover, il déclenche une procédure qui vise d'une part à connaitre son futur localisateur et d'autre part à savoir si cela va engendrer l'envoi ou pas de paquets UPDATE. Ce processus est déclenché avant que le LM ne se dissocie (se détache) de son point d'attachement courant, d'où le nom de HIP$_{DISASS}$.

En se référant à la *Figure 21*, après un établissement du Base Exchange (paquets 1) avec le LM, le pLRVS envoie au S-RVS concerné un message d'« assignation » (paquet 2) afin que le LM garde le même préfixe au niveau de tous les PoA gérés par ce S-RVS. Lorsqu'il reçoit un message de « réservation » (paquet 3 ou 5) de la part du LM contenant des informations du futur PoA, deux cas de figure se présentent au pLRVS. Si ce PoA est géré par le S-RVS courant (paquet 3, cas de micromobilité), le pLRVS envoie directement un message de « confirmation » (paquet 4) au LM pour lui signifier que son préfixe courant sera conservé. Dans le cas contraire (paquet 5, cas de macromobilité), le LM changera de préfixe réseau, car le futur PoA est géré par autre S-RVS. Le pLRVS enverra donc un paquet d'« assignation » (paquet 6) au S-RVS concerné en laissant vide le champ du préfixe réseau à assigner. Ce S-RVS assigne alors de manière dynamique un nouveau préfixe au LM puis envoie un message de « confirmation » (paquet 7) au pLRVS contenant le nouveau préfixe du LM. Celui-ci se chargera de relayer le message (paquet 8) au LM.

3.3.1.2. HIP_{DISASS} et la micromobilité

Dans le cadre de la micromobilité, voir *Figure 24*, le LM garde le même préfixe réseau lorsqu'il se lie à un nouveau PoA. A la réception d'un message de « confirmation », le LM s'aperçoit que son préfixe ne sera pas changé, et par conséquent n'échange aucun message UPDATE avec ses pairs (hôtes distants).

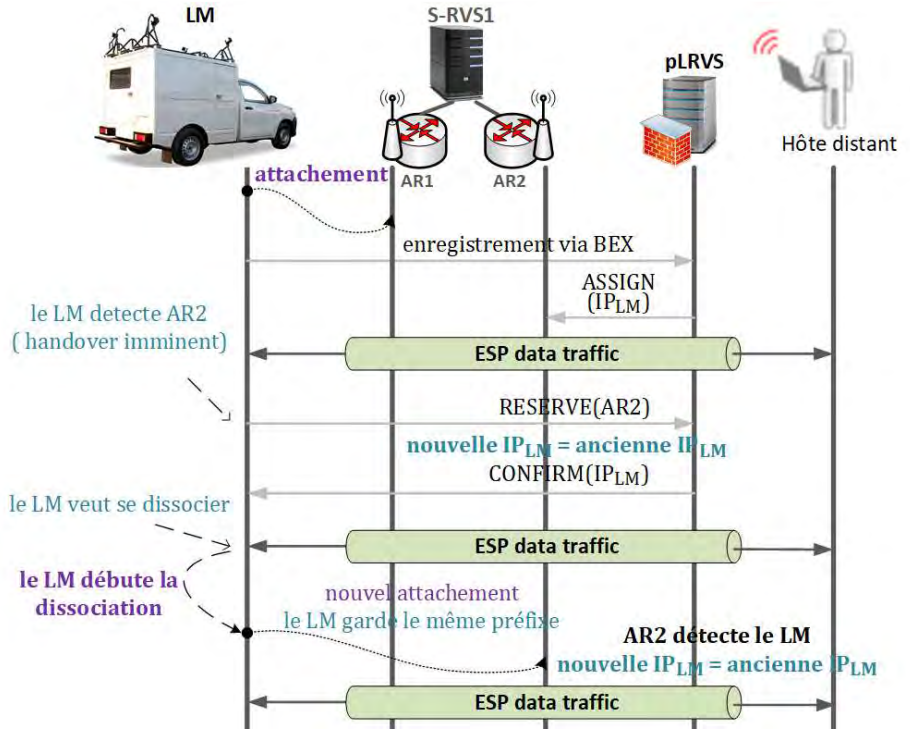


Figure 24 : Schéma de micromobilité dans HIP_{DISASS}

3.3.1.3. *HIP$_{DISASS}$ et la macromobilité*

Dans le cadre de la macromobilité, voir *Figure 25*, le LM change de préfixe réseau lorsqu'il se lie à un nouveau PoA. En effet, le nouveau PoA est géré par un autre S-RVS du domaine. A la réception d'un message de « confirmation », le LM s'aperçoit qu'il sera en situation de macromobilité lors du handover. Le LM, disposant de son futur préfixe, peut dès lors anticiper sur l'envoi du premier paquet UPDATE avant de se dissocier de son PoA courant.

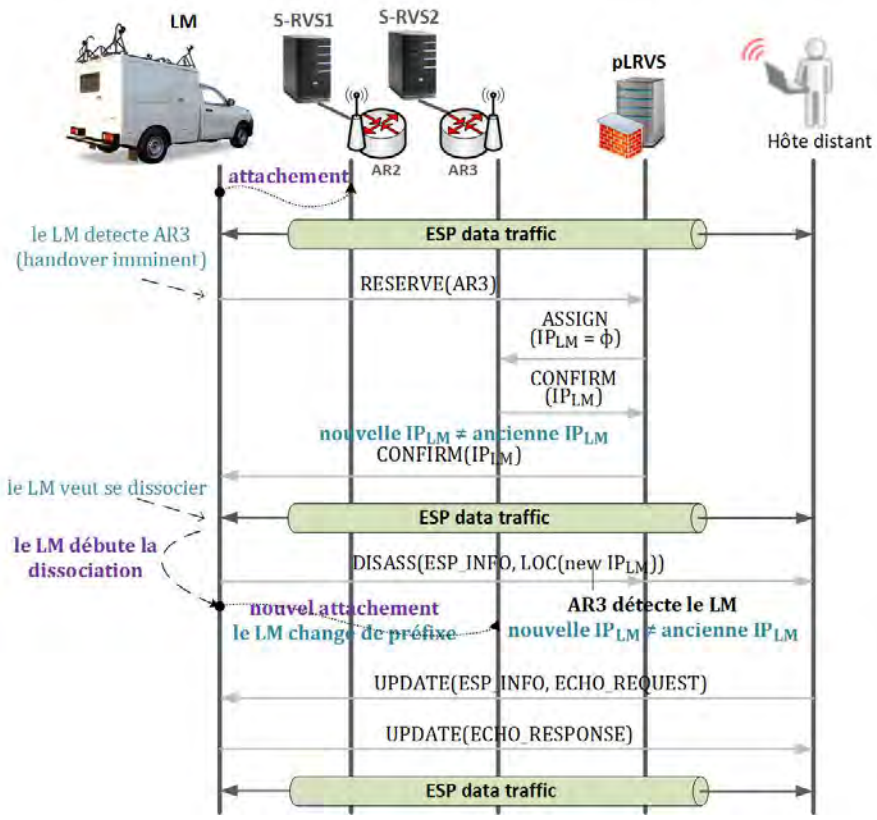


Figure 25 : Schéma de macromobilité dans HIP$_{DISASS}$

Le fait d'envoyer un message DISASS (paquet UPDATE anticipé) bien avant le changement de PoA vise à réduire la latence processus UPDATE. En effet, le processus d'attachement au niveau PoA et la configuration du nouveau préfixe réseau du LM se fait entre l'intervalle séparant l'envoi de message DISSAS et la réception du message UPDATE venant de l'hôte distant.

3.3.1.4. *Transport des paquets HIP$_{DISASS}$*

Les messages RESERVE, ASSIGN et CONFIRM seront envoyés par le biais de paquets NOTIFY natif de HIP. Ce choix se justifie pour deux raisons :

- Le paquet NOTIFY est léger, facile à traiter et on peut y insérer facilement des données additionnelles.
- Le paquet NOTIFY n'entraine pas de changement sur les associations HIP.

La *Figure 26* illustre l'utilisation de paquets NOTIFY lors d'une procédure de mobilité.

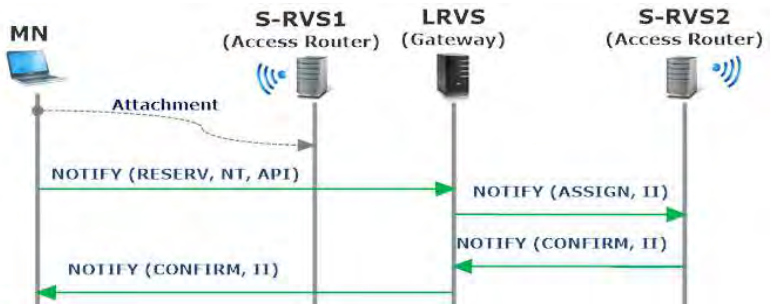


Figure 26 : Utilisation du paquet NOTIFY dans HIP$_{DISAS}$

- *Format du message RESERVE*

Le message RESERVE, voir *Tableau 5*, sert de messages de réservation et est destiné au pLRVS, mais peut être relayé au RVS dans certains cas.

Tableau 5 : Format message RESERVE

Notify Message Type	
RESERVE	
Node Type (NT)	
Source	Interface Mac Address
Attachment Point Information (API)	
BSSID	ESSID

Le premier champ à renseigner est le NT (Node Type). Il contient les paramètres source qui identifie le type de requête de réservation (2 s'il s'agit d'un relai) et l'adresse MAC du MN. Dans le champ API (Attachment Point Information), le BSSID et l'ESSID du futur PoA y sont renseignés.

- *Format du message ASSIGN*

Le message ASSIGN, voir *Tableau 6*, est envoyé lors de l'enregistrement du MN au niveau du pLRVS ou dans une situation de macro-mobilité.

Tableau 6 : Format message ASSIGN

Notify Message Type	
ASSIGN	
Interface Information (II)	
Interface MAC address	Reserved IP Address

Ce message a un seul champ nommé II (Interface Infos) qui renseigne deux paramètres, à savoir : l'adresse MAC du MN et l'adresse IP réservée.

- *Format du message CONFIRM*

Le message CONFIRM, voir *Tableau 7*, est envoyé par le S-RVS au pLRVS qui se chargera de le relayer au MN.

Tableau 7 : Format message CONFIRM

Notify Message Type	
CONFIRM	
Interface Information (II)	
Interface MAC address	Reserved IP Address

Dans ce message, le S-RVS renseigne l'adresse MAC et le préfixe réseau réservé au MN.

3.3.2. Evaluation analytique de la latence du handover

Ce modèle analytique s'inspire de celui de Muslam et al. [130] et permet de faire une comparaison entre les schémas de mobilité HIP, HIP_{MUSLAM} et HIP_{DISASS}.

3.3.2.1. Cas de la micromobilité

En se référant à la *Figure 24*, on définit :

- La latence, L_{MD}, due à la détection/sollicitation d'un lien d'accès au niveau couche IP.
- La latence, L_{AD}, due à la détection de l'attachement du LM au niveau du S-RVS.
- La latence, L_{IP_CONFIG}, due à la reconfiguration du nouveau localisateur.
- Les latences $L_{LUI\_LRVS}, L_{LU2\_LRVS}$ et L_{LU3_LRVS}, dues aux paquets UPDATE échangés entre le LM et le pLRVS.
- Les latences, L_{LU1} et L_{LU2}, dues aux paquets UPDATE échangés entre le S-RVS et le pLRVS.
- Les latences L_{LUI_CN}, L_{LU2_CN} et L_{LU3_CN}, dues aux paquets UPDATE échangés entre le LM et le nœud correspondant (hôte distant).

La latence du schéma de handover L_{HIP} (voir *Annexe 1*) devient :

$$L_{HIP} = L_{MD} + L_{IP\_CONFIG} + L_{LUI\_CN} + L_{LU2\_CN} + L_{LU3\_CN} \quad (1)$$

La latence du schéma de handover L_{MUSLAM} (voir *Annexe 1*) devient :

$$L_{MUSLAM} = L_{AD} + L_{LUI} + L_{LU2} \quad (2)$$

La latence du schéma de handover $L_{microDISASS}$ devient :

$$L_{microDISASS} = L_{AD} \quad (3)$$

Ces résultats prouvent bien l'avantage du schéma HIP_{DISASS} par rapport aux autres schémas de mobilité. En effet, lors du handover aucun paquet UPDATE n'est échangé : tout est fait à l'avance, c'est le principe de la mobilité proactive.

3.3.2.2. Cas de la macromobilité

En se référant à la *Figure 25*, la latence du handover $L_{macroDISASS}$ devient :

$$L_{macroDISASS} = L_{AD} + L_{IP\_CONFIG} + L_{LU2\_CN} + L_{LU3\_CN} \quad (4)$$

Par rapport à la latence L_{HIP}, la latence de $L_{macroDISASS}$ présente un gain de latence L_{LU1_CN} du fait que le premier message UPDATE LU1 est remplacé par le message *DISASS* envoyé de manière proactive.

3.4. Résultats expérimentaux

Les résultats expérimentaux serviront à étudier, d'une part, l'impact du protocole HIP niveau des applications et des infrastructures réseaux, et d'autre part, les performances des schémas de mobilité proposés.

3.4.1. Tests de performance du protocole HIP

Le but de cette section est de tester les performances du protocole HIP sur un ordinateur monocarte (SBC, Single Card Computer) de type Raspberry Pi model B+ (CPU : ARM 700 MHz, RAM : 512 MB, OS : Raspbian).

- *Durée du Base Exchange*

Le Base Exchange, *Figure 15*, consiste à l'échange de quatre paquets (handshake) entre deux nœuds nommés Initiateur et Répondeur. Le but de ce test est de déterminer le temps nécessaire pour établir une association HIP.

- *Influence de la difficulté du puzzle*

Pour mesurer l'impact de la difficulté K du challenge cryptographique sur la durée du Base Exchange, nous mesurons le temps $T2$ pour différentes valeurs de K. La résolution du puzzle se fait par la méthode brute force.

- *Influence du groupe Diffie-Hellman*

Comme pour le puzzle, nous appliquerons le même principe c'est-à-dire mesurer la durée $T2$ en fonction des groupes Diffie-Hellman

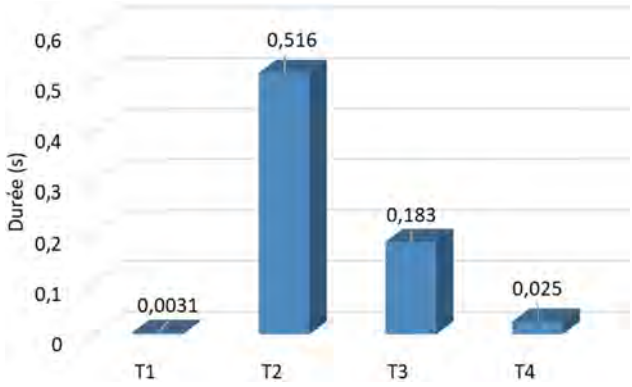
- *Mesure du RTT (Round Trip Time)*

Lors de son usage avec le protocole HIP, le paquet ICMP est encapsulé dans un entête ESP. Pour mesurer l'impact de cette encapsulation, nous déterminerons le RTT en effectuant des ping6 en utilisant le HIT et l'adresse IPv6 des hôtes.

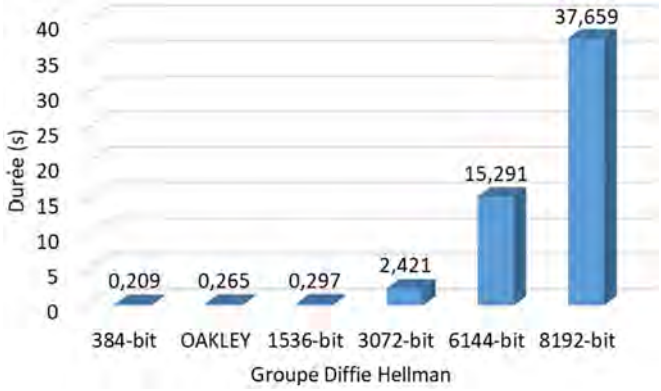
- *Mesure du débit*

Le protocole HIP modifie la structure des paquets traditionnels en les chiffrant via ESP, et en introduisant des informations de contrôle. Pour voir comment HIP affecte le débit de transmission, nous effectuerons des mesures via différent type de modèle : TCP/IP et HIP.

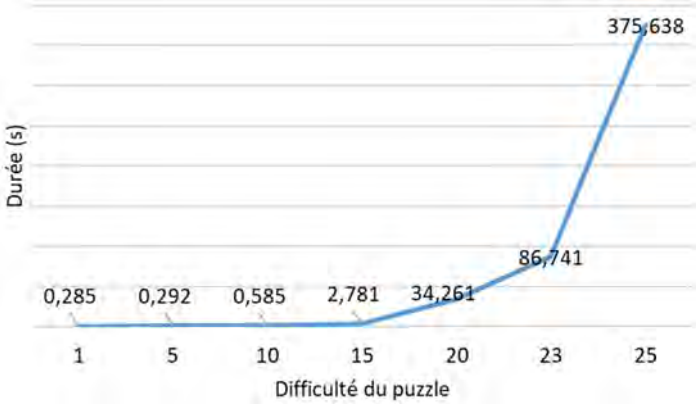
Les résultats de tests, voir *Figure 27*, montrent que la durée du Base Exchange est largement dominée par $T2$ et $T3$. Afin de mieux cerner les causes, nous proposons d'étudier les facteurs impactant sur $T2$.



a) Durée de Base Exchange



b) Délais de traitement T2 vs groupe de Diffie Hellman



c) Délais de traitement T2 vs difficulté du puzzle

Figure 27 : Performances du protocole HIP

En fonction des paramètres du DH (*Figure 27.b*) et du puzzle (*Figure 27.c*), on constate une évolution exponentielle de $T2$. Par conséquent, agir sur ces deux paramètres pourra réduire la latence du Base Exchange. En prenant les paramètres

par défauts (groupe $DH = OAKLEY$, difficulté puzzle $K = 10$), la latence du Base Exchange avoisine les 1000ms. Une fois la communication initialisée via le Base Exchange, un autre facteur à prendre en compte est l'introduction d'un entête ESP dans les paquets. Ainsi, comparée au TCP/IP, le RTT pour HIP augmente de 24% pour atteindre les 81,208 ms. Cela est due à l'augmentation de la taille des paquets causé de l'entête ESP et du temps de traitement de cet entête. Le débit relativement faible (TCP/IP : 591kbits/s ; HIP : 496kbits/s) peut constituer une entrave au bon fonctionnement de certaines applications. Quant à la procédure de mobilité, elle prend 228.89 ms. Cependant, pour les applications temps réels, une valeur beaucoup plus faible est préférable.

3.4.2. Tests de performance des schémas de mobilité proposés

Pour évaluer les performances du schéma de mobilité HIP_{DISASS}, nous avons apporté des modifications à OpenHIP afin qu'il supporte HIP_{DISASS}. La variable n correspond au nombre de nœuds communiquant avec le LM juste avant le handover. La procédure de handover est initiée lorsque le RSSI du second PoA détecté par le LM est supérieure au RSSI de son PoA courant. Une valeur de hystérésis est fixée à 5s afin d'éviter l'effet « ping pong ».

Le *Tableau 8* montre les résultats des tests de performance obtenus dans un scénario de micromobilité.

Tableau 8 : Performances des schémas de micromobilité HIP et HIP_{DISASS}

Schéma de mobilité	Latence (s)	Pertes de paquets	Nombre de paquets de signalisation échangés avant le Handover	Nombre de paquets de signalisation échangés durant le Handover
HIP	6,18	16%	0	$3n$
HIP_{DISASS}	3,5	10%	3	0

Le *Tableau 9* montre les résultats des tests de performance obtenus dans un scénario de micromobilité.

Tableau 9 : Performances des schémas de macromobilité HIP et HIP_{DISASS}

Schéma de mobilité	Latence (s)	Pertes de paquets	Nombre de paquets de signalisation échangés avant Handover	Nombre de paquets de signalisation échangés durant le Handover
HIP	10,88	19%	0	$3n$
HIP_{DISASS}	7,82	17%	$3 + n$	$2n$

Les résultats expérimentaux confirment les résultats de l'évaluation analytique quant aux meilleures performances du schéma HIP_{DISASS}, par rapport aux autres schémas de mobilité. On note également une élimination ou une réduction de la charge réseau liée à la signalisation HIP durant le handover.

Les valeurs relativement élevées des latences sont essentiellement dues aux performances limitées des équipements de tests. En effet, des machines virtuelles partageant avec d'autres entités de test la même carte réseau. Cela génère de forte latence durant l'attachement des nœuds au PoA. Une partie des pertes de paquets est également due à la fenêtre anti-rejeu ESP qui rejette les paquets pour lesquelles les SAs sont expirées.

3.4.3. Evaluation du HIP$_{DISASS}$

Niveau infrastructures réseaux, le protocole HIP peut être utilisé aisément et de manière convenable. Cependant, HIP peut entrainer dans certains scénarios, une utilisation relativement élevée en ressource de calcul. En effet, HIP utilise des opérations cryptographiques de manière intensive, par conséquent consomme beaucoup de ressources CPU. Dans les scénarios où l'infrastructure réseau est menée à gérer plusieurs associations HIP avec plusieurs nœuds, une surcharge de ce dernier peut vite arriver. Ce cas peut arriver au pLRVS lorsque le domaine est constitué de plusieurs LM ou lorsque plusieurs hôtes distants s'enregistrent au domaine.

Niveau LM, lorsque le véhicule est impliqué dans plusieurs associations HIP avec des applications/utilisateurs distants, le SBC risque de ne plus supporter la surcharge pour les mêmes raisons que les infrastructures réseau. Certaines applications du LM exigent une latence très faible, tandis que certaines applications exigent une garantie en termes de sécurité et de continuité de service en cas de handover. Or, HIP induit une latence dans la communication des hôtes. Par conséquent, il ne pourra pas être déployé sur l'ensemble du réseau véhiculaire, notamment au niveau de la communication V2V caractérisée par des applications critiques et à très fortes sensibilités temporelles.

En outre, le modèle de communication de HIP est incompatible avec les besoins applicatifs et de communications des Laboratoires Mobiles de Nouvelles Générations. En effet, dans le modèle de LM basé sur HIP, le véhicule est considéré comme une « mule » [11] servant à transporter des dispositifs de relève et de communication. Or, avec l'avènement des véhicules connectés, les dispositifs de relève (capteurs) et de communication font partie intégrante du fonctionnement de ces derniers. Cela permet aux véhicules d'accéder, de consommer, de créer, d'enrichir, de diriger et partager l'information entre eux, les individus, les organismes, les milieux d'affaires... Or le modèle de communication HIP n'est ni conçu, ni optimisé pour cela (communication V2X). Il est donc nécessaire de trouver un nouveau modèle de communication et de réseau plus adapté pour les Laboratoires Mobiles de Nouvelles Générations.

3.5. Conclusion

Dans ce chapitre, un ordinateur monocarte a été utilisé comme ordinateur de bord (OBU) du Laboratoire Mobile (LM) responsable de la collecte de données de capteurs. Des mécanismes de sécurité et des schémas de mobilité basés sur le protocole HIP ont été proposés afin de gérer les besoins de communication du LM. Les tests de performance montrent qu'un ordinateur monocarte peut être utilisé ordinateur de bord pour le LM à condition que certains paramètres de sécurité tels que la difficulté du puzzle et le groupe DH soient ajustés. Cependant, l'évaluation des performances a montré que l'utilisation intensive d'opérations cryptographique limite l'évolutivité (scalabilté) de l'architecture HIP_{DISASS} proposé. De même, le modèle de communication de HIP présente des limites en ce qui concernent les besoins applicatifs et de communication des véhicules connectés.

Le but du chapitre suivant est de proposer un nouveau modèle de LM exploitant la grande capacité de perception et de communication des véhicules connectés. Un nouveau modèle de communication beaucoup plus élaboré pour le LM sera également proposé.

Chapitre 4 : Proposition d'un modèle de LM à base d'un véhicule vu comme une plateforme communicante intelligente

Ce chapitre propose un nouveau modèle de Laboratoire Mobile (LM) où le véhicule est doté de capacités de traitement et de raisonnement sur les données. Deux adaptations du modèle Sensing as a Service (S^2aaS) sont proposées afin d'exploiter pleinement les capacités du LM. Ensuite, un exemple d'application pratique du S^2aaS est présenté. Cette application permet de choisir le meilleur itinéraire en prenant pour critère la qualité de l'air. Enfin, un guide d'implémentation d'une partie des modèles S^2aaS sur Android Automotive est proposé.

4.1. Modèle de Réseau de Capteurs Véhiculaire Intelligent

Un bon système de capteurs est un gage d'une bonne intégration des fonctions d'intelligence dans les véhicules. Dans [17], les auteurs posent les principaux défis à relever dans les réseaux de capteurs intelligents (ISN, Intelligent Sensors Network). Ainsi, l'intelligence des réseaux de capteurs peut être atteinte de quatre façons : *sensibilité spatiale, sensibilité de donnée, sensibilité de groupe, et sensibilité de contexte*.

En psychologie, le concept de *sense of place* [133] ou *sentiment d'appartenance* fait référence à la perception individuelle et sociale concernant la présence humaine à un endroit et à un moment donné. Ce concept se modélise via trois entités, à savoir : *Self, Others* et *Environment*. L'entité *Self* fait référence au lien personnel d'un individus avec un lieu, tandis que *Others* englobe les caractéristiques perçues et le comportement des habitants. Enfin, l'entité *Environment* est constituée des caractéristiques physiques de l'environnement. Le modèle proposé dans [18] permet d'extraire cette sensibilité spatiale de manière autonome et dynamique en analysant des données collectives librement accessibles via les réseaux sociaux.

Nous proposons de modéliser un Réseau de Capteurs Véhiculaire Intelligent (IVSN, Intelligent Vehicular Sensor Network) à partir des deux concepts suscités. Ainsi, les définitions des types de sensibilité développés pour l'ISN [17] seront modifiées afin de les adapter au contexte des véhicules connectés. Ensuite, la sensibilité spatiale sera modélisée par le biais de trois entités, à savoir : *Self, Others* et *Infrastructure*.

4.1.1. Modèle fonctionnel du IVSN

La *Figure 28* représente le modèle du IVSN proposé. Ce modèle est constitué de trois entités, à savoir :

- *Self :* matérialise un unique véhicule avec son propre système de capteurs, sa propre unité de traitement et ses propres interfaces de communication.
- *Others :* matérialise l'ensemble des véhicules se trouvant au voisinage de l'entité *Self.*
- *Infrastructure :* c'est l'agrégateur de données, et une fois ces données transformées en information, elle les renvoie aux entités *Self* et *Others*.

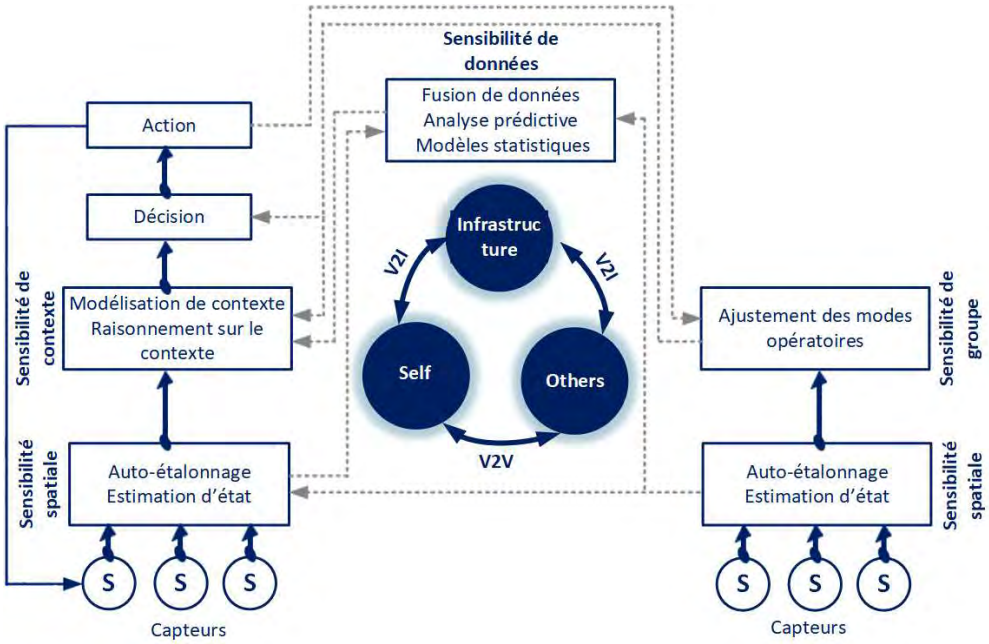


Figure 28 : Modèle fonctionnel du IVSN proposé

Le *Self* développe un processus cognitif en s'aidant des données issues de ses propres capteurs et des capteurs des véhicules environnants matérialisés par l'entité *Others*. En effet, grâce à la sensibilité spatiale, le *Self* est en mesure de connaitre la localisation physique des véhicules et infrastructures environnants. Cette sensibilité permet d'auto-étalonner les capteurs de *Self* via la prise en compte des mesures issues des capteurs voisins. Ainsi, le *Self* sera capable de savoir si ses capteurs effectuent des mesures correctes, de valider les mesures effectuées par les capteurs des voitures environnants, de pouvoir faire des estimations ou prédictions à des points où l'on note l'absence de capteurs… Avec la *sensibilité de groupe*, le *Self* et *Others* pourront extraire et disséminer des informations liées au mouvement des véhicules environnants, savoir leurs états et par conséquent ajuster leurs propres modes opératoires afin maintenir, par exemple, la connectivité du groupe dans diverses conditions.

Contrairement à *Self* et *Others* qui sont de nature éphémère dans le IVSN, l'entité *Infrastructure* est omniprésente et est considéré comme la « mémoire » de l'IVSN dans le sens où elle peut utiliser des données issues de véhicules ayant récemment constitués les entités *Self* et *Others*. Grâce à la *sensibilité de données*, l'*Infrastructure* peut extraire des informations de hautes qualités en explorant la corrélation spatiale et temporelle des données, de prédire et construire des modèles statistiques. Ainsi le *Self* pourra, à partir des données issues de ses propres capteurs et ceux de *Others*, prendre une décision et par conséquent agir de la meilleure façon possible. Finalement, grâce à la *sensibilité de contexte*, le *Self* pourra collecter et traiter les informations de contexte issues de ses capteurs, et modifier son mode opératoire en se basant sur ce contexte.

D'après la description du modèle proposé, les architectures de type Vehicular Clouds ou Hybrid Vehicular Clouds [58] seront les mieux adaptées. Les échanges au niveau du cloud local se feront via la communication V2V. Ainsi, les entités *Self* et *Others* pourront échanger et traiter leurs données localement. L'entité *Infrastructure* peut être matérialisée par une RSU, est sera accessible par le biais d'une communication V2I. La RSU sera chargée de collecter les données brutes des véhicules afin de les traiter puis les redistribuer à ces derniers, ou bien de les transmettre vers un serveur cloud ou au middleware.

4.1.2. Pistes d'implémentation

Dans la littérature, plusieurs techniques sont utilisées pour transformer les données brutes des capteurs en connaissance (high level information). Ainsi dans [17], les auteurs proposent des techniques basées sur le machine learning pour formater, représenter, modéliser, raisonner sur les données issues des ISN. Dans [134], une méthode d'auto-étalonnage de capteurs intelligents basée sur les réseaux de neurones artificielles est proposée. Cette méthode permet de résoudre les problèmes liés au seuil, à la variation du gain et à la non-linéarité des mesures. Des méthodes d'analyse géostatistiques sont proposées dans [135] pour prédire la valeur de la température en cas de non disponibilité de mesures sur l'endroit cible.

D'autres travaux de recherche se focalisent sur la fusion de données. Dans [136], une méthode basée sur le partitionnement est proposée pour la fusion de données multidimensionnelles (multi-sources, hétérogènes, dynamiques, clairsemés). L'algorithme proposé permet de réduire la complexité spatiale et temporelle. Dans [137], les auteurs proposent une méthode générique de détection d'événement dangereux sur la route en se basant sur la fusion de données distribuées. Les auteurs utilisent la théorie de Dempster-Shafer pour améliorer la robustesse de leur algorithme face aux erreurs de mesure et à la dynamique des VANET. De manière générale, les méthodes d'estimation (filtre de Kalman, méthodes de covariance), de classification (Machine à vecteurs de support, réseaux de neurones artificielles, clustering) et d'inférence (inférence bayesienne, logique

floue, théorie de Dempster-Shafer) sont les plus employées pour la fusion de données issues de capteurs multiples [138].

On retrouve également des travaux de recherche qui traitent des problèmes liés au comportement, influence, état et dynamique d'un groupe. Dans [139], les auteurs utilisent des informations relatives au mouvement des nœuds (détecteur d'inclinaison, accéléromètre) pour la construction d'une dynamique de groupe. Un algorithme de corrélation incrémentale est utilisé afin de décider si un nœud appartient ou pas au groupe. Leurs travaux présentent des applications intéressantes notamment le contrôle d'une chaine de distribution dans le domaine des transports et logistiques. L'approche de la théorie du jeu est proposé dans [140] pour fournir des services de santé à la volée. Le jeu est disputé entre plusieurs véhicules qui priorisent l'accès des services de soins aux patients suivant la valeur du gain (état du patient). Dans [60], les auteurs proposent un algorithme basé sur le *crowd sensing* pour la découverte de routes moins congestionnées.

La sensibilité de contexte est le domaine de recherche suscitant le plus de challenge, d'intérêts, d'applications et de productions scientifiques. On peut retrouver à travers plusieurs papiers [141], [142], [143] une revue complète des différents travaux traitant de ce sujet. Dans [141], une taxonomie des techniques d'acquisition, de modélisation, de raisonnement et de distribution de contexte est présentée. Les techniques de raisonnement reposent essentiellement sur l'apprentissage supervisé et non supervisé, la logique floue, l'ontologie, la logique probabiliste. Dans [142], les auteurs classifient les applications suivant trois dimensions incluant l'environnement (urbain, rural, autoroute…), système/application (type de service, architecture, communication), sensibilité de contexte (type de contexte, méthode de collecte). Les auteurs font également une présentation de l'ensembles des projets saillants relatives à la sensibilité de contexte dans les VANET.

4.1.3. Application: Sensing as a Service (S^2aaS)

De manière générale il y a trois acteurs dans l'architecture du S^2aaS : le fournisseur de données brutes (raw data provider), le fournisseur de service (network, middleware, applications providers) et le consommateur. Les données suivent les transformations suivantes : on passe de données brutes (symboles) en information (réponse aux questions : Qui ? Quoi ? Où ? Quand ?), puis de l'information en connaissance (réponse à la question : Comment ?) ensuite de connaissance à la compréhension (appréciation du : Pourquoi ?) et enfin on passe de la compréhension au savoir (évaluation de la connaissance).

Le modèle S^2aaS proposé, *Figure 29*, est une architecture de type Hybrid Vehicular Clouds. Contrairement aux modèles proposés dans la littérature qui s'intéressent aux smartphones [21] ou des sources variées de données [61], [62], [64], notre modèle s'intéresse exclusivement aux véhicules. Par conséquent, sa conception est basée sur le modèle de communication des véhicules connectés

(communication V2X) et sur l'IVSN. Les capteurs incorporés dans les véhicules seront les sources de données, la RSU sera chargée de la collecte puis de la fusion de données, le middleware représente le cœur du S²aaS et est chargé de la transformation de l'information en connaissance. Un fournisseur d'applications sera chargé de proposer une variété d'applications aux consommateurs afin de leurs permettre de comprendre, de tirer du savoir et des profits à partir des données traitées par le S²aaS. L'architecture du S²aaS est constituée de plusieurs couches, à savoir :

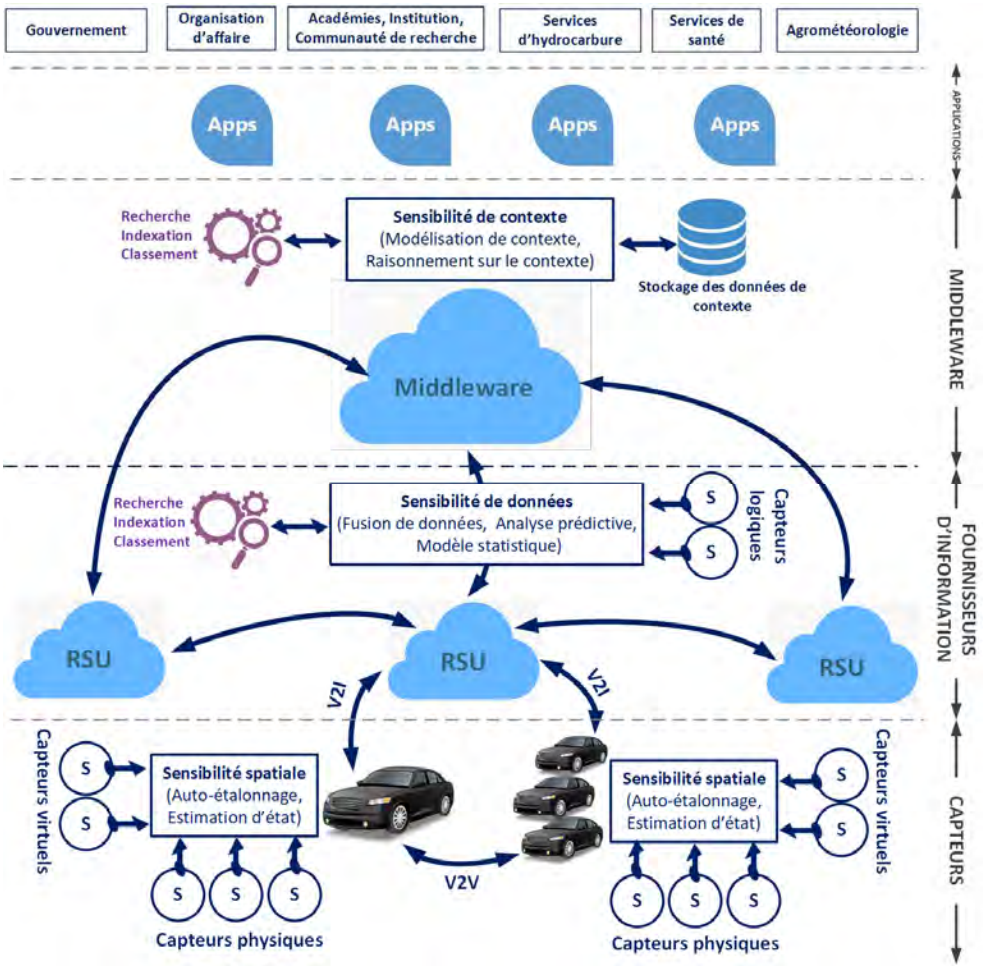


Figure 29 : Architecture globale du modèle S²aaS proposé

- *Les couches basses :* héritent des fonctionnalités des IVSN et constituent donc le cloud local.
- *Les couches hautes :* se situent au niveau du cloud conventionnel et sont accessibles via Internet.

Le première couche basse est la *couche des capteurs* constituée de véhicules fournissant des données brutes issues de capteurs physiques et virtuels. Cette couche utilise la sensibilité spatiale afin de vérifier la validité des mesures et

éventuellement procéder à l'auto-étalonnage des capteurs. Les données brutes seront donc moins assujetties à contenir des valeurs erronées.

La seconde couche basse est le *fournisseur d'informations* constitué essentiellement de RSU. Ces derniers sont chargés de faire la collecte de données brutes via les méthodes push et pull, puis de procéder à un traitement de ces données en utilisant principalement des techniques de fusion de données. La RSU est alors considéré comme un capteur logique dans le sens où il fournit une information beaucoup plus pertinente. La RSU détient des capacités de stockage lui permettant de stocker une certaine quantité de données durant un certain bout de temps avant un stockage définitif dans le cloud. Au besoin, les RSU peuvent communiquer entre eux afin d'échanger des données brutes ou des informations.

La première couche haute est le *middleware*. Cette couche est la plus importante du S^2aaS, car la majorité des fonctionnalités y sont implémentées. En effet la gestion des autorisations et l'orchestration de tous les services s'effectuent à ce niveau. Le middleware doit avoir au moins les fonctionnalités suivantes : (1) coordination de la recherche, indexation et classement des informations ; (2) acquisition, modélisation et raisonnement du contexte ; (3) analyse des données pour le business intelligence ; (4) stockage des données de contextes, (5) interopérabilité. Le middleware n'a accès qu'aux capteurs logiques c'est-à-dire ne peut accéder aux capteurs physiques et virtuels. En effet, il n'existe pas une communication directe entre le middleware et les véhicules. La RSU jouera donc le rôle d'intermédiaire entre le middleware et la couche des capteurs. Suivant l'information demandée par le middleware, la RSU sera chargée de collecter les données brutes nécessaires.

La seconde couche haute est la *couche des applications* qui est l'interface entre le client et le middleware. Suivant les actions du client, l'application enverra toutes les requêtes nécessaires au middleware. Dans certaines situations, l'application pourra agir de manière proactive sans attendre des actions du client. Les connaissances ainsi obtenues à partir du middleware permettront à l'application d'aider le client à la compréhension d'un sujet ou d'un phénomène

4.2. Modèle Sensing as a Service : première proposition

Dans cette première proposition de modélisation du S^2aaS, deux types de modèles seront utilisés : le modèle formel et le modèle fonctionnel. Ce modèle s'approche essentiellement de la logique des modèles proposés par Perera et al. [35] et Abdelwahab et al. [64].

4.2.1. Modèle fonctionnel

Dans le modèle proposé, voir *Figure 30*, deux processus se déroulent en parallèle, à savoir : la collecte des propriétés de contexte des capteurs automobiles et le traitement des requêtes des clients du S^2aaS.

- *Collecte des propriétés de contexte*

Chaque véhicule publie de façon périodique les propriétés de contexte de ses capteurs incorporés vers une RSU. A la réception de chaque publication, la RSU met à jour sa table de propriétés de contexte stockée dans une base de données CPDB (Context Proprieties DataBase). Vu la dynamique des VANET, la durée de validité des propriétés de contexte demeure très court. Par conséquent, si une RSU ne reçoit pas la publication de propriétés de contexte d'un véhicule durant une certaine période, l'entrée correspondante est supprimée de la CPDB.

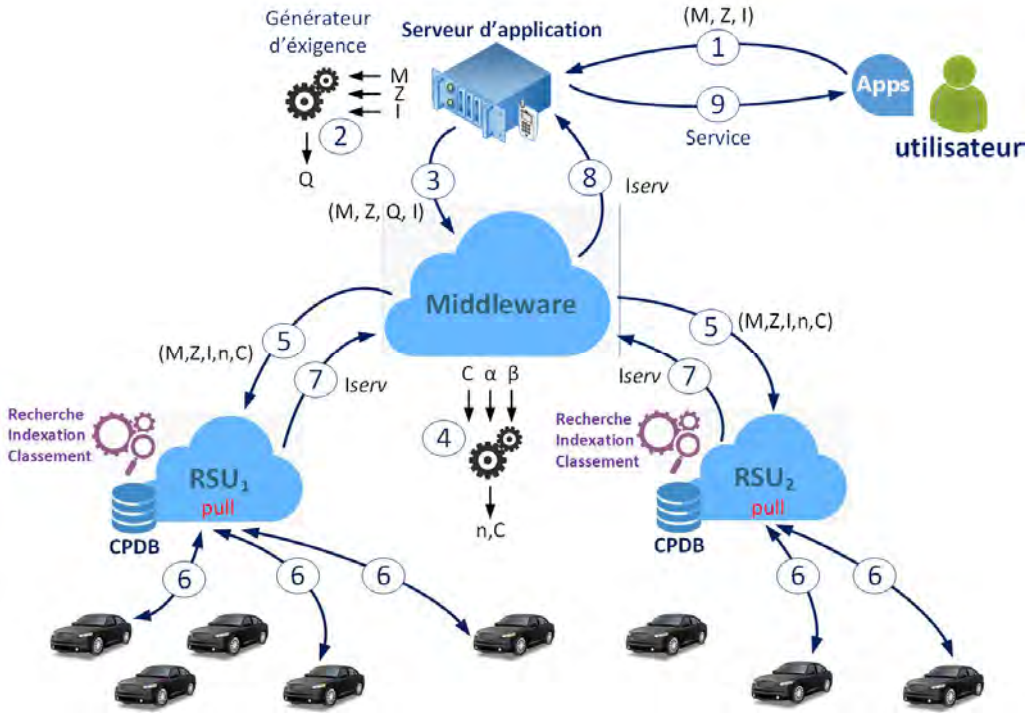


Figure 30 : Principe de fonctionnent du S²aaS : première proposition

- *Traitement des requêtes du client S²aaS*

Le client utilise une application pour accéder à un service. Cette application fournit une interface permettant au client d'indiquer les types de mesures souhaitées M, la zone de relève Z et le traitement associé à ces mesures I ①. Cette requête est traitée par un serveur d'application au niveau de la couche des applications. Ce traitement ② consiste à générer est un ensemble de critères de qualité Q, définis par le serveur d'application et le middleware, afin d'aider l'utilisateur, sans connaissance avancée sur les technologies de capteurs, d'utiliser les services du S²aaS. L'étape suivante consiste à envoyer une requête de mesure contenant les paramètres M, Z, I et Q au middleware ③. A la réception de cette requête, le middleware détermine le nombre optimal n de capteurs et les valeurs optimales des propriétés de contexte C à utiliser ④. Ensuite, le middleware sollicite l'ensemble des RSU présentes dans la zone Z pour collecter les mesures demandées ⑤. En

fonction de C et de M, chaque RSU détermine d'abord les capteurs éligibles, puis fait un classement et une sélection des n meilleurs capteurs. Une requête par la méthode pull sera envoyée par la RSU vers l'ensemble des véhicules incorporant au moins un capteur sélectionné ⑥. Une fois les réponses reçues, la RSU traite ces données brutes pour les transformer en informations I_{serv}. La RSU est alors en mesure de répondre ⑦ à la requête tâche de capture envoyée plus tôt par le middleware. Ce dernier est chargé de transférer I_{serv} au serveur d'application ⑧, qui à son tour pourra fournir le service demandé par le client ⑨.

4.2.2. Modèle formel

Ce modèle formel décrit le processus de demande, de traitement et de fourniture de services.

- *Détermination de la tâche de capture*

Cette tâche consiste à décrire une zone d'intérêt Z, un ensemble de mesures M, les informations recherchées I et la qualité de mesure Q. Soient :

$M = \{M_1, M_2, \dots , M_i, \dots\}$ l'ensemble des mesures, où M_i un type de mesure telle que la température, l'humidité, la concentration de gaz…

$Q = \{Q_1, Q_2, \dots, Q_i, \dots\}$ l'ensemble des qualités de mesure pour M, où Q_i exprime un ensemble de qualités requises pour chaque capteur devant effectuer une mesure M_i.

$Q_i = \{q_{i2}, q_{i2}, \dots, q_{ij}, \dots\}$ un ensemble de qualités requises, où q_{ij} est la valeur seuil de qualité d'une propriété de contexte.

I représente les informations recherchées telles que des résultats statistiques, des données analysées pour le business intelligence...

- *La requête tâche de capture*

Cette requête est envoyée par le middleware à la RSU. Elle contient les paramètres n, C, M et I.

Soient :

$CP = \{cp_1, cp_2, \dots, cp_i, \dots\}$ l'ensemble des propriétés de contexte.

$C = \{C_1, C_2, \dots, C_i, \dots\}$ l'ensemble des valeurs optimales des propriétés de contexte, où C_i correspond à la valeur optimale de la propriété cp_i.

n le nombre optimal de capteurs devant effectuer la mesure demandée.

n_0 le nombre seuil minimal de capteurs devant effectuer la mesure demandée.

α_i un poids indiquant l'importance relative d'une propriété de contexte cp_i sur le coût de mesure.

β_i un poids indiquant l'importance relative d'une propriété de contexte cp_i sur la qualité de mesure.

α_0 un poids indiquant l'importance relative de n sur le coût de mesure.

β_0 un poids indiquant l'importance relative de n sur la qualité de mesure.

Les valeurs de n et de C_i sont obtenues par la résolution d'un problème d'optimisation. Le but étant de minimiser le coût de la mesure (charge réseau, risque de ne pas trouver un capteur éligible) tout en maximisant sa qualité. La formulation standard du problème de programmation multiobectif peut s'écrire comme suit :

$$\begin{aligned} &Minimize \quad \alpha_0 n + \sum_{i \in CP} \alpha_i C_i \\ &Maximize \quad \beta_0 n + \sum_{i \in CP} \beta_i C_i \\ &S.C \\ &\quad \begin{cases} n \geq n_0 \\ C_i \geq q_i \end{cases} \end{aligned} \tag{5}$$

- *La tâche de classement*

Cette tâche est exécutée par la RSU, et consiste à classer les capteurs suivant les paramètres indiqués dans la requête *tâche de capture*. Soient :

$S = \{S_1, S_2, \dots, S_i, \dots\}$ l'ensemble des capteurs dont leurs propriétés de contexte sont stockées dans la CPDB de la RSU.

$S' = \{S'_1, S'_2, \dots, S'_j, \dots\}$ est un sous-ensemble de S contenant l'ensemble des capteurs éligibles.

$M_{S'_j} = f(S'_j)$ l'ensemble des mesures que peut effectuer le capteur S'_j.

On définira un capteur éligible comme étant un capteur respectant les contraintes suivantes :

$$S'_j \in S' \; si \begin{cases} M_{S'_j} \cap M \neq \emptyset \quad et \\ C_i^{S'_j} \geq q_i \end{cases} \tag{6}$$

Où $C_i^{S'_j}$ est la valeur de la propriété de contexte C_i relative au capteur S'_j.

L'avantage d'identifier les capteurs éligibles réside dans la sauvegarde des ressources de calcul et dans l'accélération du processus de classement.

Perera et al. [35] ont développés une technique d'indexation basée sur la distance Euclidienne pondérée appelée CPWI (Comparative Priority Based Weighted Index). Le CPWI est calculé pour chaque capteur éligible S'_j. Dans notre modèle, nous définissons CPWI comme suit :

$$CPWI = \sqrt{\sum_{i \in CP} \left[W_i \left(C_i - C_i^{S'_j} \right)^2 \right]} \quad (7)$$

W_i est le facteur de pondération normalisée pour le contexte de propriété C_i.

$$W_i = \frac{1}{2} \left(\frac{\alpha_i}{\sqrt{\sum_{j \in CP} \alpha_j^2}} + \frac{\beta_i}{\sqrt{\sum_{j \in CP} \beta_j^2}} \right) \quad (8)$$

Ainsi, le classement des capteurs se fera dans l'ordre décroissant de CPWI. Pour la sélection des capteurs, les n premiers capteurs dans le classement sont sélectionnés. Si $\dim S' \leq n$, alors tous les capteurs sont sélectionnés.

- *La requête pull*

Cette requête est envoyée par la RSU vers l'ensemble des véhicules incorporant au moins un capteur sélectionné. La méthode pull est une méthode utilisée pour récupérer les données de capteurs à la demande.

4.2.3. Optimisations réseau

Afin d'effectuer la collecte de données, la RSU envoie des requêtes pull à un ensemble de véhicules. Cette procédure n'est pas très viable du fait que la communication véhiculaire est caractérisée par un fort taux de perte de paquets, et en plus, et le besoin de réduction de la charge réseau demeure récurant. Dans ce qui suit, nous proposons quelques méthodes d'optimisation réseau pour le S^{2}aaS.

4.2.3.1. Méthode 1 : évitement des pertes de paquets

Cette méthode consiste à envoyer une requête pull vers un véhicule lorsque la probabilité pour que ce dernier reste dans la zone de couverture de la RSU est supérieure à une certaine valeur seuil. Cette valeur peut être exprimée sous forme de quantité de temps ξ nécessaire pour l'envoi de la requête pull, de son traitement par le véhicule et de l'envoi de la réponse à la RSU. En se basant sur des modèles de mobilité de véhicules, il est possible de prédire l'instant où chaque véhicule quittera la zone de couverture de la RSU. Le véhicule devra ainsi, en plus de l'envoi périodique des propriétés de contexte, envoyer sa position p, sa vitesse v et sa prochaine intersection n_I. Doté de ces paramètres, la RSU sera en mesure de déterminer l'instant ξ_I où le véhicule atteindra l'intersection hors zone de couverture de la RSU. Ainsi :

$$S'_j \in S' \; si \begin{cases} M_{S'_j} \cap M \neq \emptyset \quad et \\ C_i^{S'_j} \geq q_i \quad et \\ \xi_I^{S'_j} \geq \xi \end{cases} \quad (9)$$

4.2.3.2. Méthode 2 : mise en cache des données de capture

Afin de minimiser la charge réseau, des techniques de mise en cache sont le plus souvent employées. La méthode proposée consiste à stocker les mesures des capteurs durant une certaine durée de validité. Ainsi, lorsqu'un capteur est sélectionné, la RSU vérifie d'abord si les mesures de ce capteur sont présentes dans sa cache et que le véhicule embarquant ce capteur est toujours dans sa zone de couverture. Si c'est le cas, la RSU n'aura plus besoin d'envoyer une requête pull, mais utilisera plutôt les données en cache du capteur. Ainsi :

$$S'_j \in S' \; si \begin{cases} M_{S'_j} \cap M \neq \emptyset \;\; et \\ C_i^{S'_j} \geq q_i \;\; et \\ et \; S'_j \; est \; dans \; la \; zone \; de \; couverture \; de \; la \; RSU \end{cases} \quad (10)$$

4.2.3.3. Méthode 3 : capteurs zombies

Cette méthode est une extension de la *méthode 2* et consiste à ajouter des capteurs zombies dans le réseau. Ces derniers sont des capteurs dont les véhicules les embarquant ont quittés la zone de couverture de la RSU, mais dont les mesures demeurent toujours stockées dans la cache de la RSU et en même temps ont une entrée dans la CPDB de la RSU. Ainsi :

$$S'_j \in S' \; si \begin{cases} M_{S'_j} \cap M \neq \emptyset \;\; et \\ C_i^{S'_j} \geq q_i \;\; ou \\ S'_j \; est \; un \; capteur \; zombie \end{cases} \quad (11)$$

Cette méthode est moins contraignante que la *méthode 2*, car n'exige plus la présence des véhicules dans la zone de couverture de la RSU.

4.2.3.4. Méthode 4 : méthode1 + méthode3

Cette méthode combine les techniques d'évitement de perte de paquets (*méthode 1*) et de capteurs zombies (*méthode 3*). Le but étant de réduire simultanément les pertes de paquets et la charge réseau.

4.2.4. Etude de performance des méthodes d'optimisation proposées

4.2.4.1. Description de la simulation

L'étude de performance se fera par le biais d'une simulation sous Python, et consiste à générer une carte, à y insertion des RSU et des véhicules. Seul la communication V2I est considérée dans cette simulation. L'aire d'étude est un carré (*6km* x *6km*) divisé en neuf zones de dimension (*2km* x *2km*) comportant chacune une RSU au centre. Les véhicules sont insérés au hasard sur une intersection de rue et peuvent choisir aléatoirement la vitesse et la direction à emprunter. Celles-ci peuvent changer aux intersections suivantes qui sont chacune distantes de *250m*. Le *Tableau 10* donne l'ensemble des paramètres de simulation.

Pour le classement des capteurs, les propriétés de contexte suivantes sont utilisées : *mesures supportées, précision, temps de réponse, intervalle de mesure, résolution*. L'ensemble M contient les mesures suivantes : *température, humidité, pression atmosphérique, luminosité, concentration de gaz et de particules*.

Une série de 10 requêtes tâche de capture ont été effectuées. Les résultats obtenus représentent les moyennes des valeurs de performances recueillies à partir des différentes RSU impliquées dans la simulation.

Tableau 10 : Paramètres de simulation du modèle S²aaS

Dimension de la carte	6000m x 6000m
Nombre de RSU	9
Porté de la RSU	1000m
Nombre de véhicules initiaux	900
Vitesse minimale, vitesse maximale	20-60km/h
Nombre de capteurs nécessaires *n*	40
Intervalle de publication des propriétés de contexte	10s
Temps de validité des propriétés de contexte	15s
Temps de validité d'une requête pull	15s
Intervalle avant l'envoi d'une nouvelle requête *tâche de capture*	60s

4.2.4.2. Résultats et discussion

Les premiers résultats, *Figure 31*, concernent la *méthode 1* et montrent l'impact de la valeur seuil ξ sur les pertes de paquets.

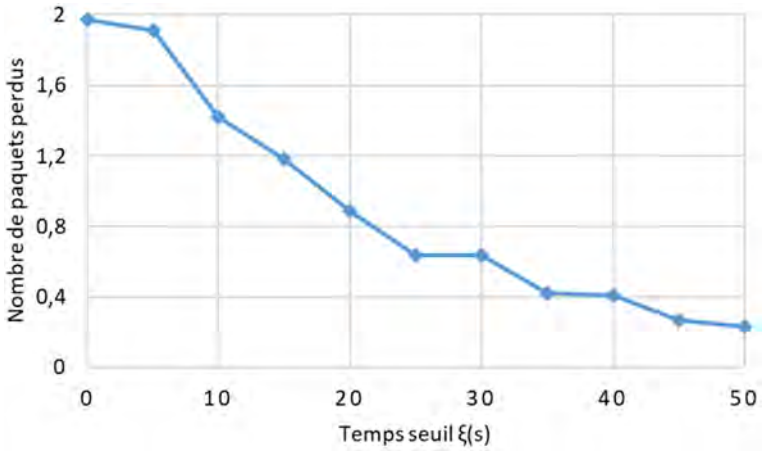


Figure 31 : Pertes de paquets en fonction de la valeur seuil ξ

Pour le modèle S²aaS initial, c'est-à-dire pour $\xi = 0s$, les pertes de paquets s'élèvent à 1,97. Lorsque $\xi = 5s$, les pertes de paquets passent à 1,91. Il n'y a pas donc d'améliorations notables comparé au modèle S²aaS initial. Par contre pour

$\xi = 10s$, les pertes de paquets passent à 1,42 ; soit une réduction de 0,55 comparée au S^{2}aaS initial. En prenant des valeurs de ξ de plus en plus élevées, les pertes diminuent considérablement. Le système s'approche d'un taux de perte de paquets nul lorsque ξ avoisine 50s. Cependant, en optant pour les grandes valeurs de ξ, la zone de capture des véhicules se voit aussi réduite, ce qui compromet l'efficacité du S^{2}aaS. En effet, suivant le *Tableau 10*, pour $\xi = 50s$, les capteurs incorporés dans un véhicule roulant à 20km/h ne pourront pas être sélectionnés par la RSU lorsqu'il s'approche de 280m du prochain point d'intersection hors zone couverture de cette RSU. Pour une vitesse de 60km/h, la distance de non-sélection des capteurs est de 833m. Ainsi, des capteurs hauts de gamme risquent ne pas être sélectionnés à cause de la vitesse du véhicule les incorporant ; cela impactera négativement sur la qualité du S^{2}aaS.

En somme, un compromis entre la réduction du risque de perte de paquets et la qualité de capture est requis. Dans la suite, nous fixerons $\xi = 10s$, signifiant que la portée de la RSU sera réduite d'une distance comprise entre 56 à 167m selon la vitesse des véhicules.

La *Figure 32* montre l'impact des différentes méthodes proposées pour la réduction des pertes de paquets et de la charge réseau.

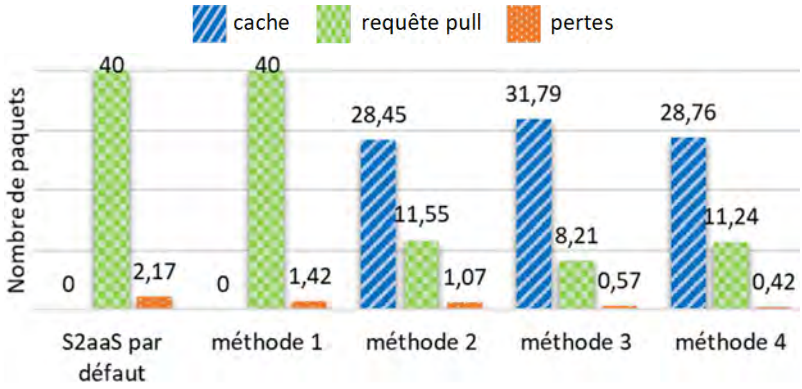


Figure 32 : Performances des méthodes d'optimisation réseau du S^{2}aaS

En exploitant les résultats indiqués sur la *Figure 32*, nous pouvons déterminer le taux de mise en cache de données et le taux de délivrance de paquets des différentes méthodes d'optimisation proposées, voir *Tableau 11*.

Tableau 11 : Détails sur les taux de mise en cache et de délivrance de paquets

	S^{2}aaS par défaut	méthode 1	méthode 2	méthode 3	méthode 4
Taux de mise en cache (%)	0	0	71,12	79,47	71,90
Taux de délivrance de paquets (%)	94,57	96,45	90,73	93,06	96,26

En se basant la *Figure 32*, on voit que le modèle *S^2aaS initial* et la *méthode 1* présente les charges réseaux les plus élevées. Cela s'explique par l'absence de mise en cache de données comme le montre *Tableau 11*. En d'autres termes, pour n mesures requises par le middleware, n requêtes pull seront envoyées aux véhicules impliqués. Les *méthodes 2, 3 et 4*, basées sur des techniques de mise en cache, présentent l'avantage de pouvoir réduire la charge réseau, voir *Figure 32*. Notons aussi que la mise en cache permet indirectement de réduire les pertes de paquets par le simple fait d'empêcher l'envoi de plusieurs requêtes pull. Malgré cet avantage, le *Tableau 11* montre que la *méthode 1* demeure la plus efficace en termes de taux de délivrance de paquets. Per contre, cette méthode ne permet pas de réduire la charge réseau. Comparée à la *méthode 2*, la *méthode 3* présente de meilleures performances. En effet, en plus d'hériter toutes les fonctionnalités de la *méthode 2*, la *méthode 3* intègre des capteurs zombies dans le modèle. L'utilisation de ces capteurs présente l'avantage de réduire encore plus la charge réseau. La *méthode 4* présente de meilleures performances en termes de taux de délivrance de paquets comparée à la *méthode 3*. Cela est prévisible, car la *méthode 4* hérite, en plus de toutes les fonctionnalités de la *méthode 3*, des fonctionnalités d'évitement de pertes de paquets définies dans la *méthode 1*. Il faudra noter cependant que les méthodes de mise en cache présentent l'inconvénient d'exploiter des mesures temps réel avec des mesures mise en cache. Cela peut ne pas être approprié pour certaines applications telles que celles de monitoring en temps réel ou de surveillance de l'apparition d'un événement.

4.2.5. Analyse qualitative

Dans le modèle de Abdelwahab et al. [64], le *minimum sensing capability* $R(j)$, l'équivalent des propriétés de contexte C, est défini sous une forme relativement basique. En effet, les propriétés utilisées (capacité de stockage minimale, puissance minimale de la CPU, temps d'exécution minimal) reflètent plus la qualité d'un serveur que celle d'un capteur. Par conséquent, $R(j)$ ne présente pas beaucoup d'intérêt pour l'IoV.

Quant au modèle CASARAM, proposé par Perera et al. [35], l'inconvénient demeure sa complexité d'utilisation. En effet, beaucoup de paramètres (valeurs des propriétés de contexte, facteurs de pondération pour chaque propriété de contexte...) devront manuellement être renseignés par l'utilisateur. Ainsi, un utilisateur sans connaissance des technologies des capteurs ne pourra utiliser leur application. Pour les utilisateurs expérimentés, l'usage de l'application demeure aussi difficile et le paramétrage risque d'être non optimale du fait des nombreuses valeurs à renseigner.

Le modèle proposé présente des avantages comparés aux travaux de Abdelwahab et al. [64] et de Perera et al [35]. Dans un premier temps, les propriétés de contexte sont pleinement exploitées pour décrire les capteurs dans l'IoV par le biais de la base de données CPDB, voir *Figure 30*. Ces mêmes propriétés de contexte permettent de déterminer les capteurs éligibles, ce qui permet

d'économiser des ressources de calcul et d'accélérer le processus de classement et de la sélection des capteurs. Dans un second temps, les modèles de Abdelwahab et al. [64] et de Perera et al [35] ne proposent pas des techniques d'optimisation réseau, contrairement à notre modèle dont quatre méthodes d'optimisation sont proposées. En outre, le modèle proposé se base sur une architecture optimisée pour la communication véhiculaire contrairement aux modèles de Abdelwahab et al. [64] et de Perera et al [35] qui sont basés sur des architectures centralisées (communication directe entre les capteurs et les middleware/serveurs). En outre, l'utilisation S^2aaS est rendue beaucoup plus aisée en optant de laisser à la charge de l'application et du middleware la gestion du paramétrage avancé. En effet, l'utilisateur ne fait qu'indiquer les types de mesures souhaitées M, la zone de relève Z et le traitement associé à ces mesures I. La détermination des critères de la qualité de mesure Q, les facteurs de pondération et les valeurs optimales des propriétés de contexte C sont fournis par le serveur d'application et le middleware. Ce qui n'est pas le cas du modèle de Perera et al. [35] qui exige que ces paramètres soient fournis de façon manuelle par l'utilisateur.

4.3. Modèle Sensing as a Service : seconde proposition

La limite principale de la première proposition réside sur l'étape de la génération de la *requête tâche de capture*. En effet, l'optimisation des paramètres liés à la sélection optimale des capteurs se base la résolution d'un problème de programmation linéaire multiobectif, voir *équation* (5). Les méthodes résolutions d'un tel problème sont complexes et, par conséquent, risque de générer une latence supplémentaire lors de la génération de la *requête tâche de capture*. De même, aucune méthode n'est proposée pour déterminer les valeurs de certains paramètres tels que les facteurs de pondération (α_i, β_i) et les valeurs seuils (n_0, q_i).

En outre, dans la première proposition, l'éligibilité d'un capteur dépend de la durée de validité définie par la RSU. Cette durée a une valeur fixe et peut ne pas convenir à toutes les applications. Par exemple, certaines applications exigent des données à temps réels, tandis que d'autres applications ont besoin de données d'historique disponibles en cache. Une autre limite de la première proposition réside sur l'uniformité du nombre de mesures (nombre de capteurs n) que doit fournir chaque RSU, sachant que la qualité de mesure peut varier d'une RSU à une autre. La répartition du nombre de capteurs à sélectionner par RSU devrait tenir compte de cela afin d'améliorer encore plus la qualité des mesures à recueillir. Enfin, chaque mesure M_i à effectuer exige le même nombre de capteurs n. Cela n'est ni flexible et ni optimale du fait que le serveur d'application peut nécessiter moins de capteurs pour une mesure $M_i \in M$, ou plus de capteurs pour une autre mesure $M_j \in M$ afin de fournir le service demandé.

Le modèle proposé dans cette seconde approche vise donc à améliorer les performances et la qualité des mesures de la première proposition.

4.3.1. Modèle fonctionnel

Dans le nouveau modèle proposé, *Figure 33*, deux processus s'exécutent en parallèle, à savoir : la collecte des propriétés de contexte et le traitement des requêtes des clients.

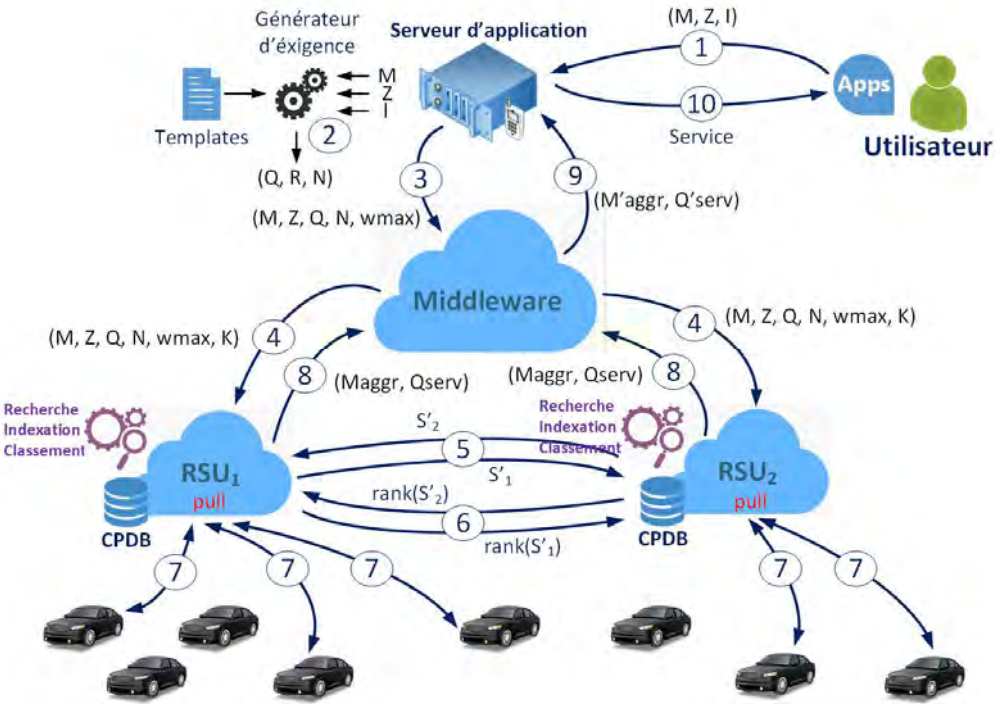


Figure 33 : Principe de fonctionnent du S²aaS : seconde proposition

- *Collecte des propriétés de contexte*

Chaque véhicule publie de façon périodique vers une RSU les propriétés de contexte de ses capteurs incorporés. A la réception de chaque publication, la RSU met à jour sa table de propriétés de contexte stockée dans une base de données CPDB (Context Proprieties DataBase). Vu la dynamique des VANET, la durée de validité des propriétés de contexte demeure très limitée. Par conséquent, si une RSU ne reçoit pas la publication de propriétés de contexte d'un véhicule durant une certaine période, l'entrée correspondante est supprimée de la CPDB.

- *Traitement des requêtes du client S²aaS*

Le client utilise une application pour accéder à un service. Cette application fournit une interface permettant au client d'indiquer les types de mesures souhaitées M, la zone de relève Z et le traitement associé à ces mesures I ①. Cette requête est traitée par un serveur d'application au niveau de la couche des applications. Ce traitement ② consiste à générer à partir de (M, Z, I), un ensemble de qualités des mesures Q, un ensemble de niveaux relatifs d'importances R parmi les contextes de propriétés et un ensemble N de nombres de capteurs nécessaires pour chaque type de mesure à collecter. A partir de R, le serveur d'application détermine le vecteur

w_{max} des poids des propriétés de contexte, puis envoie au middleware une requête de mesures avec les paramètres M, Z, Q, N et w_{max} ③. En fonction de la zone de relève Z, le middleware sollicite un ensemble K de RSU afin de collecter les mesures demandées ④. Chaque RSU_k détermine en fonction de M et de Q l'ensemble des capteurs éligibles S'_k, puis publie le résultat auprès des autres RSU ⑤ sollicitées pour la même tâche. Il s'en suivra alors, sur chaque RSU_k, un classement des capteurs éligibles S'_k, puis la publication de ce classement auprès des autres RSU ⑥. Ainsi à la base de la connaissance de l'ensemble des classements, chaque RSU_k est en mesure de sélectionner ses meilleurs capteurs dans S'_k puis d'envoyer des requêtes pull ⑦ aux véhicules embarquant ces capteurs. A la réception des réponses, les RSU agrègent les mesures obtenues M_{aggr} et la qualité effective Q_{serv} des mesures avant de les envoyer au middleware ⑧. Ce dernier agrège à nouveau les mesures venant des différentes RSU puis les transmet au serveur d'application ⑨. Ce dernier se chargera alors de traiter ces données conformément à I puis fournir le service demandé par le client ⑩.

4.3.2. Modèle formel

Ce modèle formel décrit l'utilisation des propriétés de contexte dans le processus de recherche et de sélection des capteurs.

- *Détermination du service demandé par l'utilisateur*

L'application propose une interface à l'utilisateur afin de récupérer les paramètres M, Z, I qui constituent le service souhaité.

$M = \{M_1, M_2, \dots, M_i, \dots\}$ est l'ensemble des types de mesure à collecter, où M_i indique un type spécifique de mesure tel que la température.

$Z = \{P_1, P_2, \dots, P_z, \dots\}$ est la zone géographique où s'effectueront les mesures, où P_z indique un point délimitant la zone.

I représente le service désiré tel les résultats d'une analyse statistique, une carte géographique annotée…

- *Génération des exigences du service*

Le générateur des exigences de services prend en entrée les paramètres (M, Z, I) et en s'aidant d'un ensemble de patrons (templates) de valeurs de préférence préétablies, détermine les valeurs adéquates pour Q, R et N.

$Q = \{Q_1, Q_2, \dots, Q_i, \dots\}$ est l'ensemble des qualités de mesure pour M, où Q_i exprime la qualité requise pour chaque capteur devant effectuer une mesure M_i.

$Q_i = \{(CP_1, q_1), (CP_2, q_2), \dots, (CP_j, q_j), \dots\}$ est l'ensemble des tuples exprimant pour chaque mesure M_i, le contexte de propriété CP_j et sa valeur seuil minimale requise q_j.

$R = \{R_1, R_2, \dots, R_i, \dots\}$ est l'ensemble des importances comparatives entre les propriétés de contexte définies, où $R_i = \{\dots, a_{ij}, \dots\}$ contient les importances comparatives relatives aux propriétés de contextes de Q_i.

$N = \{N_1, N_2, \dots, N_i, \dots\}$ est l'ensemble contenant, pour chaque mesure M_i, le nombre de capteurs N_i requis pour effectuer cette mesure.

La maintenance et la mise à jour de ces patrons sont gérées par le middleware. Notons également que l'application offre à l'utilisateur la possibilité de modifier Q, R et N.

- *Pondération des propriétés de contexte*

La méthode AHP (Analytic Hierarchy Process) [144] est utilisée par le serveur d'application pour calculer le poids de chaque propriété de contexte. L'importance comparative peut être considérée comme un score relatif a_{ij}, qui représente l'importance d'une propriété de contexte i par rapport à une autre propriété de contexte j [36]. AHP suggère de choisir une valeur impaire comprise entre 1 à 9 pour représenter cette importance, voir *Figure 34*.

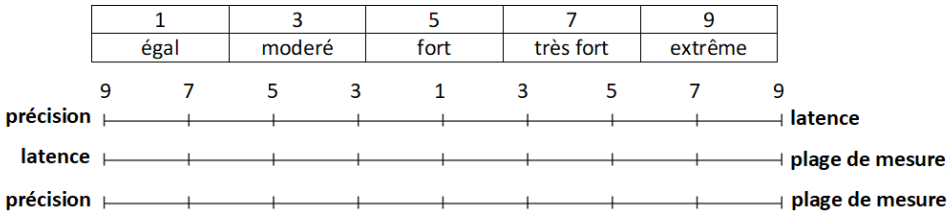


Figure 34 : Exemple de scores AHP pour les propriétés de contexte

A partir de R, une matrice de comparaison $CM_{M_i} = [b_{uv}]_{|Q_i| \times |Q_i|}$ pour une mesure M_i est créée en suivant les règles suivantes :

$$b_{uv} = \begin{cases} a_{ij} & si\ u = i\ et\ v = j \\ \dfrac{1}{a_{ij}} & si\ u = j\ et\ v = i \\ 1 & si\ u = v \end{cases} \qquad (12)$$

L'étape suivante consiste à déterminer les vecteurs propres $w_1, w_2, \dots, w_p$ de la matrice CM_{M_i}, où p est le nombre de vecteurs propres de CM_{M_i}. Les valeurs propes $\lambda_1, \lambda_2, \dots, \lambda_p$ correspondantes pourront ainsi être déterminées. Dès lors, on détermine $\lambda_{max} = max(\lambda_1, \lambda_2, \dots, \lambda_p)$, puis le vecteur propre correspondant w_{max}. AHP définie w_{max} comme le vecteur des poids des propriétés de contexte. Afin de vérifier si les scores relatifs configurés dans R_i sont consistants, c'est-à-dire vérifier s'il n'existe pas d'incohérences dans la comparaison d'importance de chaque propriété de contexte. Alonso et al. [145] proposent de tester la condition suivante :

$$\lambda_{max} < n + 0.1(1.7699n + 4.3513) \quad o\grave{u}\ n = |Q_i| \qquad (13)$$

Si la condition n'est pas vérifiée, de nouveaux scores relatifs doivent être choisis. Néanmoins, ce cas de figure ne peut pas se produire grâce à l'utilisation du générateur d'exigence. En effet, ce dernier effectue la procédure de pondération des propriétés de contexte associées à R_i proactivement. En d'autres termes, le générateur des exigences de services peut, lors de la création d'un nouveau patron, déterminer, valider puis stocker à l'avance le couple $(\lambda_{\max}, w_{max})$ associé à ce patron. Finalement, le serveur d'application envoie au middleware une requête de demande de mesures M avec les paramètres Z, Q, N et w_{max}.

- *Identification des RSU*

En fonction de la zone géographique Z, le middleware identifie l'ensemble des RSU auxquelles il transmettra la requête demande de mesures M. En plus des paramètres M, Z, Q, N et w_{max}, le middleware inclura le paramètre K dans la requête à envoyer aux RSU.

$K = \{RSU_1, RSU_2, \dots, RSU_k, \dots\}$ est l'ensemble des RSU présentes dans la zone géographique Z.

- *Identification des capteurs éligibles*

Cette étape est gérée par les $RSU \in K$, et consiste à éliminer les capteurs non utiles pour le service demandé. Le but étant de sauvegarder de ressources de calcul et d'accélérer la procédure de sélection des meilleurs capteurs.

Soient :

$S_k = \{S_{1,k}, S_{2,k}, \dots, S_{s,k}, \dots\}$ l'ensemble des capteurs dont leurs propriétés de contexte sont stockés au niveau de la CPDB de la RSU_k.

$M_{S_{s,k}} = f(S_{s,k})$ l'ensemble des mesures que peut effectuer le capteur $S_{s,k}$.

$S'_k \subset S_k$ l'ensemble des capteurs éligibles.

Un capteur $S_{s,k}$ est éligible s'il est capable d'effectuer au moins une mesure $M_i \in M$ tout en respectant l'ensemble des contraintes de qualités Q_i. Ainsi :

$$S_{s,k} \in S'_k \; si \begin{cases} M_{S_{s,k}} \cap M \neq \emptyset \quad et \\ \forall\, (CP_j, q_j) \in Q_i, \forall\, (CP_j^{S_{s,k}}, q_j^{S_{s,k}}) \in Q_i^{S_{s,k}} \; alors \; q_j^{S_{s,k}} \geq \; q_j \end{cases} \quad (14)$$

La suite consiste à déterminer, pour chaque mesure M_i, l'ensemble $S'_{k,M_i} \subset S'_k$ des capteurs pouvant mesurer M_i.

$$S_{s,k} \in S'_{k,M_i} \; si \begin{cases} M_i \in M_{S_{s,k}} \cap M \;\; et \\ \forall \left(CP_j^{S_{s,k}}, q_j^{S_{s,k}}\right) \in Q_i^{S_{s,k}}, \forall\, (CP_j, q_j) \in Q_i \; alors \; q_j^{S_{s,k}} \geq \; q_j \end{cases} \quad (15)$$

Après identification des capteurs éligibles, le RSU_k publie S'_k auprès de toutes les RSU appartenant à K.

- *Allocation des tâches aux RSU*

Pour chaque RSU_k, un ensemble de tâches lui seront attribuées. Le contenu de la tâche pour tout $M_i \in M$ et pour tout $RSU_k \in K$ peut être formulé comme suit :

$$\begin{cases} \sum_{k \in |K|} n_{M_i,k} = N_i \\ 0 \leq n_{M_i,k} \leq \left|S'_{k,M_i}\right|, \ \ n_{M_i,k} \in \mathbb{N} \end{cases} \tag{16}$$

Où $n_{M_i,k}$ est le nombre de capteurs requis pour mesurer M_i au niveau du RSU_k. Soit :

$c_{q,M_i} = \{n_{M_i,1}, n_{M_i,2}, \dots, n_{M_i,k}, \dots\}$ la $q^{ième}$ solution de (16).

Afin d'obtenir l'ensemble des solutions c_{q,M_i} possibles, Hsu et al. [36] proposent l'utilisation d'un générateur de fonction polynomial $P_{M_i}(y)$.

$$P_{M_i}(y) = \prod_{k \in |K|} \left(\sum_{n=0}^{\left|S'_{k,M_i}\right|} (G_k y)^n \right) \tag{17}$$

Où G_k est un symbole pour représenter la RSU_k.

La tâche L_{k,M_i} assignée au RSU_k est alors définie comme suit :

$$\begin{aligned} L_{k,M_i} &= \left\{ n_{M_i,k} \mid n_{M_i,k} \in c_{q,M_i} \ et \ q = \{1,2,\dots\} \right\} \\ &= \left\{ n_{M_i,k} \middle| n_{M_i,k} \in c_{1,M_i}, \ \ n_{M_i,k} \middle| \ n_{M_i,k} \in c_{2,M_i}, \dots, \ \ n_{M_i,k} \middle| n_{M_i,k} \in c_{q,M_i}, \dots \right\} \end{aligned} \tag{18}$$

Afin de déterminer c_{q,M_i} et L_{k,M_i}, l'*Algorithme 1* [36] détaillé ci-dessous est utilisé. Une fois l'ensemble des solutions connues, la suite consiste à déterminer la solution qui présentera le meilleur score en termes d'un ensemble de critères tel que la qualité du capteur, la diminution de la charge réseau…

Algorithme 1 assignation des tâches à la RSU [36]

Require : $P_{M_i}(y), N_i$

Output : L_{k,M_i}, c_{q,M_i}
$T \leftarrow$ extract coeficients of y^{N_i} in $P_{M_i}(y)$
for each item t in T **do**
    $c_{q,M_i} = \left(n_{M_i,1} = G_1^{'} s \ exponent \ in \ t, \dots, n_{M_i,k} = G_k^{'} s \ exponent \ in \ t\right)$
    for $k \in K$ **do**
        $L_{k,M_i} = union(L_{k,M_i}, n_{M_i,k})$
    end for
end for
return L_{k,M_i}, c_{q,M_i}

- *Calcul du score des taches*

Toute RSU_k doit calculer le score pour chaque tâche qu'il pourra éventuellement exécuter. Le score dépendra de la qualité des capteurs, mais aussi de leurs dépendances. Il y a dépendance entre capteurs éligibles lorsque ces derniers sont incorporés dans le même véhicule. L'avantage de cette dépendance réside sur l'agrégation des requêtes *pull* qui permet de diminuer la charge réseau.

Soient :

$V_{S'_{k,M_i,l}} = \{V_{1,k}, V_{2,k}, \dots, V_{v,k}, \dots\}$ l'ensemble des véhicules incorporant au moins un capteur dans $S'_{k,M_i,l}$.

$S_k^{V_{v,k}} = \{S_{1,k}^{V_{v,k}}, \dots, S_{r,k}^{V_{v,k}}, S_{s,k}^{V_{v,k}}, \dots\}$ l'ensemble des capteurs éligibles incorporés dans le véhicule $V_{v,k}$.

$pull_{V_{v,k}}^{aggr} = \left|S_k^{V_{v,k}}\right| - 1$ le nombre de requêtes pull pouvant être agrégées au niveau du véhicule $V_{v,k}$.

$pull_{S'_{k,M_i,l}}^{aggr}$ le nombre total de requêtes pull agrégées dans $V_{S'_{k,M_i,l}}$.

$Q_{S'_{k,M_i,l}}$ est la somme des scores dus à la qualité $Q_i^{S_{s,k}}$ de chaque capteur $S_{s,k} \in S'_{k,M_i,l}$.

$S'_{k,M_i,l} \in Cand_{M_i,k,l} = \{y \mid y \in ensemble\ de\ puissance(S'_{k,M_i}), |y| = l\}$, c'est-à-dire l'ensemble de tout sous-ensemble de S'_{k,M_i} de taille l.

Alors, on a :

$$pull_{S'_{k,M_i,l}}^{aggr} = \sum_{V_{v,k} \in V_{S'_{k,M_i,l}}} pull_{V_{v,k}}^{aggr} \tag{19}$$

Et,

$$Q_{S'_{k,M_i,l}} = \sum_{S_{s,k} \in S'_{k,M_i,l}} \sum_{\left(CP_j^{S_{s,k}}, q_j^{S_{s,k}}\right) \in Q_i^{S_{s,k}}} w_{max}\left(CP_j^{S_{s,k}}\right) \times q_{j,norm}^{S_{s,k}} \tag{20}$$

Où :

$w_{max}\left(CP_j^{S_{s,k}}\right)$ est le poids de la propriété de contexte $CP_j^{S_{s,k}}$ dans w_{max}.

$q_{j,norm}^{S_{s,k}} \in [0,1]$ est la valeur normalisée de la propriété de contexte $CP_j^{S_{s,k}}$. Comme suggéré par Perera et al. [35] et Hsu et al. [36], la méthode *min-max* sera utilisée pour la normalisation.

On appelle $m_{M_i,k,l}$ le score moyen pondéré d'un ensemble de capteurs $\{S_{1,k}, \dots, S_{s,k}, \dots, S_{l,k}\} \subset S'_{k,M_i}, l \in L_{k,M_i}$.

$$m_{M_i,k,l} = Q_{S'_{k,M_i,l}}\gamma + (1-\gamma) \times pull^{aggr}_{S'_{k,M_i,l}} \tag{21}$$

Où :

$\gamma \in [0,1]$ est un paramètre de pondération. Dans le modèle S²aaS, la qualité du capteur a beaucoup plus d'importance que la diminution de la charge réseau. Par conséquent, nous considérerons toujours $\gamma \gg 1-\gamma$.

Pour la RSU_k, nous noterons $M_{M_i,k,l}$ l'ensemble des capteurs utilisés pour l'obtention du score maximal $m^*_{M_i,k,l}$.

$$M_{M_i,k,l} = S'^*_{k,M_i,l} = \underset{S'_{k,M_i,l} \in Cand_{M_i,k,l}}{\operatorname{argmax}} \left(m_{M_i,k,l} \right) \tag{22}$$

Arrivée à ce stade, la RSU_k publie le tuple $\left(m^*_{M_i,k,l}, M_{M_i,k,l}\right)$ vers toutes les autres RSU appartenant à l'ensemble K.

- *Sélection des capteurs*

A la réception de l'ensemble des tuples $\left(m^*_{M_i,k,l}, M_{M_i,k,l}\right)$ issus des autres $RSU \in K$, la RSU_k procède au calcul du cumul des scores pour chaque solution c_{q,M_i}. La RSU_k sélectionnera alors la solution $c^*_{M_i}$ présentant le meilleur score.

$$c^*_{M_i} = \underset{\forall c_{q,M_i}}{\operatorname{argmax}} \left(\sum_{n_{M_i,k} \in c_{q,M_i}, k \in K} m^*_{M_i,k,l} \right) \tag{23}$$

Le calcul se faisant sur la base d'un même ensemble de tuples, alors la solution optimale $c^*_{M_i}$ sera identique sur l'ensemble des RSU. A partir de $c^*_{M_i}$, le RSU_k identifie l'ensemble $S^*_{M_i,k}$ de ses capteurs sélectionnés pour effectuer une mesure M_i.

$$S^*_{M_i,k} = M_{M_i,k,n_{M_i,k}} \quad avec \ n_{M_i,k} \in c^*_{M_i} \tag{24}$$

- *Envoie des requêtes pull*

Chaque RSU identifie l'ensemble des véhicules auxquels il doit envoyer une requête pull.

Soient :

$S^*_{M,k} = \{S^*_{M_1,k}, S^*_{M_2,k}, \dots, S^*_{M_i,k}, \dots\}$ l'ensemble des capteurs sélectionnés par le RSU_k pour effectuer une mesure dans M.

$V^*_k = \{V^*_{1,k}, V^*_{2,k}, \dots, V^*_{v,k}, \dots\}$ l'ensemble des véhicules incorporant au moins un capteur dans $S^*_{M,k}$.

$S_k^{*,V_{v,k}^*} = \left\{ S_{1,k}^{*,V_{v,k}^*}, \dots, S_{r,k}^{*,V_{v,k}^*}, S_{s,k}^{*,V_{v,k}^*}, \dots \right\} \mid S_k^{*,V_{v,k}^*} \subset S_{M,k}^*$ l'ensemble des capteurs embarqués dans le véhicule $V_{v,k}^*$ et sélectionnés pour effectuer une mesure dans M.

Le nombre $n_{pull,k}^*$ de requêtes pull envoyées par le RSU_k est alors égal à :

$$n_{pull,k}^* = |V_k^*| \leq \left|S_{M,k}^*\right| \quad (25)$$

A la réception de l'ensemble des réponses des véhicules présents dans V_k^*, le RSU_k agrège à son tour les réponses puis le transfert au middleware.

4.3.3. Optimisation réseaux

Pour l'acquisition de données de capteurs automobiles, la RSU doit envoyer plusieurs requêtes pull aux véhicules concernés. Afin de réduire la charge réseau et la latence, une méthode de mise en cache de données et un schéma de prise de décision décentralisée sont proposés.

4.3.3.1. Réduction de la charge réseau : mise en cache de données

Cette méthode consiste à intégrer dans le processus de sélection, des capteurs dont leurs mesures sont présentes dans la cache de la RSU. Le but est de réduire la charge réseau entre la RSU et les véhicules.

Soient :

$d_{S_{s,k}}$ la durée de mise en cache des dernières mesures du capteur $S_{s,k}$

$\left(CP_{cache}, \frac{1}{d_{cache}}\right) \in Q_i$ le tuple contenant la durée maximale d_{cache} de mise en cache de la mesure M_i.

Alors :

Le capteur $S_{s,k}$ est éligible si $\frac{1}{d_{S_{s,k}}} \geq \frac{1}{d_{cache}}$ c'est-à-dire $d_{S_{s,k}} \leq d_{cache}$

Comparée à la première proposition, voir *section 4.2*, l'éligibilité d'un capteur $S_{s,k}$ ne dépendra plus de la durée de validité définie par la RSU, mais plutôt par la durée d_{cache} définie par le serveur d'application. Cela permet de gagner en flexibilité en permettant à l'application de fixer $d_{S_{s,k}}$ selon la nature de la représentation I_i. Par exemple, pour une application en temps réel, on prendra $d_{cache} = 0$, c'est-à-dire pas de sélection de mesures mises en cache. Par contre, pour des applications moins exigeantes en temps, on peut prendre $d_{cache} > 0$.

Pour faire une analyse comparative des performances, on considérera $d_{cache} \leq T(j)$; où $T(j)$ est définie dans [36] comme étant le temps nécessaire pour que le capteur j puisse fournir ces données de mesure.

Soient :

$N_{V_{v,k}}$ le nombre de mesures impliquant les capteurs du véhicule $V_{v,k}$ durant la période $\Delta t \leq d_{cache}$.

$n^*_{pullReq,V_{v,k}}$ le nombre de requêtes pull reçues par le véhicule $V_{v,k}$.

$n^*_{pullResp,V_{v,k}}$ le nombre de réponses envoyées par le véhicule $V_{v,k}$ vers le RSU_k.

$n^*_{tot,V_{v,k}}$ le nombre total de paquets échangés entre le véhicule $V_{v,k}$ et le RSU_k.

Alors :

$$n^*_{tot,V_{v,k}} = n^*_{pull,V_{v,k}} + n^*_{resp,V_{v,k}} \quad (26)$$

- *Charge réseau engendrée par le modèle S²aaS proposé*

Le modèle proposé permet au RSU_k d'utiliser les mesures stockées en cache. Le but est d'éviter d'envoyer des requêtes pull au véhicule $V_{v,k}$ à chaque fois que le RSU_k aurait besoin des mesures du capteur $S_k^{*,V^*_{v,k}}$.

Ainsi :

– Si $d_{S_{s,k}} < d_{cache}$, aucune requête pull n'est envoyée au véhicule $V_{v,k}$ car des mesures récentes se trouvent déjà en cache, d'où :

$\forall N_{V_{v,k}}$ on a $n^*_{pullReq,V_{v,k}} = 0,\ n^*_{pullResp,V_{v,k}} = 0$ et $n^*_{tot,V_{v,k}} = 0$ (27)

– Si $d_{S_{s,k}} > d_{cache}$, c'est-à-dire pas de mesures récentes en cache, une seule requête pull est envoyée au véhicule $V_{v,k}$, d'où :

$\forall N_{V_{v,k}}$ on a $n^*_{pullReq,V_{v,k}} = 1,\ n^*_{pullResp,V_{v,k}} = 1$ et $n^*_{tot,V_{v,k}} = 2$ (28)

- *Charge réseau engendrée par le modèle S²aaS de Hsu et al.* [36]

Dans ce modèle, la technique d'agrégation de paquets est utilisée au niveau du véhicule $V_{v,k}$. Ce dernier agrège dans un seul paquet l'ensemble des réponses aux requêtes reçues via le RSU_k dans l'intervalle de temps $T(j)$. Ainsi :

$\forall N_{V_{v,k}}$ on a $n^*_{pullReq,V_{v,k}} = N_{V_{v,k}},\ n^*_{pullResp,V_{v,k}} = 1$ et $n^*_{tot,V_{v,k}} = N_{V_{v,k}} + 1$ (29)

- *Charge réseau engendrée par le modèle S²aaS de Perera et al.* [35]

Dans ce modèle, aucune technique de mise en cache, ni d'agrégation de paquets n'est proposée, par conséquent :

$\forall N_{V_{v,k}}$ on a $n^*_{pullReq,V_{v,k}} = N_{V_{v,k}},\ n^*_{pullResp,V_{v,k}} = N_{V_{v,k}}$ et $n^*_{tot,V_{v,k}} = 2N_{V_{v,k}}$ (30)

La *Figure 35* montre l'évolution de la charge réseau $n^*_{tot,V_{v,k}}$ en fonction du nombre de mesures $N_{V_{v,k}}$ impliquant un véhicule $V_{v,k}$.

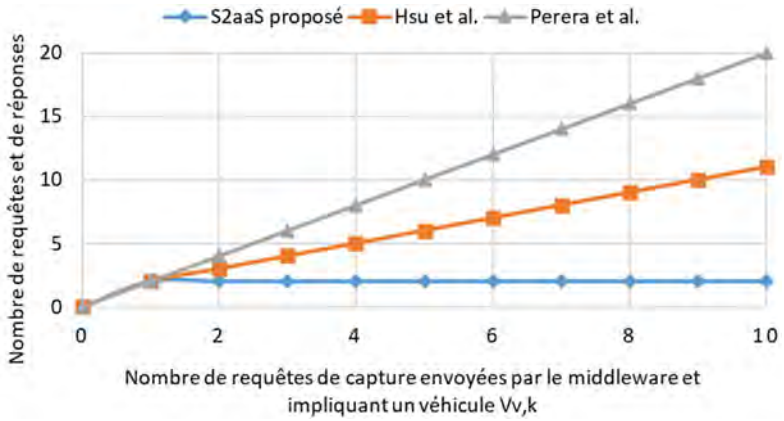


Figure 35 : Charge réseau entre la RSU_k et un véhicule sélectionné $V_{v,k}$

Comparé au modèle Perera et al. [35] et Hsu et al. [36], le modèle proposé présente une charge réseau beaucoup plus faible. En effet, notre modèle garde une charge réseau constante ($n^*_{tot,V_{v,k}} = 0$ ou $n^*_{tot,V_{v,k}} = 2$) peu importe le nombre le nombres de mesures $N_{V_{v,k}}$, ce qui n'est pas le cas des autres modèles dont la charge réseau augmente avec $N_{V_{v,k}}$.

4.3.3.2. Réduction de la charge réseau : prise de décision distribuée

L'un des objectifs du schéma de prise de décision distribuée est de réduire la charge réseau entre le middleware et les RSU. L'étude comparative concerne la charge réseau engendrée lors du traitement, par le middleware et la RSU, de la requête d'un seul utilisateur.

Soient :

$n_{publish,RSU_k}$ le nombre de paquets *publish* relayés par la RSU_k au middleware. Ces paquets transportent les propriétés de contexte des capteurs automobiles présentent dans la zone de la RSU_k.

$|M|$ le nombre de types de mesures à fournir à l'utilisateur.

$|V_k|$ le nombre de véhicules présents dans la zone de couverture de la RSU_k.

$n_{M_i,k}$ le nombre capteurs requis pour mesurer chaque grandeur $M_i \in M$ dans la zone de couverture du RSU_k.

$n^*_{sensing,RSU_k}$ la somme du nombre de requêtes envoyées par le middleware au RSU_k et du nombre de réponses envoyées par le RSU_k au middleware.

- *Charge réseau engendrée par le modèle S²aaS proposé*

Le modèle proposé est basé sur une architecture distribuée, par conséquent, les propriétés de contexte des capteurs ne sont pas transmises au middleware.

$$n_{publish,RSU_k} = 0 \quad (31)$$

Lorsque le client sollicite $|M|$ catégories de mesures, l'ensemble des requêtes du client sont agrégées dans un seul paquet. Ainsi, le middleware envoie une seule requête au RSU_k, et ce dernier envoie une seule réponse. D'où :

$$n^*_{sensing,RSU_k} = 2 \quad (32)$$

- *Charge réseau engendrée par le modèle S²aaS de Hsu et al.* [36]

Le modèle de Hsu et al. est aussi basé sur une architecture distribuée. D'où :

$$n_{publish,RSU_k} = 0 \quad (33)$$

Par contre, pour la collecte de données de capteurs, le middleware envoie séparément $|M|$ requêtes au RSU_k et reçois également $|M|$ réponses séparément de la part du RSU_k. D'où :

$$n^*_{sensing,RSU_k} = 2|M| \quad (34)$$

- *Charge réseau engendrée par le modèle S²aaS de Perera et al.* [35]

Le modèle de Perera et al. est basé sur une architecture centralisée, par conséquent, les propriétés de contexte des capteurs sont relayées au middleware par le RSU_k. D'où :

$$n_{publish,RSU_k} = |V_k| \quad (35)$$

Pour la collecte de données de capteurs, le middleware envoie séparément $n_{M_i,k}|M|$ requêtes au RSU_k et reçois également $n_{M_i,k}\,|M|$ réponses séparément de la part du RSU_k. D'où :

$$n^*_{sensing,RSU_k} = 2n_{M_i,k}|M| \quad (36)$$

La *Figure 36* montre l'évolution de la charge réseaux entre le middleware et le RSU_k.

Comme pour la stratégie de mise en cache, le modèle proposé présente la plus faible charge réseau comparé à ceux de Perera et al. [35] et Hsu et al. [36]. En effet, notre modèle garde une charge constante quel que soient $n_{M_i,k}$ et $|M|$. Le modèle de Hsu et al. [36], présente une charge réseau constante quel que soit $n_{M_i,k}$; par contre, cette charge augmente lorsque $|M|$ augmente. Le modèle de Perera et al. [35] présente une charge réseau beaucoup plus élevée. En effet, cette charge croit en fonction de $n_{M_i,k}$ et $|M|$.

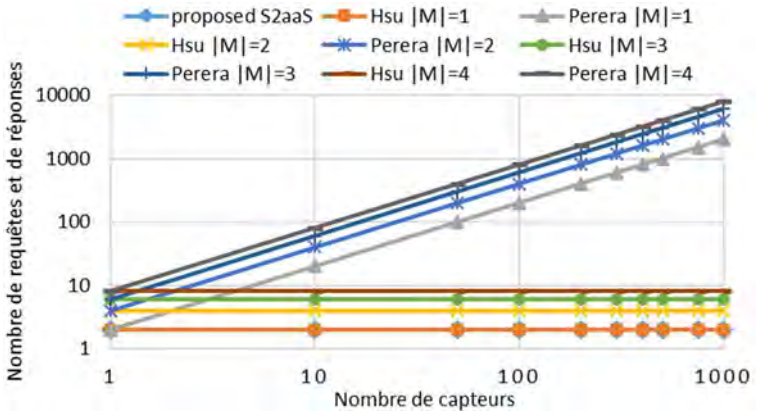


Figure 36 : Charge réseau entre le middleware et la RSU_k

4.3.3.3. Réduction de la latence : prise de décision distribuée

Deux types de latence seront déterminées, à savoir : la latence d'accès à un service S^2aaS et la latence liée à la publication des propriétés de contexte des capteurs automobiles.

Soient :

$L_{process}$: le cumul des latences dues au traitement des requêtes au niveau des RSU et du middleware.

$L_{publish}$: la latence liée à la publication des propriétés de contexte des capteurs automobiles.

$L_{collect}$: la latence liée à la collecte des données de capteurs automobiles.

L_{veh_RSU} : la latence de transmission de paquets entre un véhicule et une RSU, et vice versa.

L_{mid_RSU} : la latence de transmission de paquets entre le middleware et une RSU, et vice versa.

L_{RSUj_RSUk} : la latence de transmission de paquet entre un RSU_j et une RSU_k, et vice versa.

Vue la proximité entre les véhicules et les RSU, et l'éloignement entre les RSU et le middleware, nous considérerons que :

$$L_{RSUj\_RSUk} < L_{veh\_RSU} \text{ et } L_{veh\_RSU} \ll L_{mid\_RSU}$$

Dans le modèle de Perera et al. [35], le processus de recherche, de classement et de sélection des meilleurs capteurs est effectué de façon centralisée au niveau du middleware. Ce qui engendre un fort délai de traitement et une augmentation de la charge réseau [35], d'où $L_{process,Perera}$ très élevé. Par contre, la nature distribuée du modèle proposé et celui de Hsu et. [36], fait que les charges de traitement et réseau sont réparties entre le middleware et les différentes RSU, par conséquent, nous considérerons $L_{process,Hsu} = L_{process,proposedS2aaS} = 0$.

- *Latence du modèle S²aaS proposé*

Dans notre modèle, chaque véhicule envoie régulièrement à la RSU un paquet $publish$ contenant les propriétés de contexte de ses capteurs. D'où :

$$L_{publish} = L_{veh\_RSU} \tag{37}$$

Pour la collecte de données de capteurs, la latence concerne le moment où le middleware envoie une requête de capture à la RSU (étape ④) et le moment où ce middleware reçoit la réponse de la RSU (étape ⑧).

$$L_{collect} = L_{sensingReq} + L^{elig}_{syncRSU} + L^{rank}_{syncRSU} + L_{pull} + L_{sensingResp} \tag{38}$$

Où $L_{sensingReq} = L_{mid\_RSU}$ est la latence due à l'envoie par le middleware d'une requête de capture à la RSU (étape ④) et $L_{sensingResp} = L_{mid\_RSU}$ est la latence due à l'envoie, par la RSU, de la réponse au middleware (étape ⑧). $L^{elig}_{syncRSU} = L_{RSUj\_RSUk}$ est la latence due à la publication par chaque RSU de son ensemble de capteurs éligibles (étape ⑤). $L^{rank}_{syncRSU} = L_{RSUj\_RSUk}$ est la latence due à la publication par chaque RSU du résultat de classement de son ensemble de capteurs éligibles (étape ⑥). $L_{pull} = 2L_{veh\_RSU}$ est la différence de temps entre l'instant où la RSU envoie une requête pull au véhicule et l'instant où la RSU reçoit la réponse du véhicule. Finalement :

$$L_{collect} = 2L_{mid\_RSU} + 2L_{RSUj\_RSUk} + 2L_{veh\_RSU} \tag{39}$$

- *Latence du modèle S²aaS de Hsu et al.* [36]

En se basant sur l'architecture de collecte de données proposée par Hsu et al. [36], la passerelle agit comme une RSU en ce qui concerne la réception des propriétés de contexte des capteurs automobiles. D'où :

$$L_{publish} = L_{veh\_RSU} \tag{40}$$

Par contre, pour la collecte des données de capteurs, chaque passerelle ne fait que le classement des capteurs et renvoies le résultat au serveur (ou middleware). Ce dernier se charge de faire la sélection des meilleurs capteurs puis d'envoyer à nouveau à la passerelle la liste des capteurs sélectionnées. La passerelle envoie alors des requêtes pull à l'ensemble des véhicules dont au moins un de ses capteurs incorporés est sélectionné. A la réception des réponses des véhicules, la passerelle transmet ces réponses au serveur. Ainsi :

$$L_{collect} = L_{score} + L_{sensingReq} + L_{pull} + L_{sensingResp} \tag{41}$$

Où $L_{score} = 2L_{mid\_RSU}$ est la différence de temps entre l'instant où le middleware envoie une requête de classement à la passerelle et l'instant où le middleware reçoit la réponse de la passerelle. Finalement :

$$L_{collect} = 2L_{mid\_RSU} + 2L_{mid\_RSU} + 2L_{veh\_RSU} \tag{42}$$

- *Latence du modèle S²aaS de Perera et al.* [35]

On se basant sur l'architecture CASARAM proposé par Perera et al. [35], la passerelle jouer simplement le rôle de relais de paquets entre le middleware et les véhicules. Ainsi, pour la publication des propriétés de contexte des capteurs automobiles, on aura :

$$L_{publish} = L_{veh\_RSU} + L_{mid\_RSU} \quad (43)$$

Pour la collecte des données de capteurs ; le processus de recherche, de classement et de sélection des meilleurs capteurs est effectué au niveau du middleware. Ensuite, le middleware envoie des requêtes pull à l'ensemble des véhicules incorporant au moins un capteur sélectionné. Ces véhicules envoient chacun une réponse au middleware. D'où :

$$L_{collect} = L_{sensingReq} + L_{sensingResp} + L_{process} \quad (44)$$

Où $L_{sensingReq} = L_{sensingResp} = L_{mid\_RSU} + L_{veh\_RSU}$, finalement :

$$L_{collect} = 2L_{mid\_RSU} + 2L_{veh\_RSU} + L_{process} \quad (45)$$

En comparant les latences $L_{publish}$, nous remarquons que le modèle de Hsu et al. [36] et celui proposé présentent la même latence. Cette dernière est plus faible que celle du modèle de Perera et al. [35] du fait de la nature distribuée de notre modèle et de celui de Hsu et al. [36].

En ce qui concerne les latences $L_{collect}$, nous remarquons que le modèle proposé présente une latence beaucoup plus faible que celle du modèle de Hsu et al [36]. En effet, la différence de temps entre ces deux modèles est de $\Delta t = 2L_{mid\_RSU} - 2L_{RSUj\_RSUk}$. Comparé au modèle de Perera et al. [35], le modèle S²aaS présente également une latence plus faible. En effet, en se basant sur les tests de performance de CASARAM [35], $L_{process} \gg 2L_{RSUj\_RSUk}$ lorsque le nombre de capteurs impliqués devient grand.

4.3.4. Analyse qualitative

Le principal défaut du modèle de Perera et al. [35], nommé CASARAM, réside sur la méthode de récupération des préférences de l'utilisateur. En effet, les paramètres à renseigner par ce dernier pour pondérer l'importance de chaque propriété du capteur sont nombreux et complexes. Par conséquent, ce modèle est réservé pour des utilisateurs expérimentés en technologies des capteurs. En outre, CASARAM est un modèle centralisé, c'est-à-dire est entièrement implémenté au niveau du middleware. Et, comme mentionné par Hsu et al. [36], collecter et stocker de manière centralisée les propriétés de contexte hautement dynamiques des capteurs automobiles peut générer un large trafic réseau.

Hsu et al. [36] proposent une architecture de collecte des propriétés de contexte basée sur un modèle distribué. Ainsi, chaque véhicule envoie les propriétés des capteurs incorporés vers une passerelle locale au lieu du middleware. En outre,

avec ce modèle, l'usage du service S^{2}aaS est facilité. En effet, l'utilisateur est simplement tenu de renseigner des conditions implicites par le biais de niveaux de préférences (1 à 9) entre les propriétés de contexte. Néanmoins, lorsque le nombre de propriétés de contexte à traiter par le service demandé est élevé, le nombre de champs à renseigner par l'utilisateur devient très élevé. Par conséquent, des problèmes de consistance liées aux valeurs de préférences des propriétés de contexte pourraient apparaitre. Dans ce modèle également, après que les passerelles aient terminées de calculer le score des tâches, elles transmettent leurs résultats au serveur (middleware). Ce dernier fait la sélection des capteurs puis envoie à nouveau des requêtes vers ces passerelles afin de collecter les données sollicitées. Cette procédure est relativement longue et augmente le risque qu'un ou plusieurs véhicules quittent la zone de couverture de la passerelle avant qu'ils ne puissent recevoir une requête, la traiter puis envoyer la réponse.

Dans notre modèle, l'usage du service proposé est encore plus facilité. En effet, tout le paramétrage complexe (Q, R et N) est laissé à la charge du serveur d'application. Ainsi, les exigences de services ne sont plus, désormais, fournies par l'utilisateur, mais plutôt obtenues via un générateur d'exigences. Ce dernier se base sur un ensemble de patrons pour choisir de manière optimale les exigences de services (qualités des capteurs, types de capture, critères de préférences). Ainsi, même un utilisateur sans expérience sur la technologie des capteurs peut bénéficier des services S^{2}aaS de façon optimale. En outre, le modèle proposé est conçu pour répondre aux exigences de la communication véhiculaire. Ainsi, une architecture distribuées composées essentiellement de RSU est proposée pour la recherche, le classement, la sélection et la collecte des données de capteurs automobiles. Par conséquent, l'essentielle des échangent se feront entre une RSU et les véhicules à proximité, d'où la diminution du risque de perte de données et des délais.

4.4. Exemple d'application pratique du S^{2}aaS

Dans cette section, un exemple d'application S^{2}aaS nommée « *Dynamic Best Air Quality Routing (DBAQ-Routing)* », c'est-à-dire itinéraire dynamique basé sur la meilleure qualité de l'air, est proposée. Le service *DBAQ-Routing* détermine l'itinéraire le moins pollué en termes de concentration des différents types de polluants. Le choix de ce service est dû à l'impact très négatif de la pollution de l'air sur la santé. De récentes études épidémiologiques de l'Organisation Mondiale de la Santé (OMS), ont mises en évidence que la mort prématurée de millions de personnes (7 millions) est attribuable à la pollution de l'air [146].

4.4.1. Principe de fonctionnement du service *DBAQ-Routing*

Pour accéder au service *DBAQ-Routing*, l'utilisateur doit en premier lieu indiquer, via l'interface de l'application, sa destination, la liste des polluants à tenir en compte et la « zone acceptable » contenant l'ensemble des routes qu'il/elle accepte d'emprunter. En recevant la requête du client, le serveur d'application suit

la procédure détaillée sur la *Figure 33*. Dès la réception d'une réponse de la part du middleware, le serveur d'application détermine, pour chaque itinéraire, le niveau de pollution de l'air *APL* (Air Pollution Level) définit ainsi :

$$APL = \sum_{p \epsilon P} \sum_{S_i \epsilon S_p} \mu_p C_{S_i,norm} \quad (46)$$

Où P représente l'ensemble des polluants considérés, S_p l'ensemble des capteurs présent sur l'itinéraire et dont les données de mesures relatives au polluant $p \in P$ sont envoyées au serveur d'application, $C_{S_i,norm}$ est la valeur normalisée de la concentration observée par le capteur $S_i \epsilon S_p$, $\mu_p \epsilon [0,1]$ est un facteur de pondération représentant le niveau de toxicité du polluant p.

L'OMS a considérée 35 polluants dans sa publication des lignes directrices de la pollution de l'air [147]. Cependant, pour la détermination de l'indice de qualité de l'air (AQI, Air Quality Index), en général seul 6 types de polluants sont considérés ; il s'agit de l'ozone (O_3), des particules fines (PM2,5 et PM10), du dioxyde de souffre (SO_2), du monoxyde de carbone (CO) et du dioxyde d'azote (NO_2). En fonction des pays, différentes méthodes sont utilisées pour déterminer et interpréter l'indice AQI [148]. L'avantage de l'APL par rapport à l'AQI réside sur le fait que l'APL permet de déterminer le niveau de pollution en temps réel de l'ensemble des polluants cibles. Aussi, le facteur μ_p peut être modifié en fonction de la sensibilité de l'utilisateur par à un tel ou autre polluant. Ce qui n'est pas le cas pour la valeur du niveau standard des polluants utilisée pour l'AQI.

4.4.2. Simulation d'un cas d'utilisation

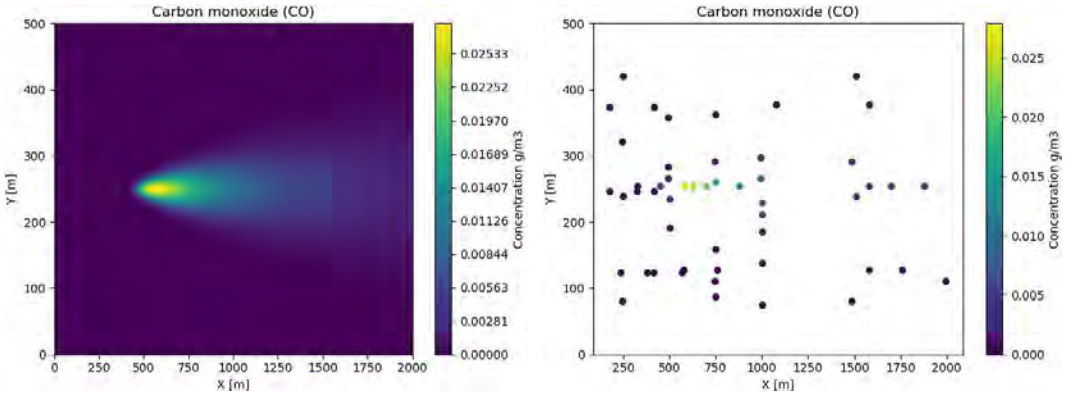
Dans ce cas d'utilisation, trois polluants sont considérés pour la détermination de l'*APL*. Il s'agit du monoxyde de carbone (CO), du dioxyde d'azote (NO_2) et des particules fines (PM2.5). On suppose que ces polluants sont émis depuis des points sources (usine, centrale électrique…). Le modèle gaussien rectiligne (gaussian plume model) [149] est utilisé pour simuler[2] la dispersion de ces polluants, c'est-à-dire détermination la concentration de chaque polluant au sol à partir d'un point source. Le *Tableau 12* fournie les détails des paramètres de simulation.

Tableau 12 : Paramètres de simulation modèle gaussien rectiligne

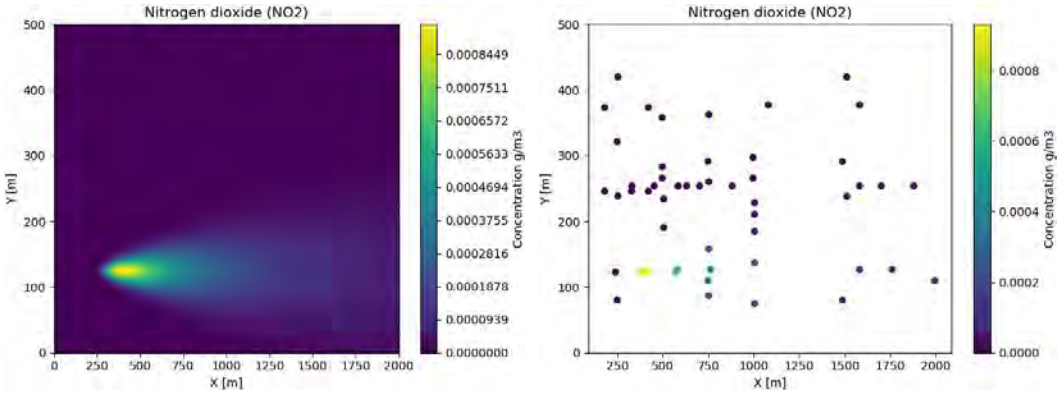
	Monoxyde de carbone (CO)	**Dioxyde d'azote (NO_2)**	**Particules fines (PM2.5)**
Coordonnées de la source	*x,y,z* =350,240,0	*x,y,z* =180,120,0	*x,y,z* =600,200,0
Débit (g/s/m^2)	150	5	0.25
Vitesse du vent (m/s)	7	7	7
Catégorie de la stabilité	*E*	*E*	*E*

Les *Figure 37.a*, *37.c* et *37.e* représentent respectivement les dispersions de CO, NO_2 et PM2.5 sur la « zone acceptable » via le modèle gaussien rectiligne.

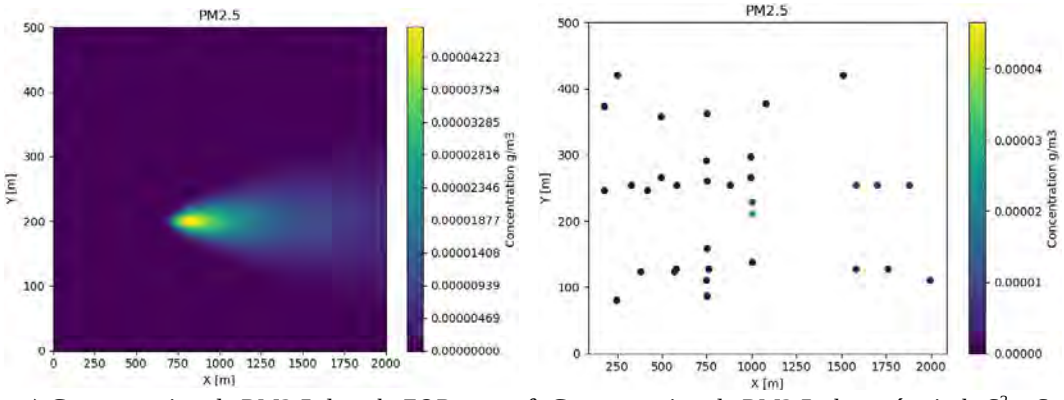
[2] https://github.com/lannymac/gaussianPlume



a) Concentration CO dans la ZOR *b) Concentration de CO observée via le S²aaS*



c) Concentration de NO_2 dans le ZOR *d) Concentration de NO_2 observée via le S²aaS*



e) Concentration de PM2.5 dans le ZOR *f) Concentration de PM2.5 observée via le S²aaS*

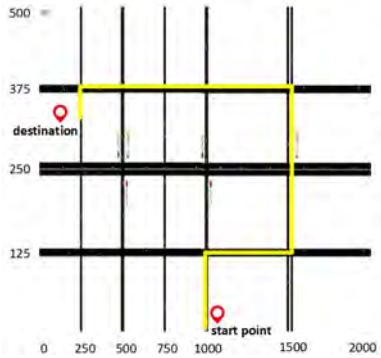
Figure 37 : Dispersion des polluants et résultats d'observation via le S²aaS

Le but du S²aaS est d'observer la dispersion des différents polluant en se basant sur les mesures des capteurs automobiles. Une extension du modèle de mobilité microscopique de Krauss [150] est utilisée pour décrire la dynamique de chaque véhicule dans la ZOR (voir *section 5.4.1*). En fonction de la qualité des capteurs, cinq classes de véhicules ont été définies. Ces cinq classes incorporent toutes des capteurs CO et NO_2. Cependant, seules les classes $C3, C4$ et $C5$ embarquent des capteurs PM2.5. La qualité de chaque capteur dépend des propriétés suivantes : sensibilité, temps de réponse, précision et plage de mesure. Les valeurs de ces propriétés sont obtenues en se référant aux fiches techniques des fabricants

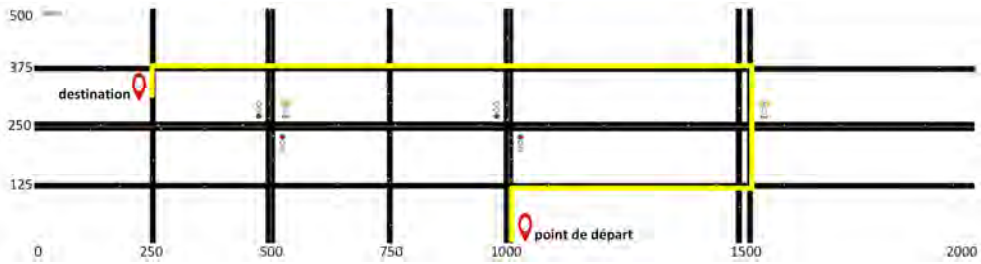
leader dans le marché des capteurs de qualité de l'air utilisés dans les véhicules [151]. Le modèle S^{2}aaS décrit plus haut (voir *section 4.3*) est implémentée via un script en Python et des modules de calcul scientifique (*numpy, sympy, itertools, matplotlib.*…). Durant la simulation, chaque véhicule publie toutes les 5s les propriétés de l'ensemble de ses capteurs vers la RSU. La ZOR consiste en une zone rectangulaire ($longueur = 2000m,\ largeur = 500m$) avec la présence de plusieurs routes, ponts, feu de circulations… Durant la simulation, le générateur d'exigence indique le nombre désiré de capteurs (50) pour chaque polluant. Le vecteur propre $w_{max} = [0.0525, 0.2817, 0.5740, 0.0918\ ]$ contient les facteurs de pondération de la sensibilité, du temps de réponse, de la précision et de la plage de mesure respectivement. Deux RSU sont sélectionnées pour effectuer les tâches décrites dans le modèle du S^{2}aaS (voir *section 4.3*).

Les *Figure 37.b, 37d* et *37.f* représentent respectivement les concentrations de CO, NO2 et PM2.5 observées par les capteurs sélectionnés. Les dispersions du CO et du NO2 sur les routes sont relativement bien observées par les véhicules. En effet, dans la simulation, l'ensemble des véhicules sont équipés de capteurs de CO et NO2. En contraste, la dispersion du PM2.5 n'est pas bien observée du fait de deux facteurs. En premier, le nombre de véhicules incorporant des capteurs PM2.5 est limité ; en effet, seul 32 mesures parmi les 50 requises ont pu être collectées. Ensuite, la source de pollution du PM2.5 se trouve dans une zone peu fréquentée par les véhicules. Le risque avec ce cas de figure est que l'application DBAQ-Routing indique un mauvais itinéraire. Pour remédier cela, des données d'historique et/ou des techniques de projection pourraient être utilisées [53], [152].

La *Figure 38* montre l'itinéraire choisi par l'application *DBAQ-Routing*.



a) Résolution carrée



b) Résolution réelle

Figure 38 : Indication de l'itinéraire présentant le plus faible APL

4.5. Guide d'implémentation sur Android Automotive

Le but de cette section est de montrer que le modèle S^{2}aaS proposé n'est pas seulement théorique et que son implémentation demeure faisable.

Android Automotive [153] est un nouveau système d'exploitation embarqué basé sur Android et conçu pour les véhicules. Contrairement à Android Auto, une application pour smartphone utilisant l'écran d'affichage de voiture comme seconde interface, Android Automotive est un système IVI (In-Vehicle Infotainment) fonctionnant sur un système SoC (System on Chip) automobile dédié.

4.5.1. Architecture d'Android Automotive

De nombreux sous-systèmes de voiture s'interconnectent avec le système IVI via différentes topologies de bus, ce qui rend difficile le développement d'un sous-système de communication générique. Ainsi, Android Automotive propose une couche d'abstraction matérielle (HAL, Hardware Abstraction Layer) qui fournit une interface cohérente avec le framework Android, quelle que soit la couche de transport physique. L'architecture d'Android Automotive, voir *Figure 39*, est composée de plusieurs couches, à savoir :

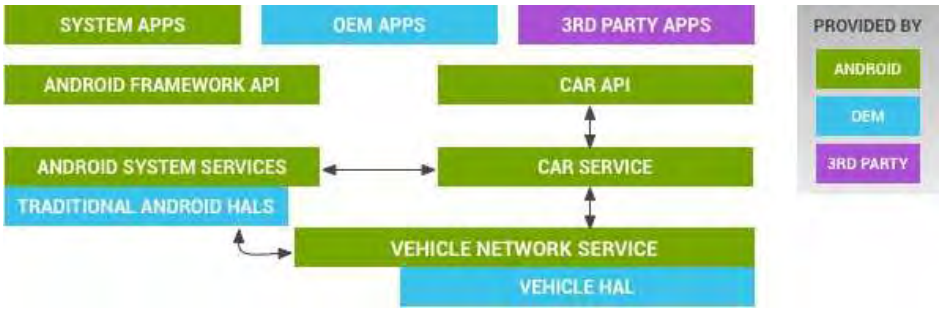


Figure 39 : Architecture d'Android Automotive [153]

- *Car API :* fournit une API () pour gérer la connectivité, les capteurs, les systèmes de HVAC, le niveau de carburant…
- *Car Service (CS) :* contient tous les services Android dédiés au véhicule tels que car-lib, odb2-lib, car-maps-placeholder…
- *Vehicle Network Service (VNS) :* contrôle l'accès au HAL du véhicule.
- *Vehicle HAL (VHAL) :* est l'interface entre les couches *Car Service* et *Vehicle Network Service*. Elle définit les propriétés OEM (Original Equipment Manufacturer) que les constructeurs peuvent implémenter. Exemple, les propriétés du capteur représentent les données réelles du capteur avec des interfaces d'accès en lecture, en écriture et en souscription.

Pour lire la sortie d'un capteur, l'interface VHAL propose un ensemble d'appels de type *get*, voir *Figure 40*.

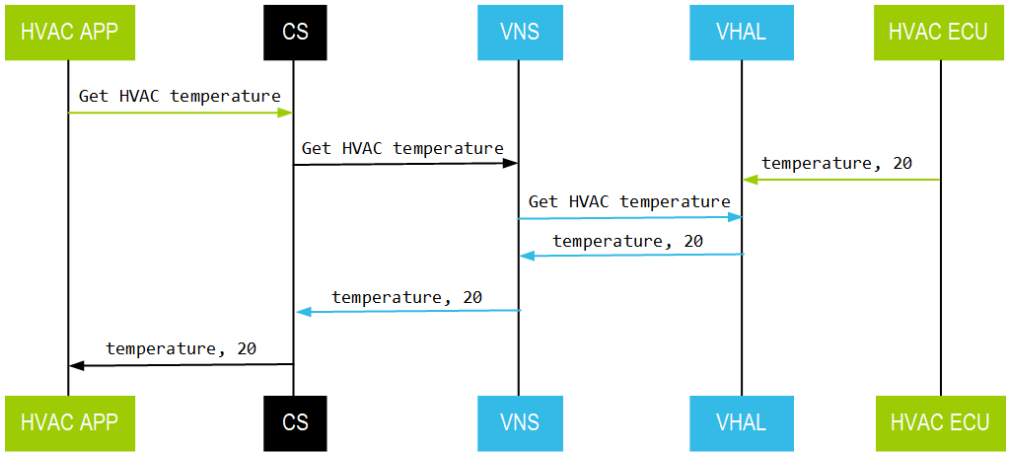


Figure 40 : Obtention de la température du système HVAC [153]

Android Automotive fournit une API nommée *CarSensorManager* pour le monitoring des données de capteurs automobiles. Cependant, cette API ne gère pas la persistance des données des capteurs et n'est pas conçue pour intégrer la sémantique de ces données. Afin d'implémenter des services/applications S^2aaS sur Android Automotive, plusieurs extensions du système Android doivent être intégrées.

4.5.2. Proposition d'extensions sur l'architecture d'Android Automotive

Les propriétés des capteurs au niveau du VHAL doivent être étendues afin d'intégrer certaines propriétés telles que la précision, le temps de réponse, la sensibilité… Ces informations permettront de représenter les propriétés du capteur de façon sémantique par le biais de l'ontologie SSN, voir le *Tableau 1*. Ensuite, une extension de l'API *CarSensorManager* optimisée pour l'application S^2aaS est proposée. Cette API, nommée *CarSensorManagerS2aaS*, est utilisée par exemple pour implémenter une application Android basée sur le S^2aaS. La *Figure 41* fournit en détails l'ensemble des extensions proposées.

4.5.2.1. L'API CarSensorManagerS2aaS

Cette API remplit deux rôles principaux : la gestion de la base de données des propriétés de capteurs automobiles et le service de requête de capteurs.

- *Gestion de la base de données des propriétés de capteurs automobiles*

Consiste au monitoring et à la mise à jour les valeurs des propriétés des capteurs dans une base de données. *SQLite* sera utilisé pour stocker les propriétés des capteurs. Contrairement aux serveurs de base de données traditionnelles comme MySQL ou PostgreSQL, l'intégralité de la base de données est stockée dans un fichier (et non sur un serveur distant). L'utilisation de la bibliothèque *Room Persistance* d'Android facilitera la gestion des bases de données *SQLite*.

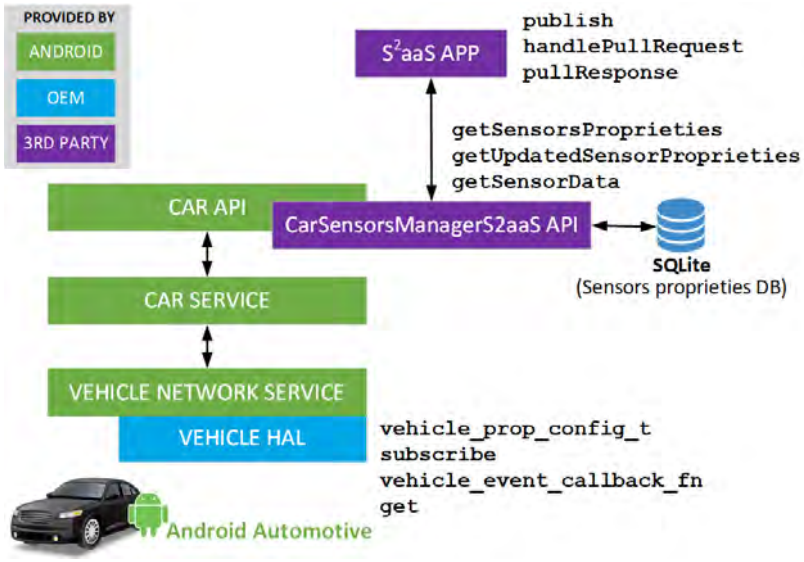


Figure 41 : Nouvelles extensions d'Android Automotive

Les interfaces VHAL suivantes seront impliquées dans la surveillance des capteurs de voiture.

```
vehicle_prop_config_t const *(*list_properties)(..., int* num_properties)
```

Cette interface est utilisée pour répertorier la configuration de toutes les propriétés prises en charge par la couche VHAL. On pourra ainsi obtenir par exemple la liste des capteurs disponibles dans la voiture.

```
(*subscribe)(..., int32_t prop, float sample_rate, int32_t zones)
```

Cette interface est utilisée pour surveiller le changement d'une propriété de capteur. Le VHAL appelle une fonction de l'API *CarSensorManagerS2aaS* pour exécuter une requête *SQLite* afin de mettre à jour la base de données des propriétés des capteurs.

```
(*vehicle_event_callback_fn)(const vehicle_prop_value_t *event_data)
```

Cette interface sert à notifier le changement de valeur des propriétés des capteurs souscrites.

- *Service de requête de capteurs*

Pour l'application S²aaS, trois appels de type *get* sont proposés : *getSensorsProprieties*, *getUpdatedSensorProprieties* et *getSensorData*.

L'appel *getSensorsProprieties* est une opération qui permet à l'application S²aaS d'obtenir la valeur de toutes les propriétés de capteurs stockées dans la base de données SQLite par le biais de l'API *CarSensorManagerS2aaS*.

L'appel *getUpdatedSensorProprieties* est une opération qui permet à l'application S²aaS d'obtenir la valeur des propriétés mise à jour d'un capteur dans *SQlite* par le biais de l'API *CarSensorManagerS2aaS*.

L'appel *getSensorData* est une opération qui permet à l'application S^{2}aaS d'obtenir la valeur de sortie d'un capteur spécifique. Lorsque l'API *CarSensorManagerS2aaS* reçoit un cet appel, l'interface VHAL suivante est appelée pour lire effectivement la valeur de la sortie du capteur :

```
(*get)(..., vehicle_prop_value_t *data)
```

4.5.2.2. S^{2}aaS App

Cette application utilise essentiellement l'API *CarSensorManagerS2aaS*. Selon le modèle fonctionnel du S^{2}aaS, voir *Figure 33*, le véhicule publie les propriétés de ses capteurs incorporés vers la RSU de façon régulière. La RSU envoie également une requête au véhicule afin d'obtenir des données de capteurs spécifiques via méthode *pull*. Les méthodes suivantes sont utilisées pour implémenter l'interaction le véhicule et la RSU.

La méthode *publish* est utilisée par le véhicule pour envoyer à la RSU les valeurs mises à jour des propriétés des capteurs.

La méthode *handlePullReques*t permet traiter la requête d'accès aux données de capteurs par la RSU.

La méthode *pullResponse* est utilisée pour envoyer une réponse à la RSU après traitement de sa requête d'accès aux données de capteurs.

Pour implémenter l'API *CarSensorManagerS2aaS*, le code source d'Android Automotive doit être personnalisé. Ceci est possible grâce au projet AOSP (Android Open Source Project), qui permet aux développeurs de créer un système Android personnalisé appelé aussi custom ROM.

4.6. Conclusion

Dans ce chapitre, un modèle de LM basé sur l'utilisation des capacités d'observation, de perception et de communication des véhicules modernes est proposé. Ce modèle, nommé IVSN, permet de transformer le véhicule en une plateforme numérique capable d'apprendre, penser et comprendre les systèmes physiques par lui-même. Les avantages de l'IVSN ont été exploités dans la proposition de deux modèles de services S^{2}aaS. Ces services permettent d'inférer de nouvelles connaissances à partir des données de capteurs automobiles. Un exemple d'application pratique du S^{2}aaS est également proposé.

Des tests de performance basés sur des modèles analytiques et sur des simulations montrent que les modèles de communication proposés pour le S^{2}aaS présentent de meilleures performances comparées aux modèles de Perera et al. [35] et Hsu et al. [36].

Dans le chapitre suivant, un modèle réseau adapté pour le S^{2}aaS et ainsi que son prototypage à base de la technologie SDN seront proposés.

Chapitre 5 : Prototypage d'un réseau SDN véhiculaire

Ce chapitre détaille l'architecture réseau SDN retenue pour la communication véhiculaire dans un contexte Sensing as a Service. L'ensemble du réseau est géré par trois types de contrôleurs : le contrôleur SDNRAN, le contrôleur SDNBACKBONE et le contrôleur SDNVANET. Des tests de performance ont permis de valider l'ensemble des schémas et fonctions réseaux proposés.

5.1. Proposition d'une architecture réseau véhiculaire SDN

L'architecture proposée a pour but d'offrir une meilleure gestion de la mobilité pour les véhicules tout en optimisant l'accès aux services S^2aaS. Pour ce faire, chaque entité réseau est dotée d'un ou de plusieurs fonctionnalités et exécute des tâches spécifiques, le tout en harmonie avec les autres entités réseau. Ainsi, les avantages du SDN seront pleinement exploités pour offrir un handover sans couture et une meilleure gestion du flux.

5.1.1. Description de l'architecture

L'architecture SDN proposée, *Figure 42*, est une architecture distribuée composée de plusieurs couches, à savoir : la couche fog, le backbone SDN et la couche des contrôleurs.

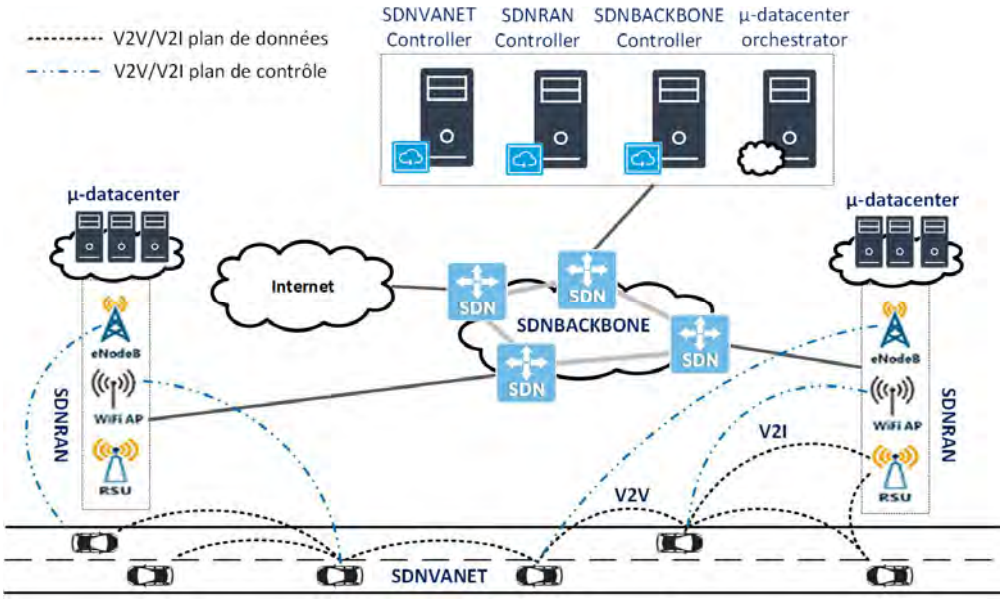


Figure 42 : Architecture globale du réseau SDN véhiculaire

5.1.1.1. La couche fog

Cette couche est constituée de nœuds fog mobiles (véhicules), d'un Réseau d'Accès Radio (RAN, Radio Acces Network) et d'un micro datacenter.

- *Nœuds fog mobiles*

C'est l'ensemble des véhicules échangeant des données avec les autres entités du fog via une communication V2V, V2I et V2X. Ces véhicules sont dotés de plusieurs interfaces sans fil gérées par un hyperviseur, implémenté dans l'ordinateur de bord du véhicule, et par le contrôleur SDNVANET.

- *Radio Access Network (RAN)*

Regroupe l'ensemble des technologies d'accès radio utilisées par les nœuds fog mobiles. Il s'agit des technologies DSRC (802.11p, WAVE), WLAN (802.11n/ac) et cellulaires (3G, LTE). Le RAN est ainsi doté de RSU, Points d'Accès (AP) et de Station de Base (eNode B) gérés par le contrôleur SDNRAN.

- *Micro Datacenter*

Il est constitué d'un ensemble de micro serveurs servant à la collecte, au stockage, au traitement et l'analyse de données issues des nœuds fog mobiles. Cela permet de fournir des services dédiés à l'IoV, notamment le S^2aaS. La conteneurisation peut être utilisée ici à la place de la virtualisation de machines. Les ressources micro datacenter seront gérées par un orchestrateur de conteneurs.

5.1.1.2. Le backbone SDN

Le backbone SDN est un réseau distribué composé essentiellement de switches SDN. Ce réseau permet de créer une architecture plate, c'est-à-dire exempt de toute hiérarchie et le plan de données est directement acheminé vers Internet ou vers un autre réseau sans passer par une entité centrale. Cet acheminement est assuré par les switches SDN sous le contrôle du contrôleur SDNBACKBONE.

5.1.1.3. La couche des contrôleurs

Toute l'intelligence du réseau est centralisée au niveau de la couche des contrôleurs. Ces derniers ont une vision globale du réseau et gèrent la mobilité des nœuds fog, les ressources réseau et les micros datacenter.

- *Contrôleur SDNRAN*

Ce contrôleur gère le plan de contrôle du RAN et du handover des nœuds fog mobiles. Ce contrôleur orchestre également le mécanisme d'allocation ou de libération de ressources au niveau du RAN.

- *Contrôleur SDNBACKBONE*

C'est un contrôleur SDN classique, c'est à dire gère le plan de contrôle filaire au niveau des switches SDN du backbone.

- *Contrôleur SDNVANET*

C'est un contrôleur qui gère exclusivement le plan de contrôle du VANET (communication V2V et V2I). Ainsi, pour routage, le statut et la topologie des véhicules sont utilisés.

- *Orchestrateur des micro Datacenter*

Gère l'allocation ou la libération de ressources au niveau des serveurs des micro Datacenter. Cette orchestration se fait par le biais d'instanciation, de réplication, de reconfiguration et de migration de container au niveau des serveurs.

5.1.2. Stratégie d'association flux/interface

Afin d'assurer une meilleure transmission des paquets, un certain nombre de stratégies de gestion des interfaces sans fil seront mises en place. Une stratégie d'association flux/interface au sein des nœuds fog mobiles permet de choisir l'interface réseau la mieux adaptée pour transmettre un flux. Ainsi, pour chaque interface, on définira un ensemble de flux bien spécifique qui va y transiter.

- *Interface 802.11p*

Cette interface est optimisée pour la communication véhiculaire en mode ad hoc, il est donc plus judicieux de l'utiliser pour assurer le transport du plan de données de l'IVSN. Ce plan de données comprend les données échangées entre les véhicules en mode broadcast, et les échanges entre les véhicules et la RSU en mode unicast multisaut. Cependant, le plan de contrôle de l'IVSN ne pourra pas transiter via cette interface du fait de la difficulté à maintenir une connexion stable.

- *Interface WiFi (802.11n/ac)*

Le WiFi est caractérisé par une grande robustesse et un coût d'acquisition, de déploiement et de maintenance relativement bas. La connectivité WiFi sera choisie pour le transport du plan de contrôle des switches SDNVANET incorporés dans les véhicules. Ce plan de contrôle est essentiellement constitué d'instructions venant du contrôleur SDNVANET et d'informations liées au statut des véhicules (positions, vitesse, état des interfaces radio…). Cependant, le WiFi ayant une portée limitée, il est donc pas omniprésent. Par conséquent, afin de garantir aux véhicules un accès continue au contrôleur SDNVANET, il est nécessaire de trouver une option secondaire de communication en cas d'indisponibilité du WiFi.

- *Interface 3G/LTE*

Cette interface sera utilisée comme interface secondaire pour la transmission des plans de contrôle du SDNVANET. En effet, la technologie LTE est robuste et adaptée pour les nœuds mobiles. En outre, elle est omniprésente. Cependant le coût de connectivité relativement élevé de la 4G, fait qu'elle ne sera pas l'interface de communication à privilégier. Un schéma de mobilité est donc nécessaire pour gérer la handover entre le WiFi et le LTE.

5.1.3. Détails du prototypage du réseau SDN

La mise en place du réseau SDN de test se fera à travers le déploiement du SDNBACKBONE, du SDNRAN et du SDNVANET. La *Figure 43* montre le prototype retenu.

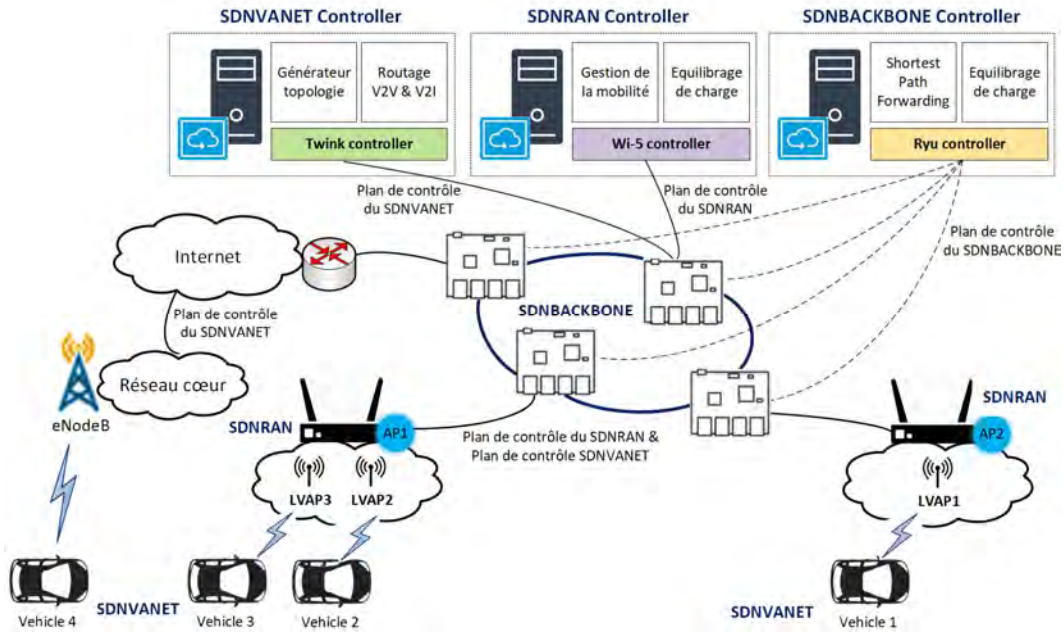


Figure 43 : Architecture de test retenue

Le déploiement du réseau SDN de test devra suivre un certain ordre. Tout d'abord le backbone devra être mise en place, ensuite le réseau d'accès radio et enfin les switches SDNVANET incorporés dans les véhicules.

5.1.3.1. Déploiement du SDNBACKBONE

Le backbone SDN est constitué de quatre switches SDN Zodiac FX et du contrôleur SDNBACKBONE basé sur le framework Ryu [154]. Une application de routage nommée PureSDN[3] est implémentée au niveau de ce contrôleur.

Ryu, codé en Python, offre un riche ensemble d'applications SDN permettant d'optimiser la gestion du réseau. Ryu offre la possibilité de modifier les applications natives ou d'en implémenter d'autres. L'application PureSDN est une application Ryu de gestion du trafic réseau proposée par Huang MaChi. PureSDN se base sur des informations réseau telles que la topologie et la bande passante disponible sur les différents liens pour déterminer les meilleures routes.

Le switch SDN Zodiac FX est constitué de quatre ports, et est destiné pour des projets éducatifs et de recherche. Grâce son bas coût, les switches Zodiac FX permettent aux chercheurs de mener des expériences sur des équipements réels.

[3] https://github.com/Huangmachi/PureSDN

5.1.3.2. Déploiement du SDNRAN

Le déploiement du SDNRAN sera fait sur la base de l'architecture du projet Wi-5[4] [155]. Wi-5 est un fork de Odin [156] qui est un framework WLAN d'entreprise programmable qui permet d'implémenter une politique de gestion de réseaux WiFi via la création de point d'accès virtuels (LVAP, Light Virtual Access Point). Le LVAP permet de virtualiser les associations entre l'AP physique et les stations clientes (STA). Le LVAP est représenté par un tuple constitué de l'adresse MAC de la station (STA_MAC), de l'adresse IP de la station (STA_IP), d'un BSSID virtuel de l'AP (AP_vBSSID) et d'un SSID virtuel de l'AP (AP_vSSID).

$$LVAP = (STA\_MAC, STA\_IP, AP\_vBSSID, AP\_vSSID)$$

Wi-5 propose des améliorations et de novelles fonctionnalité à Odin. Son architecture consiste en un ensemble de RAN (Radio Acces Network) gérés par un contrôleur SDN Wi-5.

Le RAN est constitué d'Agents Odin installés sur des points d'accès WiFi TP-LINK AC1750 v2 (Archer C7). Open vSwitch est utilisé pour la création de commutateur virtuel sur les AP. Click Modular Router est également installé sur chaque AP afin d'assurer des fonctions de routage en interne.

Le Wi-5 Controller est implémenté à partir du du framework SDN floodlight. Le protocole Openflow sert à contrôler les commutateurs virtuels Open vSwitch créés sur les AP. Le protocole Odin est également utilisé. C'est une extension d'Openflow qui sert à contrôler les interfaces IEEE 802.11n/ac des AP.

5.1.3.3. Déploiement du SDNVANET

Le déploiement du SDNVANET se fera grâce au module OpenFlow Twink[5]. Ce dernier, écrit en Python, propose un ensemble de fonctions qui permettent la création et le parsing de messages OpenFlow. Twink peut être utilisé pour implémenter aussi bien un contrôleur que des switches SDN. En outre, Twink facilite la création d'extensions du protocole OpenFlow. Ainsi, l'implémentation du contrôleur SDNVANET et des switches SDNVANET, constitués d'ordinateur monocarte de type Raspberry Pi, s'est fait à la base de ce module.

5.2. Schémas de routage du SDNBACKBONE

Le Backbone SDN utilise PureSDN comme application de routage. Cette dernière est constituée du module *Network Awareness* chargé de la collecte des informations relatives au réseau (nombre de switches SDN, état des liens…), et du module *Network Monitor* chargé de la collecte des informations relative au trafic réseau. PureSDN propose des algorithmes de routage utilisant comme critère le nombre de sauts ou la bande passante libre.

[4] https://github.com/Wi5/odin-wi5
[5] https://github.com/hkwi/twink

5.2.1. Description des schémas de routage

Le backbone est composé de switches SDN (nœuds) et d'un ensemble de liens entre les switches. Ce backbone peut être représentée par un graphe orienté $G(N, L)$ composé de nœuds $n \in N$ et de liens $l \in L$. Ces liens sont caractérisés par leur nœud d'origine a_l, leur nœud de destination b_l et leurs attributs ou poids w_l. Le backbone peut transporter des flux $f \in F$ caractérisés par leur origine $src_f \in N$ et leur destination $dst_f \in N$. Pour chaque flux f, il existe un ensemble de chemins $p_f \in P$ allant de la source $src_f \in N$ à la destination $dst_f \in N$.

5.2.1.1. Routage basé sur le nombre de sauts (weight=hop)

Ce type de routage considère comme meilleure route celle qui présente le moins de sauts (hop) entre la source et la destination. Soient :

$w_{l,hop} = 1$, le poids de tous les liens

$p_{f,hop} = \{p_f^k\}$, les K chemins les plus courts pour le flux f en termes de sauts, où $k = 1, 2, \dots, K$ et p_f^k est un chemin simple, c'est-à-dire ne passant pas deux fois par un même nœud. On a $p_{f,hop} \in p_f$.

La détermination de $p_{f,hop}$ est basée sur l'algorithme de Jin Y. Yen [157].

PureSDN se base sur le module NetworkX [158] pour effectuer le classement et la sélection de la meilleure route. NetworkX est un module python dédié à la manipulation et l'étude de réseaux complexes. En se basant sur la topologie du réseau, NetworkX génère un graphe orienté et détermine l'ensemble des chemins simples pour chaque paire de source/destination.

5.2.1.2. Routage basé sur la bande passante libre (weight=bw)

Ce type de routage considère comme meilleure route celle qui présente le plus de bande passante libre. Le but est d'équilibrer la charge réseau en distribuant le flux sur l'ensemble des routes disponibles.

Soient :

C_l : la capacité maximale du lien l

$w_{l,bw}$: la bande passante libre du lien l

$N_{p_f^k}$: le nombre total de nœud dans p_f^k

$p_{f,bw} = p_{f,bw}^1$: le chemin présentant la valeur maximale de la bande passante minimale des liens dans $p_{f,hop}$. On a $p_{f,bw} \in p_{f,hop}$.

La détermination de $p_{f,bw}^1$ est basée sur l'*Algorithme 2*.

Algorithme 2 sélection du meilleur chemin basé sur la bande passante libre

Require : $G, f, p_{f,hop}$
1: **Output:** $p_{f,bw}^{1}$
2: $max\_bw\_of\_paths \leftarrow 0$
3: **for** each $p_f^k \in p_{f,hop}$ **do**
4: $min\_bw \leftarrow C_l$
5: **for** $i = 1$ to $N_{p_f^k}$ **do**
6: $bw \leftarrow w_{l,bw}$ of link l: $[i, i+1] \in p_f^k$
7: $min\_bw \leftarrow min(bw, min\_bw)$
8: **if** $min\_bw > max\_bw\_of\_paths$ **then**
9: $max\_bw\_of\_paths \leftarrow min\_bw$
10: $p_f^1 \leftarrow p_f^k$
11: **end if**
12: **end for**
13: **end for**
14: **return** $p_{f,bw}^{1}$

PureSDN se base sur les statistiques OpenFlow pour déterminer la bande passante libre sur les liens des switches SDN. En effet, ces statistiques fournissent entre autres la capacité et le nombre de paquets transmis sur les port des switches.

Pour déterminer le débit de transmission d_{Tx} (octets/s) sur le port d'un switch, PureSDN utilise la formule suivante :

$$d_{Tx} = \frac{n_j^{Tx} - n_i^{Tx}}{t_j - t_i} \tag{47}$$

Où n_i^{Tx} et n_j^{Tx} désignent le nombre total d'octets transmis sur le port respectivement à l'instant t_i et t_j. Connaissant la capacité C (*Kbits/s*) du port, on peut en déduire la bande passante libre BW_{free} sur ce port :

$$BW_{free} = \max\left(C - \frac{8d_{Tx}}{1000}, 0\right) \tag{48}$$

5.2.1.3. Routage basé sur la latence (weight=lat)

PureSDN ne gère pas ce type de routage. Néanmoins, plusieurs travaux [159]–[161] intègrent le critère de délai pour le routage. Soient :

$w_{l,delay}$: le délai du lien l

$p_{f,delay} = p_{f,delay}^{1}$: le chemin présentant le délai minimal dans $p_{f,hop}$. $p_{f,delay} \in p_{f,hop}$.

$$p_{f,delay}^{1} = min\left(\sum_{p_f^k \in p_{f,hop}} \sum_{l \in p_f^k} w_{l,delay}\right) \tag{49}$$

Notons que OpenFlow n'intègre pas la mesure de délai dans ses spécifications. Toutefois, des techniques de mesure sont proposées dans la littérature [162], [163]. Cependant, ces dernières ne sont pas très fiables et engendrent des charges réseau. Pour pallier cela, dans [161], les auteurs proposent une technique d'estimation du délai basée sur des méthodes analytiques et empiriques dépendant du matériel/plateforme. Ce qui constitue une limite pour son utilisation. Afin de garantir une bonne fiabilité des applications de routage sur le contrôleur SDNBACKBONE, le routage basé sur le délai ne sera pas implémenté.

5.2.2. Proposition de fonctionnalités additionnelles pour PureSDN

PureSDN se limite au routage de paquets entre les différents switches SDN du backbone, et ne gère pas la commutation de paquets des hôtes reliés aux switches. Ainsi, il est nécessaire d'implémenter sur le contrôleur SDN une fonction de commutation, et aussi, une fonction de broadcast.

5.2.2.1. Proposition d'une fonction de commutation

La table d'accès de l'application PureSDN est un dictionnaire Python au format suivant :

$$\{(sw, port): (ip, mac)\}$$

La fonction de commutation proposée consiste, d'une part, à consulter cette table d'accès afin d'identifier le port du switch où est relié le destinataire du paquet, et d'autre part, à donner l'ordre de transfert de ce paquet vers ce port.

Cependant, cette méthode de commutation ne résout pas la gestion de hôtes multiples sur un même port du switch. En effet, le format de la table d'accès ne permet d'associer qu'un seul couple (ip, mac) pour chaque port du switch. Pour palier cela, dans le nouveau format proposé, chaque port du switch est associé à une liste de tuples (ip, mac) de l'ensemble des hôtes qui y sont liés.

$$\{(sw, port): [(ip_{host1}, mac_{host1}), (ip_{host2}, mac_{host2}) \dots (ip_{hostn}, mac_{hostn})]\}$$

5.2.2.2. Proposition d'une fonction de diffusion des paquets DHCP

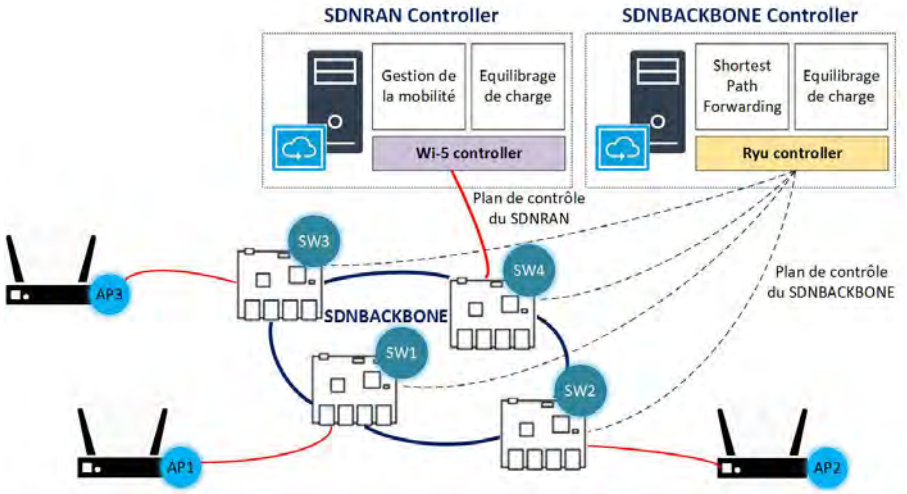
PurseSDN ne propose que deux méthodes de diffusion de paquets. La première, la méthode par défaut, et est optimisée pour le protocole ARP. Elle ne diffuse les paquets que sur les ports dont les hôtes ne sont pas encore identifiés. Le serveur DHCP étant déjà identifié, ne pourra pas, par conséquent, recevoir des messages de diffusion via cette méthode de diffusion. La fonction de commutation de paquets DHCP proposée consiste donc à forcer le contrôleur à utiliser la seconde méthode de diffusion. Cette dernière est basée sur l'inondation.

5.2.3. Evaluation des performances

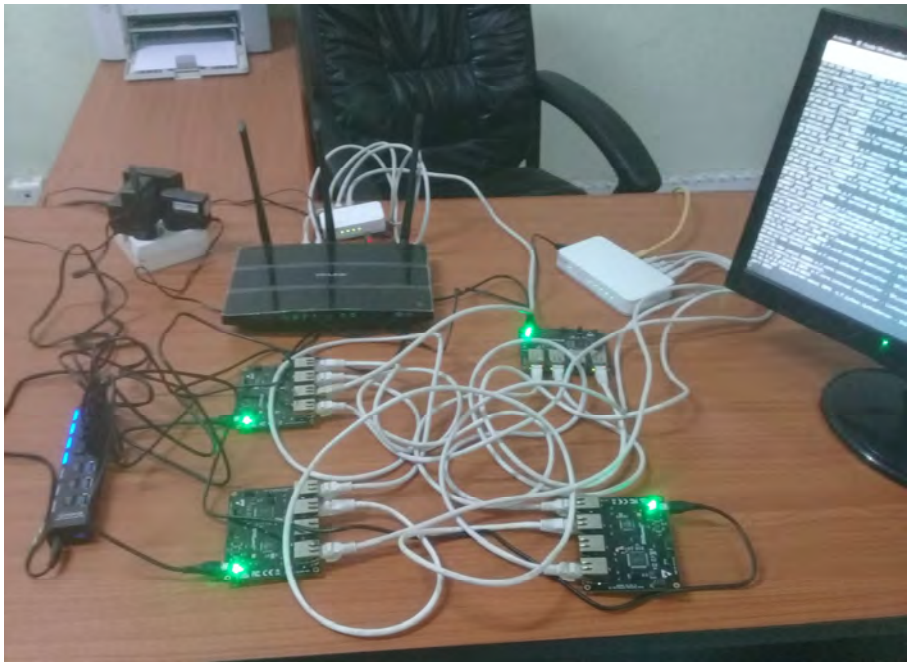
Deux types de routage sont testés au niveau du contrôleur SDNBACKBONE ; il s'agit du routage basé sur le nombre de sauts et le routage basé sur la bande passante libre.

5.2.3.1. *Détails des expérimentations*

Afin d'évaluer les performances réseau des schémas de routage, trois points d'accès (AP1, AP2 et AP3) sont utilisés pour communiquer avec le contrôleur SDNRAN. Le but est d'observer le comportement du backbone SDN selon sa sollicitation. La *Figure 44* fournit les détails de l'architecture à évaluer.



a) Vue schématique



a) Vue du banc de test

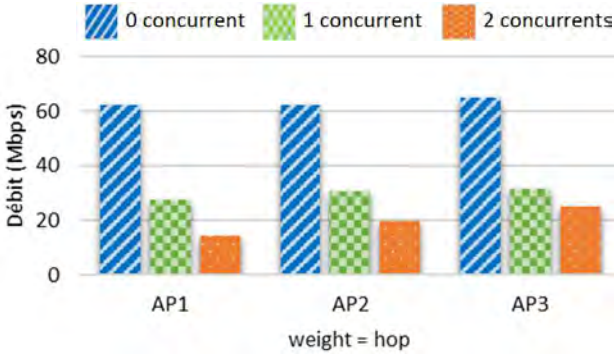
Figure 44 : Détails l'architecture à évaluer

Les AP seront testés individuellement (0 concurrent), puis deux à deux (1 concurrent) et enfin à trois (2 concurrents). Cela permettra d'évaluer l'impact de la montée en charge réseau sur les performances du backbone pour chacun des

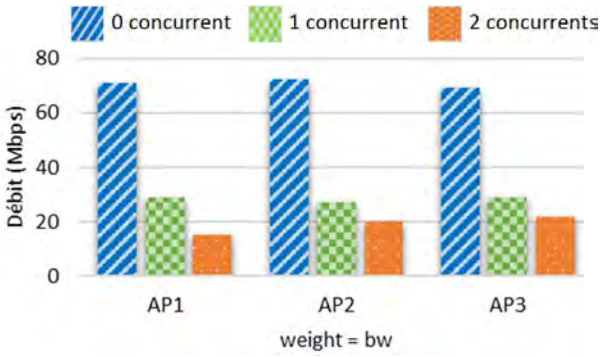
schémas de routage à tester. L'outil *iperf* est utilisé pour mesurer le débit de transmission, et l'outil *ping* pour mesurer le délai (RTT). Notons que le débit maximal des interfaces Ethernets des switches Zodiac FX est de 100Mbps.

5.2.3.2. Résultats et discussion

La *Figure 45* donne les résultats des tests de performance relative au débit transmission des paquets entre les différents AP et le contrôleur SDNRAN.



a) Critère de sélection du chemin : nombre de sauts



b) Critère de sélection du chemin : bande passante libre

Figure 45 : Débit de transmission des paquets des AP vers le contrôleur SDNRAN

Lorsque le critère de routage est le nombre de sauts (*weight=hop*), *Figure 45.a*, on a les performances suivantes en termes de débit :

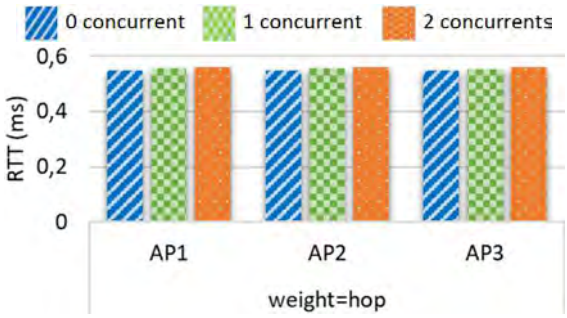
- *AP1 :* débit de 62,1 ; 27,5 et 14,41Mbps pour respectivement comme nombre de concurrents 0, 1 et 2.
- *AP2 :* débit de 62,3 ; 30,4 et 19,6Mbps pour respectivement comme nombre de concurrents 0, 1 et 2.
- *AP3 :* débit de 65,1 ; 31,5 et 25,1Mbps pour respectivement comme nombre de concurrents 0, 1 et 2.

Lorsque le critère de routage est la bande passante libre (*weight=bw*), *Figure 45.b*, on a les performances suivantes en termes de débit :

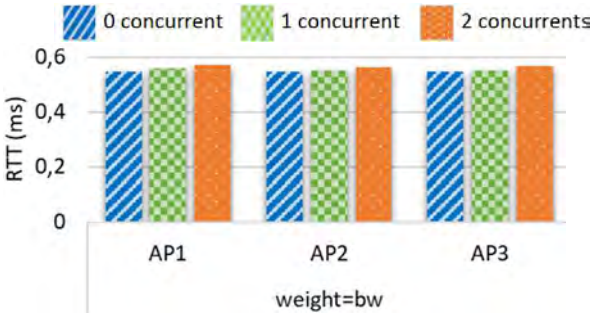
- *AP1 :* débit de 71,1 ; 28,8 et 15,1Mbps pour respectivement comme nombre de concurrents 0, 1 et 2.
- *AP2 :* débit de 72,4 ; 27,1 et 20,1Mbps pour respectivement comme nombre de concurrents 0, 1 et 2.
- *AP3 :* débit de 69,5 ; 28,8 et 21,6Mbps pour respectivement comme nombre de concurrents 0, 1 et 2.

Le constat général est que le débit de transmission des AP diminue en fonction du nombre de concurrents, c'est-à-dire lorsque la charge réseau augmente. Cela s'explique du fait que la bande passante du backbone est partagée entre les AP concurrents. On peut noter également que le switch SW4 partage sa bande passante pour transporter le flux de l'ensemble des AP vers le contrôleur SDNRAN. Ce qui explique cette diminution progressive du débit.

Les résultats présentés sur la *Figure 46* montrent que les schémas de routage et le nombre de concurrents sur le backbone impactent faiblement sur le délai de transmission (0,550ms en moyenne). La principale raison est que le temps de traitement des paquets n'est pas directement lié au nombre de concurrents. En effet, le calcul des routes se fait niveau du contrôleur SDNBACKBONE, et les switches SDN ne font que consulter leurs tables de flux et transférer les paquets. Ce processus est relativement à temps constant. Cependant, une longue file d'attente peut apparaitre lorsque les switches SND sont sursollicités.



a) Critère de sélection du chemin : nombre de sauts



b) Critère de sélection du chemin : bande passante libre

Figure 46 : Délai de transmission des paquets AP vers le contrôleur SDNRAN

En somme, on retiendra que les schémas de routage basés sur le nombre de sauts et sur la bande passante libre présentent quasiment les mêmes performances. Néanmoins, une différence pourra être observée lors des montés en charges. Dans ce cas, le schéma de routage basé sur la bande passante libre permettra de mieux gérer cette surcharge réseau en la distribuant (équilibrage de charge) sur l'ensemble des switches SDN. Vu les caractéristiques de la communication véhiculaire, une montée en charge réseau peut vite arriver du fait de la densité variable des véhicules. En effet, les feux de circulation et les points d'intersection sont des facteurs de création de congestion réseau.

5.3. Schémas de mobilité du SDNRAN

Deux entités seront responsables de la gestion de la mobilité des véhicules. Nous aurons des schémas de mobilité horizontale gérés par le réseau (contrôleur SDNRAN) et un schéma de mobilité verticale géré par le terminal (véhicule). Ce dernier cas de mobilité est une alternative, et n'est à utiliser que lorsque la connectivité du véhicule ne peut plus être gérée par le contrôleur SDNRAN.

5.3.1. Mobilité gérée par le Contrôleur SDNRAN

Deux approches de gestion du handover seront testées, à savoir des schémas de mobilités réactifs et des schémas de mobilité proactifs. Les applications SDN de mobilité réactives sont basées sur des triggers/callback pour traiter les événements réseau, tandis que les applications SDN de mobilité proactives sont basées sur le traitement à intervalle régulier d'informations collectées sur le RAN.

5.3.1.1. Schémas de mobilité réactifs

La gestion de la mobilité se fera à l'aide d'applications Odin réactives. La puissance du signal WiFi reçue sera la principale grandeur déclenchant (trigger) l'appel de fonctions de callback. Ainsi, lorsque le niveau de puissance du signal d'une station cliente (véhicule) atteint une valeur seuil critique, le Odin Agent (AP) le signale au Contrôleur SDNRAN. Le traitement de cet événement est assuré par une application dérivée du *Wi-5 Mobility Manager*. L'équilibrage de charge entre les AP est également prise en compte via l'application *Wi-5 Load Balancer*. Cependant, il n'existe aucune coordination ces deux applications.

Selon la manière dont les canaux WiFi sont configurés, deux types de handover peuvent avoir lieu.

- *Handover intracanal*

La procédure de handover intracanal est caractérisée par le non changement de canal lorsque la station cliente (véhicule) quitte son AP d'origine pour s'associer à un autre AP. Cette procédure demeure totalement transparente à la station, par conséquent, aucune reconfiguration réseau ne se fera à son niveau. La *Figure 47* détaille la procédure de handover intracanal.

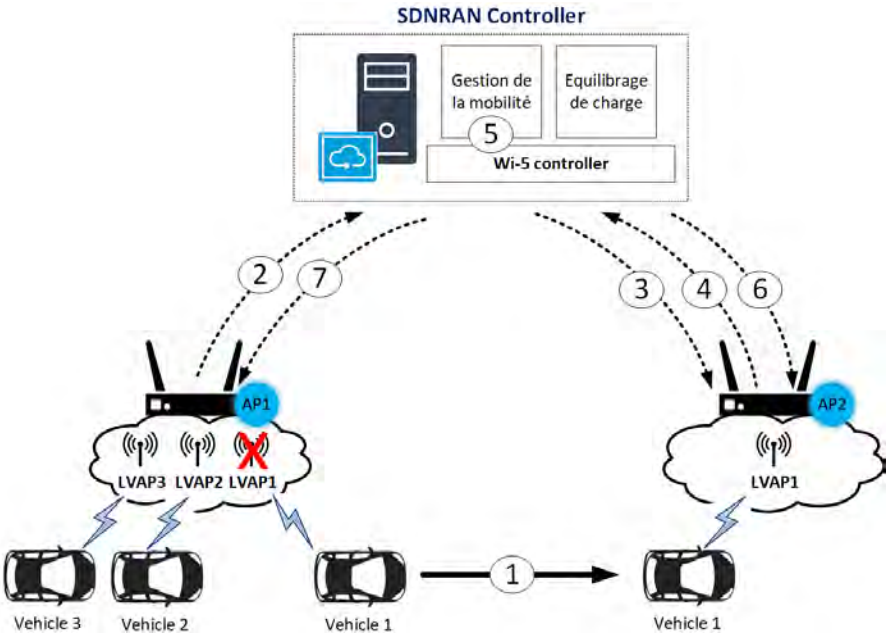


Figure 47 : Schéma de handover intracanal

Lorsque le véhicule (STA) s'éloigne de son AP d'origine ①, ce dernier (AP1) détecte que la puissance du signal de la STA est en deçà d'une valeur seuil définie. Par conséquent l'AP1 envoie un message *PUBLISH* au Contrôleur SDNRAN ②. A la réception de ce message, le contrôleur envoie un message *Scan Request* aux AP voisins ③. Dans notre cas, c'est l'AP2 qui recevra ce message et effectuera un scan puis enverra un message Scan *Response* au contrôleur SDNRAN ④. Lorsque le contrôleur reçoit les réponses des AP voisins, ce dernier exécute une fonction de sélection du meilleur AP, c'est-à-dire celui fournissant une plus grande puissance de réception pour la station cliente ⑤. Une fois le meilleur AP sélectionné, le contrôleur effectue alors une migration de LVAP. Pour ce faire, le contrôleur envoie à l'AP sélectionné (ici AP2) une commande de création du LVAP associé à la station cliente concernée ⑥, puis envoie une commande de suppression de ce LVAP à l'AP d'origine (ici AP1) ⑦.

A la fin de la procédure de handover, la station cliente est associée à l'AP2 « sans qu'elle en prenne conscience ».

- *Handover intercanal*

L'utilisation d'un même canal pour l'ensemble des AP peut engendrer des problèmes d'interférence et de scalabilité [164]. Dans Wi-5 une application d'assignement de canal (Applications for Channel Assignment) est utilisée pour optimiser l'assignement de canaux aux AP. Notons qu'un handover intercanal engendre l'échange de messages L2 dû au changement de canal par l'interface WiFi de la station. Néanmoins, cela reste transparent pour les couches réseau supérieures. La *Figure 48* détaille la procédure de handover intercanal.

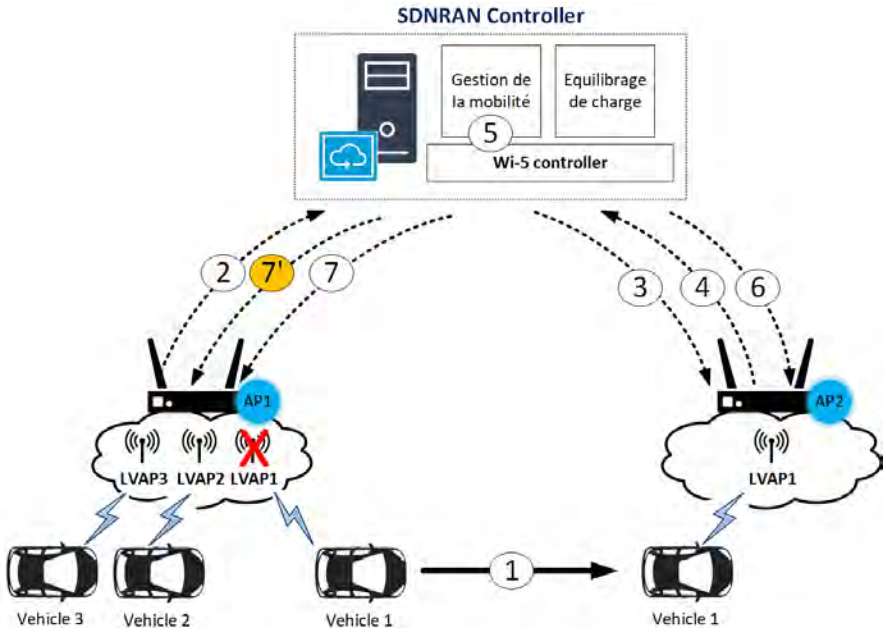


Figure 48 : Schéma de handover intercanal

Ce schéma de handover est quasi identique à celui du handover intracanal. La différence réside sur l'envoi, par le Controller SDNRAN, d'une commande *send CSA* à l'AP d'origine (7′). A la réception de cette commande, l'AP envoie une série de messages CSA (Channel Switch Announcement) à la station. Cette dernière fait basculer alors son interface WiFi vers le nouveau canal indiqué dans le message CSA. Une commande de suppression de ce LVAP à l'AP d'origine (ici AP1) ⑦ clos la procédure de handover.

A la fin de la procédure de handover, la station cliente « pense » avoir simplement effectuée un changement de canal.

5.3.1.2. Schéma de mobilité proactif

Ce schéma, dérivé de l'application *Wi-5 SmartApSelection*, gère de manière coordonnée la mobilité et l'équilibrage de charge. Des scans périodiques sont effectués afin de collecter des informations relatives à l'état du réseau et, au besoin, exécuter des handovers. La procédure de handover sera le plus souvent intercanal du fait que les AP fonctionneront sur des canaux différents. Les détails de la procédure sont fournis au niveau de la *Figure 49*.

La procédure commence par l'envoi, par le Contrôleur SDNRAN, d'une commande de scan des stations au niveau de chaque AP ①. Ces derniers renvoient le résultat de leurs scans au Contrôleur SDNRAN ②. Avec les informations reçues, le contrôleur exécutera un certain nombre de tâches ③ à savoir :

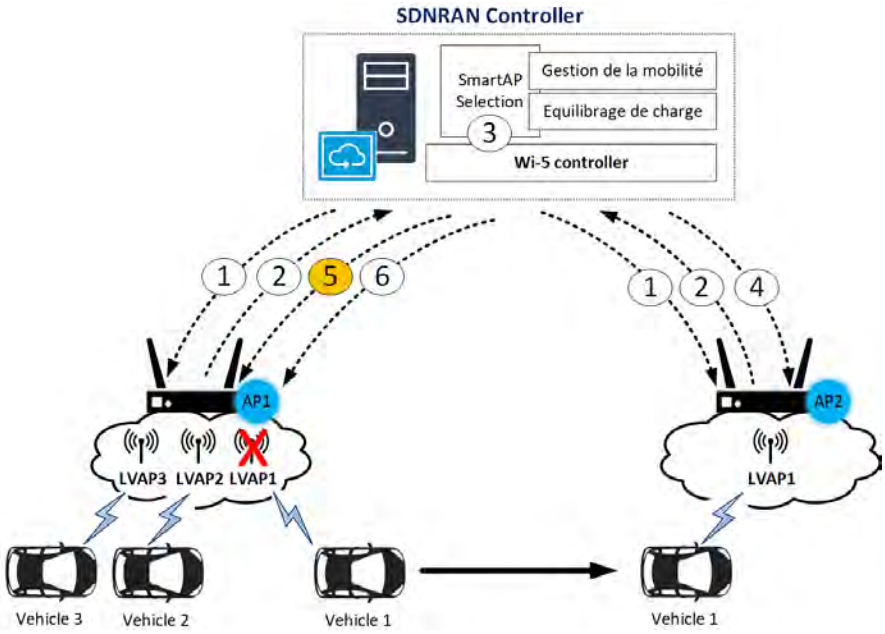


Figure 49 : Schéma de handover proactif

– *Génération de la matrice RSSI :* cette matrice contient le niveau de puissance du signal WiFi (RSSI) perçu par chaque AP.

$$RSSI = \begin{matrix} AP1 & AP2 & AP3 & \\ \begin{pmatrix} RSSI1.1 & RSSI1.2 & RSSI1.3 \\ RSSI2.1 & RSSI2.2 & RSSI2.3 \\ RSSI3.1 & RSSI3.2 & RSSI3.3 \end{pmatrix} & & & \begin{matrix} STA1 \\ STA2 \\ STA3 \end{matrix} \end{matrix} \quad (50)$$

Afin de tenir compte des données d'historiques, la valeur de chaque élément $RSSI_{i,j}$ de la matrice sera ajustée comme suit :

$$RSSI_{i,j} = \alpha * new\_RSSI_{i,j} + (1 - \alpha) * old\_RSSI_{i,j} \quad (51)$$

$0 \leq \alpha < 1$ est un paramètre de pondération des données d'historique.

– *Vérification du seuil RSSI :* Pour chaque station i, on vérifie si son RSSI avec l'AP auquel elle est liée, est supérieur à une valeur seuil. Si c'est le cas, la station i est maintenue sur l'AP courant. Sinon, on recherche au niveau de la ligne i de la matrice, l'APj fournissant un meilleur RSSI. La station i devra alors effectuer un handover vers ce nouvel APj.

– *Equilibrage de charge :* une fois les stations réparties sur les différents AP, on cherche au niveau de la matrice RSSI, l'ensemble des AP capables de fournir aux stations un RSSI supérieur à la valeur seuil fixée. On identifiera ensuite l'AP ($APmin$) où se sont connectées le moins de stations ($minSTAs$) et l'AP ($APmax$) où se sont connectées le plus de stations ($maxSTAs$). Enfin, on calcule le nombre moyen de stations connectées par AP ($avgSTAs$). Si $minSTAs < avgSTAs\ ET\ maxSTAs > avgSTAs$

alors la station présentant le meilleur RSSI par rapport à APmin et n'étant pas connectée à APmin fera un handover vers ce dernier.

A la suite de la procédure de décision de handover, le Contrôleur SDNRAN envoie des commandes de création de LVAP aux AP où les stations vont migrer ④. Ensuite, le contrôleur envoie des commandes commande *send CSA* aux APs d'origines des stations devant migrer ⑤ puis enfin des commandes de suppression de LVAP à ces APs ⑥.

A la fin de la procédure de handover, la station cliente « pense » avoir simplement effectuée un basculement de canal. S'il n'y pas de changement de canal, alors la station cliente n'a pas « conscience » de sa migration vers un autre AP.

5.3.2. Mobilité gérée par l'hyperviseur de passerelles/interfaces sans fil

Ce schéma de handover est une alternative ; il est proposé afin de basculer vers un réseau LTE lorsque le véhicule est sur le point de perdre le signal WiFi. En effet, le véhicule ne pouvant bénéficier d'une connectivité WiFi tout au long de son parcours devra chercher une connexion alternative. Ce type de handover est appelé handover vertical ou intertechnologie, il est exclusivement géré par le terminal (véhicule). Etant de niveau L2 et L3, par conséquent induit une reconfiguration IP de la station. La *Figure 50* détaille la procédure de handover intertechnologie.

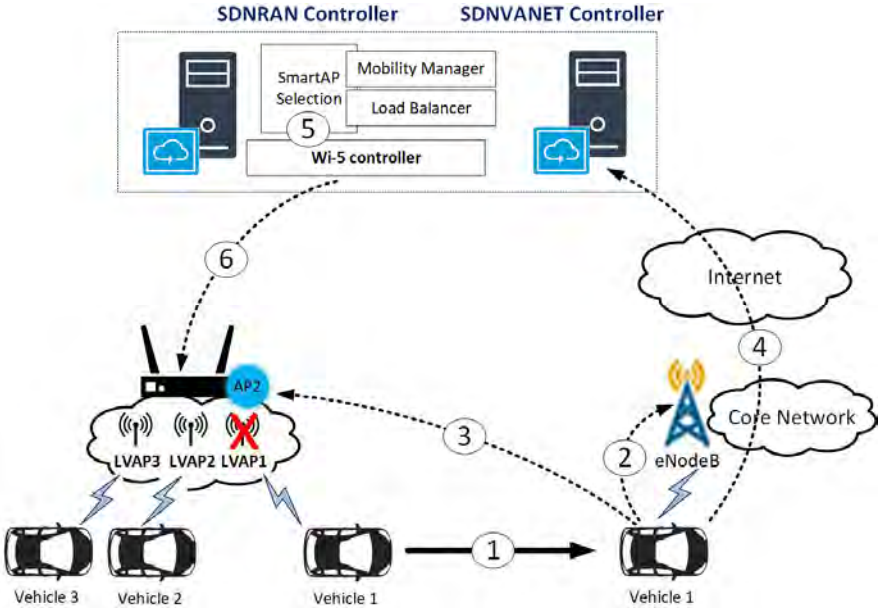


Figure 50 : Schéma de handover intertechnologie

Lorsque le véhicule s'éloigne de son AP courant (ici AP2) ①, l'hyperviseur remarque la dégradation progressive du signal WiFi. Ainsi, si la puissance du signal atteint une certaine valeur seuil critique (nettement inférieure à celle utilisée par le Contrôleur SDNRAN), alors l'hyperviseur se connecte sur un réseau LTE ② tout en se déconnectant du réseau WiFi via l'envoi d'une trame de *disassociation* à l'AP

où il s'est lié ③. Pour assurer son accessibilité, l'hyperviseur devra informer le Contrôleur SDNVANET de ce nouveau changement par l'envoi du statut du véhicule ④. En parallèle, lorsque le Contrôleur SDNRAN remarque une inactivité de la station (interface WiFi du véhicule) durant un intervalle de temps donnée ⑤, il envoie une commande de suppression de LVAP vers l'AP concerné ⑥.

A la fin de la procédure de handover, la station cliente est « consciente » du changement de la configuration réseau, car étant elle-même l'initiatrice du handover.

5.3.3. Evaluation des performances des schémas de mobilité

En se référant à la stratégie d'association flux/interface définie plus haut, on remarque que les paquets transitant via le backbone SDN sont essentiellement constitués du plan de contrôle échangé entre le contrôleur SDNVANET et les switches SDNVANET incorporés dans les véhicules. L'évaluation des performances portera essentiellement sur la latence de handover et sur le délai de la transmission du plan de contrôle.

5.3.3.1. Détails des expérimentations

Un ordinateur monocarte Raspberry Pi 3 (Cortex-A53 x64 1.2 GHz, SRAM 1GB, WiFi 2.4GHz 802.11b/g/n, OS Raspbian Stretch Lite) est utilisé comme switch SDNVANET, voir *Figure 51*. Ce dernier est également doté d'un modem 4G LTE USB Huawei E8372. Ce modem apparait comme une interface Ethernet ; ce qui va facilite grandement sa gestion. En effet, une fois le modem connecté au réseau de l'opérateur mobile, pour basculer du WiFi au LTE, et vice versa, il suffit de simplement activer ou désactiver l'interface Ethernet du modem. Cette procédure est plus efficace par rapport à celle consistant à connecter ou déconnecter le modem du réseau 4G (forte latence, importante signalisation...).

Le contrôleur de mobilité au niveau du switch SDNVANET est implémenté à l'aide de script Shell Linux. Les outils *nmcli* et *wpa_supplicant* sont utilisés pour le suivi des statuts réseau et le contrôle des interfaces réseau.

Pour la mobilité gérée par le contrôleur SDNRAN, les paramètres de l'application *MobilityManager* (schéma de mobilité réactif) sont entre autres la puissance seuil du signal WiFi ($SIGNAL\_THRESHOLD = -56dB$), le nombre de triggers ($NUMBER\_OF\_TRIGGERS = 5$), la durée de réinitialisation du trigger ($TIME\_RESET\_TRIGGER = 1s$) et la valeur de l'hystérésis ($HYSTERESIS\_THRESHOLD = 15s$). Ainsi, la procédure de handover est déclenchée lorsque l'AP reçoit, pendant 1s, au moins 5 paquets d'une station (véhicule) avec un seuil de puissance inférieure à $-56dB$ et qu'aucun handover n'ai eu lieu durant les 15s précédentes. Ce paramétrage permet d'éviter l'exécution répétitive de handovers, « effet ping pong », sur une courte période due à une perturbation temporaire.

Figure 51 : Vue de l'ordinateur de bord

Pour l'application *SmartApSelection* (schéma de mobilité proactif), également gérée par le contrôleur SDNRAN, les paramètres sont entre autres la puissance seuil du signal WiFi ($SIGNAL\_THRESHOLD = -56dB$), la valeur de l'hystérésis ($HYSTERESIS\_THRESHOLD = 4s$), l'intervalle de scan des stations ($SCANNING\_INTERVAL = 200ms$) et le facteur de pondération alpha ($\alpha = 0{,}2$).

Au cours du déplacement du véhicule, ce switch SDNVANET échange à intervalle régulier (3s) avec le contrôleur SDNVANET des messages OpenFlow echo (ECHO REQUEST et ECHO RESPONSE). Ces messages sont utilisés, entre autres, pour tester l'accessibilité (vivacité) du contrôleur SDNVANET et déterminer la latence de communication. Cette latence est définie comme étant la différence de temps entre l'instant où le switch envoie un paquet ECHO REQUEST et l'instant où il reçoit le paquet ECHO RESPONSE.

5.3.3.2. Performance des schémas de mobilité gérés par le SDNRAN

Le *Tableau 13* montre les résultats relatifs au temps de traitement, par le contrôleur SDNRAN (Wi-5), de la procédure de handover pour l'ensemble des schémas de mobilité proposés. Comparé aux schémas de mobilité réactifs, on note que le schéma de mobilité proactif nécessite moins de temps pour le traitement de la procédure de handover. En effet, le traitement d'un handover proactif intercanal prend 0,5231s tandis que les schémas de handover réactifs intra et intercanal nécessitent respectivement 1,0097s et 1,0077s. Cela s'explique du fait que, lors du traitement du handover, le schéma de mobilité proactif ne nécessite pas l'échange de messages scan (*Scan Request* et *Scan Response*) entre le contrôleur Wi-5 et les AP voisins. A la place, les résultats de scan sont envoyés de manière proactive (tous les 200ms) vers le contrôleur. En outre, on note que le nombre de handovers traités

par le schéma de mobilité proactif (4 handovers) est supérieur à celui traités par les schémas de mobilité réactifs (2 handovers). Cette différence se justifie par le fait que la valeur de l'hystérésis est réduite à 4s pour le schéma de mobilité proactif alors que pour les schémas de mobilité réactifs, elle est de 15s. En outre, certains handovers peuvent être déclenchés pour les besoins de l'équilibrage de charge.

Tableau 13 : Durée de traitement de la procédure de handover

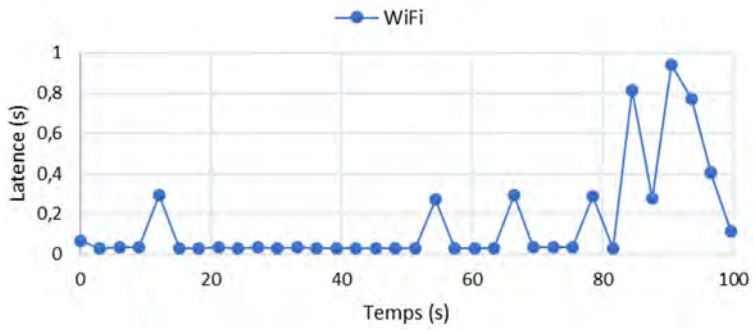
	Handover réactif intracanal	Handover réactif intercanal	Handover proactif intercanal
Nombre de handovers traités	2	2	4
Temps de traitement moyen (s)	1,0097	1,0077	0,5231

Le *Tableau 14* présente les résultats relatifs à la latence de handover notée sur l'interface WiFi du switch SDNVANET. On remarque une latence nulle pour le handover réactif intracanal, car cette procédure est totalement transparente au niveau de l'OBU (station cliente) ; et par conséquent aucune reconfiguration réseau n'est effectuée. Par contre pour les handovers réactifs et proactifs intercanal, on note respectivement une latence de 0,0794s et 0,0851s. Cette latence est due au changement de canal effectué au niveau de l'interface WiFi.

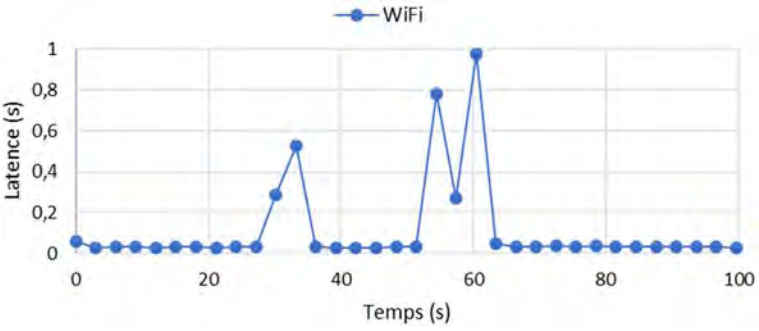
Tableau 14 : Latence de handover niveau L2 sur l'interface WiFi

	Handover réactif intracanal	Handover réactif intercanal	Handover proactif intercanal
Latence (s)	0	0,0794	0.0851

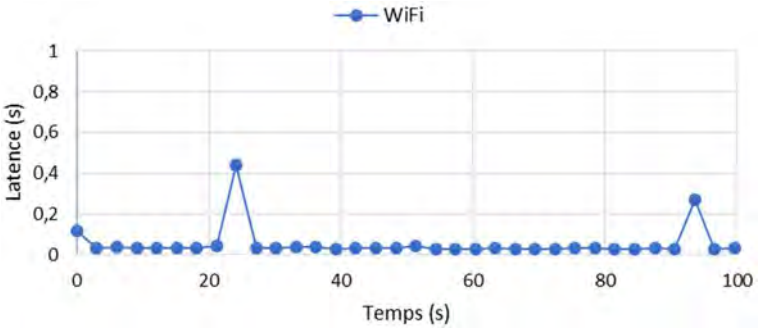
Enfin, la *Figure 52* détaille l'évolution de la latence relative aux échanges de messages echo OpenFlow entre le switch SDNVANET et le contrôleur SDNVANET. On note une latence moyenne de 0,155s pour le schéma de handover réactif intra-canal, *Figure 52.a*, contre une latence moyenne de 0,112s et 0,052s pour respectivement les schémas de handover réactif intercanal, *Figure 52.b*, et proactif intercanal, *Figure 52.c*. Ces écarts de latence sont dus essentiellement à la différence du nombre d'apparitions de pics de latence au niveau des différents schémas de handover. La dégradation du signal, causé par l'éloignement du véhicule, pourrait expliquer l'apparition de ces pics. En outre, on retiendra que la communication entre le switch SDNVANET et le contrôleur SDNVANET est beaucoup plus stable lorsque le schéma de handover proactif intercanal est utilisé. Cela est dû principalement au nombre de handover, voir *Tableau 13*, exécuté par ce schéma. En effet, le handover proactif permet au switch SDNVANET de basculler plus rapidement vers un autre AP lorsqu'une dégradation du signal WiFi est remarquée.



a) Handover réactif intracanal



b) Handover réactif intercanal



c) Handover proactif intercanal

Figure 52 : Latence messages echo OpenFlow – mobilité gérée par le SDNRAN

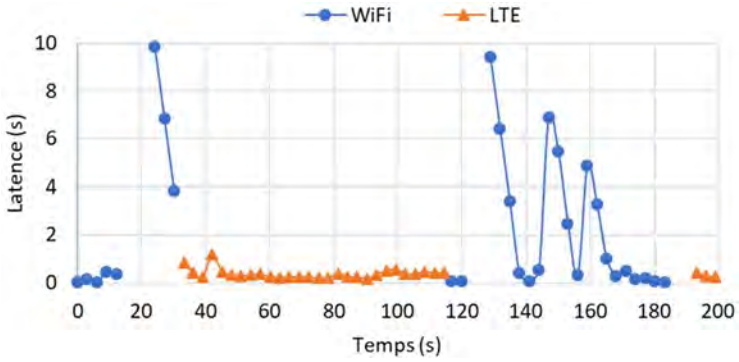
En somme, en se basant sur les résultats obtenus, on peut conclure que l'ensemble des schémas de mobilité gérés par le SDNRAN peuvent être utilisés pour les besoins de communication des véhicules. En effet, les résultats du *Tableau 14* montrent que la latence induit lors du handover est très faible pour être ressentis par le switch SDNVANET et le contrôleur SDNVANET. En outre, la *Figure 52* montre que dans l'ensemble, la latence de communication entre ce switch et le contrôleur SDNVANET est relativement faible, et est stable. Par contre, la durée de traitement de la procédure du handover, *Tableau 13*, peut constituer une entrave dans certains scénarios de communication véhiculaire. En effet, dans les scénarios où le véhicule roule à très grande vitesse (autoroute, zone à faible affluence…), il se peut que le véhicule perde sa connectivité WiFi avant la fin du traitement de la

procédure de handover. Néanmoins, avec le schéma de mobilité proactif, ce délai de traitement est réduit et peut, par conséquent, garantir le maintien de la connectivité WiFi.

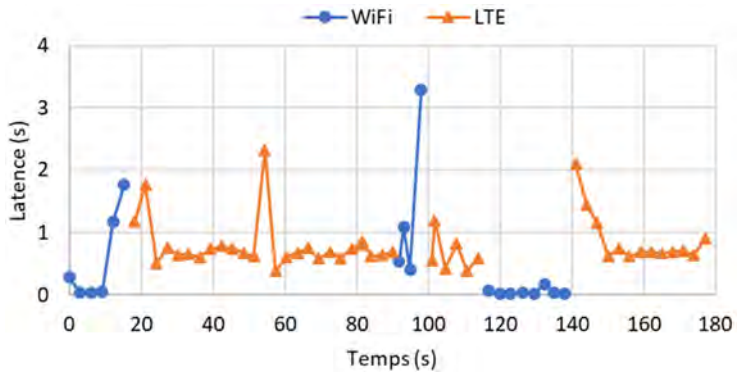
Nous retiendrons dans la suite le schéma de mobilité proactif, car étant le plus performant. Cependant, ce schéma nécessite une remontée périodique (tous les 200ms) des résultats de scan des AP ; ce qui peut contribue à augmenter la charge du plan de contrôle RAN. Il est alors nécessaire de bien optimiser les ressources du backbone SDN pour tenir compte de ces charges réseau supplémentaires.

5.3.3.3. Performance du schéma de mobilité géré par l'hyperviseur

Dans cette partie, il s'agit essentiellement d'observer le délai de communication entre le switch SDNVANET et le contrôleur SDNVANET dans le cadre d'un handover intertechnologie (WiFi vers LTE, et vice-versa). La *Figure 53* montre les résultats des tests de performance.



a) Puissance seuil du signal WiFi $= -80dB$



b) Puissance seuil du signal WiFi $= -70dB$

Figure 53 : Latence messages echo OpenFlow – mobilité gérée par l'hyperviseur

Pour le premier test, *Figure 53.a*, la valeur seuil de la puissance du signal WiFi déclenchant un handover est fixée à $-80dB$. Notons cependant que cette valeur seuil est considérée comme à une connexion WiFi instable, ce qui explique l'apparition de certains pics (9.84s, 9.40s, 6.90s et 4.86s) et des pertes de paquets importantes (10.44%). Lorsque le contrôleur de mobilité remarque que la puissance

du signal WiFi a atteint la valeur de $-80dB$, alors il bascule vers l'interface LTE. Avec la connectivité 4G, on note une stabilité de la latence de communication entre le switch SDNVANET et le contrôleur SDNVANET, avec certes une légère plus grande latence. L'interface LTE étant une considérée comme secondaire, dès que le contrôleur de mobilité détecte un signal WiFi dont le RSSI est inférieur à $-80dB$, alors il bascule vers la connectivité WiFi.

Pour le second test, *Figure 53.b*, la valeur seuil est fixée à $-70dB$. Le premier constat est l'amélioration de la qualité de communication entre le switch SDNVANET et le contrôleur SDNVANET. En effet, on note l'apparition de moins de pics (1.76s et 3.28s), qui d'ailleurs sont de valeurs plus faibles comparées à celle à $-80dB$. Cependant avec une telle valeur seuil de puissance, on note une diminution de l'utilisation de l'interface WiFi lors des échanges de messages OpenFlow echo. En effet, réduire la valeur seuil de puissance du signal WiFi admissible, revient à réduire virtuellement sa zone de couverture.

En résumé, les résultats de performance montrent que la transmission du plan de contrôle peut se faire via les interfaces WiFi et LTE du switch SDNVANET. Lorsque le seuil de puissance du signal WiFi devant déclencher un handover est faible niveau, le switch SDNVANET peut utiliser plus longtemps la connectivité WiFi (43,32% à $-80dB$), mais au risque d'apparition de pics de latence et de pertes de paquets. Par contre, lorsqu'on choisit un niveau seuil du signal WiFi plus élevé, on notera une meilleure qualité de la communication entre le switch SDNVANET et le contrôleur SDNVANET, cependant cela réduit l'utilisation de l'interface WiFi (27,42% à $-70dB$). La connectivité LTE présente certes de bonnes performances, mais demeure coûteuse, par conséquent elle ne devra pas être privilégiée. Lorsqu'une bonne couverture WiFi est assurée, en densifiant la présente de points d'accès WiFi tout au long des routes, on peut parvenir à utiliser convenablement la connectivité WiFi comme médium d'échange du plan de contrôle SDNVANET.

5.4. Schémas de routage du SDNVANET

Le routage dans les VANET est une opération complexe car devant face à plusieurs contraintes liées à la forte mobilité des véhicules, à leurs topologies hautement dynamiques et aux fréquentes ruptures de connexion. Le modèle S^2aaS est basé sur le IVSN, il hérite donc d'un modèle de communication basé sur la dissémination et le traitement collaboratif de données. Cette collaboration se fait via un groupe de véhicules ayant comme point commun une même zone d'intérêt.

Dans ce qui suit, nous ne considérerons que le géocast comme modèle de communication V2V. La communication vers la RSU se fera en mode unicast en suivant les principes de la communication V2I.

5.4.1. Geocast (V2V)

Le geocast consiste à envoyer des paquets de données d'une source vers tous les véhicules présents sur une zone géographique appelée Zone Of Relevance (ZOR) [42]. Notons que le Contrôleur SDNVANET est responsable de la délimitation de chaque ZOR. L'établissement d'un schéma de routage au sein de chaque ZOR se fera en plusieurs étapes :

- *Etape 1 : collecte du statut des véhicules*

Tous les véhicules (nœuds fog mobiles) devront envoyer à intervalle de temps régulier ΔT leur statut au Contrôleur SDNVANET. Cet intervalle est déterminé en fonction de la densité de la circulation. L'envoi du statut se fera via l'interface WiFi où LTE comme indiqué par la stratégie d'association flux/interface décrite plus haut. On définira le $statut_i$ à l'instant t d'un véhicule V_i comme un ensemble contenant les éléments suivants :

$$statut_i = \{p_i, v_i, dir_i, dst_i, PW_i\} \tag{52}$$

Où p_i est la position, v_i la vitesse, dir_i est la direction, dst_i est la destination (paramètre facultatif), PW_i est la puissance d'émission de l'interface 802.11p.

- *Etape 2 : génération de la topologie*

Après réception du statut des véhicules, le Contrôleur SDNVANET est en mesure de déterminer la topologie réseau de chaque ZOR à l'instant t. La détermination de la topologie à l'instant $t + \delta t$, avec $\delta t < \Delta T$, se fera en employant un algorithme de prédiction de trajectoire. Dans notre cas, l'algorithme utilisé est dérivé du modèle de la voiture suiveuse (car following model) proposé par Krauss [150]. C'est un modèle de mobilité microscopique qui décrit la dynamique individuelle de chaque véhicule en fonction des positions et des vitesses des véhicules voisins. Le modèle de Krauss est utilisé dans beaucoup d'études et aussi par des simulateurs tel que SUMO (Simulation of Urban MObility) [165]. Le modèle de Krauss peut être formulé comme suit :

$$\begin{aligned} v_{safe}(t) &= v_l(t) + \frac{g(t) - g_{des}(t)}{\tau_b + \tau} \\ v_{des}(t) &= min\left[v_{max}, v(t) + a(v)\Delta t, v_{safe}(t)\right] \\ v(t + \delta t) &= max[0, v_{des}(t) - \eta] \\ x(t + \delta t) &= x(t) + v\delta t \end{aligned} \tag{53}$$

Où v_{safe} est la vitesse maximale de sécurité, v_l et v_f sont respectivement la vitesse du véhicule leader et du véhicule suiveur. La grandeur $g = x_l - x_f - l$ est le gap entre le véhicule leader de position x_l et le véhicule suiveur de position x_f, l étant la longueur du véhicule. Le gap désiré est déterminé par $g_{des} = \tau v_l$ où τ est le temps de réaction du conducteur. $\tau_b = \bar{v}/b$ est l'échelle de temps où $\bar{v} = (v_l +$

$v_f)/2$ exprime la moyenne des vitesses entre les véhicules leader et suiveur, b est la décélération maximale. v_{des} est la vitesse désirée et elle est définies comme le minimum entre la vitesse maximale v_{max}, la vitesse maximale de sécurité v_{safe} et la vitesse $v(t) + a(v)\Delta t$ produite par l'accélération maximale a. Une perturbation aléatoire $\eta > 0$ est introduite pour permettre une déviation de la vitesse désirée calculée.

Afin de tenir compte des feux de circulation, nous avons introduit dans ce modèle, une nouvelle vitesse nommée v_{safeTL} où TL signifie Traffic Light. Ainsi, lorsque le feu passe de l'orange ou au rouge, v_{safeTL} est défini comme suit :

$$v_{safeTL}(t) = \frac{g(t) - g_{des}(t)}{\tau_b + \tau} \tag{54}$$

La différence entre v_{safeTL} et v_{safe} est que le feu de circulation, lorsqu'il passe à l'orange ou au rouge, est imaginé comme un véhicule leader en mode stationnaire, c'est-à-dire $v_l = 0$. En somme, avec ce modèle, en connaissant la position $x(t)$ du véhicule à l'instant t, on peut prédire sa position à l'instant $t + \delta t$. Par conséquent, la topologie des véhicules peut être prédite à chaque instant.

- *Etape 3 : établissement des routes*

L'établissement des routes est géré par le Contrôleur SDNVANET, et s'effectue à la base de la connaissance de la topologie du ZOR prédite à chaque intervalle de temps δt. Les variables utilisées pour l'algorithme de géocast sont regroupées sur le *Tableau 15*.

Tableau 15 : Notation utilisée pour l'algorithme de géocast

V	L'ensemble des véhicules appartenant au ZOR
R_i	La portée de l'interface 802.11p du véhicule V_i
D_{ij}	La distance euclidienne entre le véhicule V_i et un autre véhicule $V_j \in V$
$V_{i,direct}$	L'ensemble des véhicules du ZOR pouvant directement communiquer avec le véhicule V_i, c'est-à-dire avec un seul saut.
$V_{i,\phi}$	L'ensemble des véhicules du ZOR n'ayant pas encore reçus le message de geocast de V_i
$V_{i,relay}$	L'ensemble des véhicules du ZOR pouvant être des relais potentiels pour $V_{i,\phi}$
$V_{i,relay}^{j}$	L'ensemble des véhicules choisis pour relayer le message du véhicule V_i au véhicule $V_j \in V_{i,\phi}$. On a $V_{i,relay}^{j} \subset V_{i,relay}$ et dim $V_{i,relay}^{j} = 2$
V_{direct}^{opf}	L'ensemble des véhicules accessibles via une transmission directe par V_i lors d'un geocast. V_{direct}^{opf} est utilisé pour la génération de la table des flux openflow.
V_{relay}^{opf}	L'ensemble des véhicules qui joueront effectivement le rôle de relais lors d'un geocast de V_i. V_{relay}^{opf} est utilisé pour la génération de la table des flux openflow.

L'*Algorithme 3* décrit le mécanisme d'établissement des routes. La première phase consiste à détecter l'ensemble des véhicules $V_{i,direct}$ accessibles via un seul saut. Le rapport $\frac{D_{ij}}{R_i}$ exprime le niveau de rapprochement des véhicules V_i et V_j, et permet d'estimer la probabilité de transmission avec succès d'un message entre ces deux véhicules. Ainsi plus ce rapport est petit, plus grand sera la chance pour qu'il ait succès de transmission. Les véhicules appartenant à $V_{i,direct}$ formeront les premiers éléments de l'ensemble $V_{i,relay}$. Pour chaque véhicule $V_j \in V_{i,\phi}$, seuls deux véhicules appartenant $V_{i,relay}$ et présentant le ratio $\frac{D_{ij}}{R_i}$ le plus faible seront choisis pour former l'ensemble $V_{i,relay}^{j}$. De ce fait, en cas d'échec de transmission de l'un des relais, V_j pourra toujours espérer sur le second pour recevoir le paquet. A chaque relayage de message (itération), l'ensemble des véhicules $V_j \in V_{i,\phi}$ qui recevront le message constitueront le nouvel ensemble $V_{i,relay}$. Le relayage s'arrête lorsque $V_{i,\phi} = \{\phi\}$ ou $V_{i,relay} = \{\phi\}$.

A la fin de l'exécution de l'*Algorithme 3*, V_{direct}^{opf} contiendra pour chaque véhicule $V_i \in V$, l'ensemble des véhicules $V_{i,direct}$ avec lesquels V_i est un voisin direct. Ainsi, lorsqu'un véhicule V_i souhaite diffuser un message de geocast, il enverra ce message directement aux véhicules de l'ensemble $V_{i,direct}$. V_{relay}^{opf} contient pour chaque véhicule V_i, l'ensemble des véhicules V_j qui recevront le message de geocast de la part des véhicules relais $V_{i,relay}^{j}$.

Algorithme 3 sélection des chemins de routage

Require : V

Output : $V_{direct}^{opf}, V_{relay}^{opf}$
for each $V_i \in V$ **do**
  select $V_{i,direct}$
  $V_{i,relay} \leftarrow V_{i,direct}$
  $V_{i,\phi} \leftarrow V_{i,\phi} \setminus V_{i,direct}$
  $V_{direct}^{opf} \leftarrow V_{direct}^{opf} \cup \{(V_i, V_{i,direct})\}$
  while $V_{i,\phi} != \{\phi\}$ **and** $V_{i,relay} != \{\phi\}$ **do**
    for each $V_j \in V_{i,\phi}$ **do**
      select $V_{i,relay}^{j}$
      $V_{i,relay}^{next} \leftarrow V_{i,relay}^{next} \cup V_j$
      $V_{relay}^{opf} \leftarrow V_{relay}^{opf} \cup \{(V_j, V_{i,relay}^{j})\}$
    end for
    $V_{i,relay} \leftarrow V_{i,relay}^{next}$
    $V_{i,\phi} \leftarrow V_{i,\phi} \setminus V_{i,relay}^{next}$
  end while
end for
return $V_{direct}^{opf}, V_{relay}^{opf}$

- *Etape 4 : génération de la table des flux Openflow*

La génération de la table des flux se fera en se basant sur les ensembles V_{direct}^{opf} et V_{relay}^{opf}. A chaque intervalle ΔT, une table de flux sera générée pour chaque véhicule $V_i \in V$. Plus précisément, ΔT sera subdivisée en petit intervalle δt où l'on exécutera l'*Algorithme 3*. La table de flux durant l'intervalle ΔT sera la juxtaposition de l'ensemble des routes calculées et valides à chaque instant $t + \delta t < \Delta T$ où $t \in [0, \Delta T]$. Une fois l'ensemble des tables de flux générées, le Contrôleur SDNVANET se chargera d'envoyer à chaque véhicule $V_i \in V$, sa table de flux correspondante.

5.4.2. Routage unicast (V2I)

La communication V2I se fera en mode unicast, et sera également basée sur le routage géographique. Le Contrôleur SDNVANET construit une route pour chaque véhicule $V_i \in V$ en utilisant un algorithme de relayage glouton qui consiste à sélectionner, pour chaque saut, le voisin le plus proche de la destination. Il ne sera pas donc nécessaire de refaire des calculs de routes, car ces dernières peuvent être extraites à partir de V_{direct}^{opf} et V_{relay}^{opf}. En effet, l'*Algorithme 3* utilise le principe du relayage glouton lors de la sélection des véhicules relais $V_{i,relay}^{j}$. La génération et l'envoi des tables OpenFlow, pour chaque véhicule $V_i \in V$, se fera comme décrit dans l'étape 4 pour la communication V2V.

5.4.3. Evaluation de performances des schémas de routage V2V et V2I

5.4.3.1. Description de la simulation

L'évaluation de performance se fera via un simulateur codé en Python qui implémente l'extension du modèle microscopique de Krauss proposé, du protocole de routage SDNVANET proposé, ainsi que les protocoles OLSR (Optimized Link State Routing Protocol) [166] et GPSR (Greedy Perimeter Stateless Routing) [167]. La particularité de ce simulateur est que les véhicules circulent sur des routes à deux voies avec l'intégration de feux de circulation, voir *Figure 54*

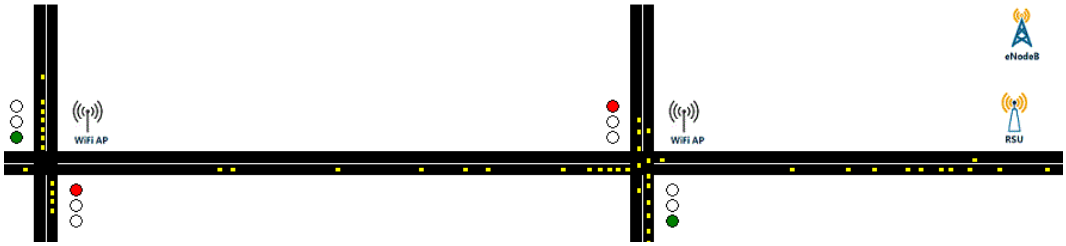


Figure 54 : Interface graphique tronquée du simulateur

Chaque véhicule dissémine, via le géocast, les mesures de ses capteurs. De même, la RSU collecte les mesures des capteurs automobile en utilisant la méthode pull en mode unicast. Le *Tableau 16* regroupe l'ensemble des paramètres de simulation.

Tableau 16 : Paramètres de simulation des schémas de routage SDN

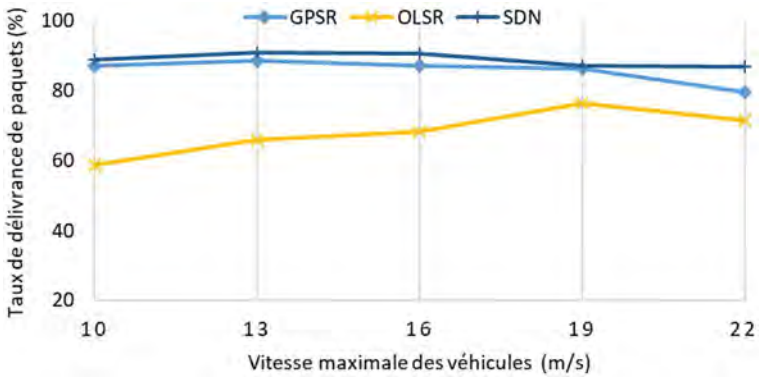
Paramètres du modèle de Kraus	$\tau = 1s,\ l = 7.5m, a = 0.8m\ s^{-2},$ $b = 4.5m\ s^{-2}$
Coordonnées des intersections (x, y)	(500m, 250m) et (1000m, 250m)
Flux de véhicules sur la route principale	voie1 = 1/6veh/s, voie2 = 1/25veh/s
Flux de véhicules sur la première rue perpendiculaire	voie1 = 1/30veh/s, voie2 = 1/23veh/s
Flux de véhicules sur la seconde rue perpendiculaire	voie1 = 1/23veh/s, voie2 = 1/35veh/s
Porté des interfaces 802.11p, 802.11n, LTE	500m, 200m, 5000m
Position RSU (x,y)	(1500m, 250m)
Délimitation du ZOR (x1, y1, x2, y2)	(100m, 150m, 2000m, 350m)
Intervalle d'envoi du statut	17s
Intervalle de publication des mesures	5s
Durée de simulation	100s

5.4.3.2. Taux de délivrance de paquets

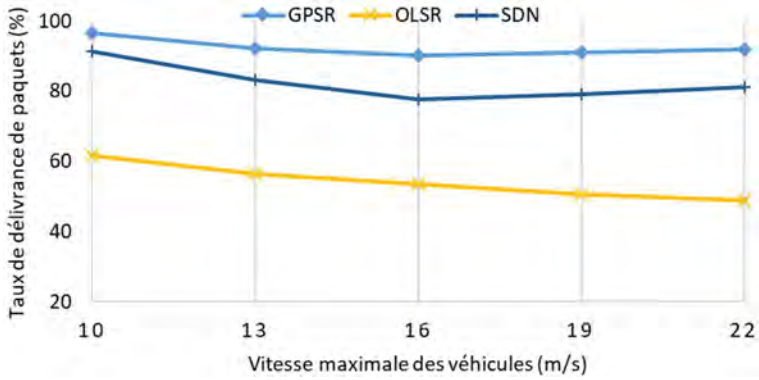
En mode unicast, *Figure 55.a*, le schéma de routage SDNVANET proposé présente les meilleures performances comparées au schéma OLSR et GPSR. La capacité de prédiction de la position des véhicules, par le Contrôleur SDN VANET, en est la principale cause. En effet, cette capacité de prédiction fait que le schéma de routage SDNVANET soit moins impactés par les changements hautement dynamiques de la topologie des VANET. Ce qui n'est pas le cas pour les protocoles de routages classiques tels que OLSR et GPSR. Par conséquent, ces deux protocoles sont beaucoup plus vulnérables aux changement brusque de topologie.

En mode géocast ou broadcast *Figure 55.b* le schéma de routage GPSR/inondation présente de meilleures performances en termes de taux de délivrance de paquets. Néanmoins, le schéma de routage SDNVANET proposé présente des performances assez proches de celui du GSPR. Ce dernier, étant un protocole de routage unicast, utilise la méthode d'inondation pour le broadcast. Ce qui justifie ce gain de performance. Cependant, l'inconvénient de la méthode d'inondation et qu'elle génère une très forte charge réseau, ce qui peut créer des congestions réseau, et réduire, par conséquent le taux de délivrance de paquets.

Nous remarquons également que le taux de délivrance de paquets décroit avec la vitesse des véhicules. Pour le cas du schéma SDNVANET, la cause vient du fait que les écarts entre les positions prédites et celles effectives des véhicules croient avec la vitesse. Les feux de circulation ont aussi un impact sur le taux de délivrance de paquet dans le sens où ils perturbent le flux de trafic routier, notamment en en fracturant de façon brutale la topologie des véhicules.



a) Mode unicast



b) Mode géocast (broadcast)

Figure 55 : Comparaison des taux de délivrance de paquets

Dans ce qui suit, nous mesurerons la charge réseau pour mieux évaluer les performances du schéma de routage SDNVANET.

5.4.3.3. Charge réseau due au routage

En se basant sur la *Figure 56*, on constate que le schéma SDNVANET proposé présente une charge réseau de routage très faible (entre 1523 et 1900 paquets) comparée aux schémas de routage classique OLSR et GPSR. En effet, la charge réseau du schéma SDNVANET vient principalement du relayage des paquets qu'effectuent les véhicules dans le ZOR. Or, cela ne survient que lorsqu'il y a des messages échanger. Les autres sources causant des charges réseau ont été éliminées grâce à la stratégie d'association flux/interfaces proposée. Ainsi, aucun paquet de signalisation ne transite via l'interface 802.11p, de même, qu'aucun paquet balise (message hello) n'est généré.

Le protocole OLSR présente également une charge réseau réduite (entre 41712 et 32388 paquets) comparée à celle du GPSR. En effet, le protocole OLSR est particulièrement optimisé pour réduire la charge réseau de routage grâce à une sélection minutieuse des nœuds relais.

La principale cause la charge réseau de routage très élevé élevée (entre 212771 et 260622 paquets) du protocole GPSR vient du fait de l'usage de la méthode d'inondation pour le mode broadcast. La conséquence d'une telle charge réseau est un épuisement rapide de la bande passante ; et également une augmentation du délai pour l'accès au canal et des pertes de paquets.

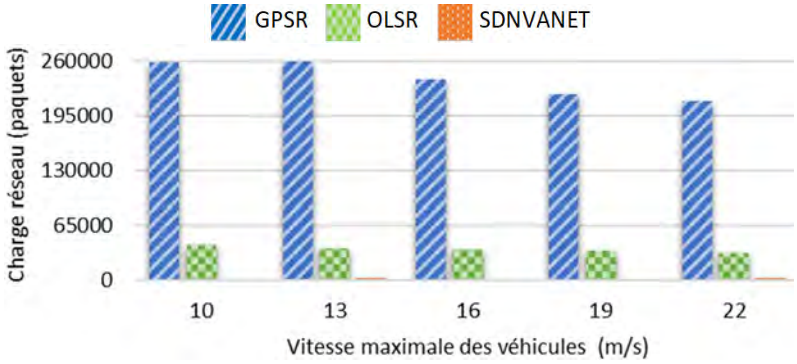


Figure 56 : Comparaison de la charge réseau des protocoles de routage

Afin d'analyser le débit et le délai du protocole de routage SDNVANET proposé, nous comptons implémenter dans nos travaux futurs le protocole d'accès au canal EDCA. Ce protocole permettra également d'étudier l'impact de la charge réseau sur le taux de délivrance de paquets.

5.5. Conclusion

Ce dernier chapitre a été consacré au prototypage d'un réseau SDN véhiculaire. Le prototypage consiste à mettre en place en ensemble de contrôleurs SDN devant orchestrer les différentes entités réseau à savoir les switches du backbone, les points d'accès du RAN et les modules de communication intégrés dans les ordinateurs de bord des véhicules.

Les tests de performance nous ont permis d'observer le comportement global du réseau SDN véhiculaire. Nous avons étudié les performances (débit et RTT) du backbone SDN en fonction de sa sollicitation. Au niveau du RAN, plusieurs schémas de mobilité ont été testés. Ensuite, nous avons observé comment ces schémas de mobilité impactent sur la qualité de communication entre le contrôleur SDN et les modules de communication intégrés dans les véhicules. Enfin, au niveau du VANET, des schémas SDN de routage géocast et unicast ont été proposés et évalués. Les métriques observées lors des tests de performance (via une simulation) sont le taux de délivrance de paquets et la charge réseau engendrée.

D'après les différentes observations, il en ressort que le SDN demeure une bonne solution pour répondre aux besoins de communication des LM et du S^2aaS.

Conclusion générale

Dans les deux premiers chapitres de cette thèse, nous avons présenté le concept de Laboratoire Mobile de Nouvelle Génération ou LM. Nous avons notamment montré comment les capteurs automobiles pouvaient être exploités dans le but de répondre à plusieurs besoins d'observation et d'étude de phénomènes physiques exprimés par les acteurs de l'industrie, gouvernement, médecine, organisme de recherche… Ensuite, l'aspect communication véhiculaire a été abordé en montrant notamment l'évolution du véhicule partant d'un simple nœud dans les VANET, jusqu'à une plateforme intelligente multi-communicante dans l'IoV. Un état d'art des schémas de mobilité et de gestion du flux pour la communication véhiculaire est également proposé. Il s'agit des protocoles de mobilité basés sur WAVE, IPv6, HIP et SDN.

Les trois derniers chapitres ont constitué nos différentes contributions dans cette thèse. Ainsi, le protocole HIP a été utilisé pour concevoir une architecture de communication sécurisée de même que pour définir des schémas de mobilité. Une étude de performance de HIP (latence Base Exchange, latence handover, débit) a été faite à base d'un ordinateur monocarte de type Raspberry Pi. Les résultats ont montré que les ordinateurs monocartes supportent bien le protocole HIP à condition d'ajuster certains paramètres en lien avec les opérations cryptographiques (difficulté du puzzle, groupe Diffie-Hellman). Néanmoins, l'implémentation d'un hyperviseur à base de HIP pose plusieurs problèmes. En effet, cela nécessite d'une part, une modification de l'architecture Internet actuelle, et d'autre part, les multiples associations HIP parallèles engendrent une surcharge CPU importante au niveau de l'ordinateur monocarte et des infrastructures réseaux (serveurs de rendezvous, pLRVS, S-RVS…). Cela constitue une limite pour l'évolutivité (scalabilité) du schéma HIP véhiculaire proposé. En outre, le modèle de communication de HIP ne couvre pas l'ensemble des besoins de communication des LM. Tout ceci nous a poussé à proposer des solutions beaucoup plus adaptées.

Un nouveau modèle de communication, nommée IVSN a été proposé dans un premier temps. Ce modèle repose sur quatre types de sensibilité (spatiale, groupe, données, contexte) permettant aux véhicules d'apprendre, de penser et de comprendre les systèmes cyberphysiques par eux-mêmes. Les avantages de l'IVSN ont été exploités dans la proposition d'un service de type S^2aaS qui d'inférer de nouvelles connaissances à partir des données de capteurs recueillies. Des techniques de découverte, de classement et de sélection de capteurs permettent à l'utilisateur, même sans expérience, de pouvoir exploiter au mieux les données issues des LM.

Un premier modèle S^2aaS a été proposé. L'évaluation de ce modèle montre qu'il est possible de réduire les pertes de paquets, mais avec comme contrainte une réduction de la zone de couverture des RSU. Des techniques d'optimisations réseaux ont permis d'améliorer le taux de délivrance de paquets. La limite principale de ce modèle réside sur les techniques d'optimisation de la sélection des capteurs des LM. En effet, la résolution d'un problème de programmation linéaire multiobectif intervient dans la procédure de sélection des capteurs. Or, cette méthode de résolution est complexe, par conséquent risque de générer une latence pour l'accès aux services S^2aaS. En outre, des limites au niveau fonctionnel ont également été recensées. Ainsi, un second modèle S^2aaS a été proposé afin de remédier ces problèmes en améliorant notamment le modèle fonctionnel du S^2aaS et les méthodes de sélection et de classement des capteurs. Un exemple d'application pratique du S^2aaS a été proposé. Il s'agit du *DBAQ-Routing* qui permet d'indiquer au conducteur l'itinéraire le moins pollué en termes de concentration des différents types de polluants. Enfin, un guide d'implémentation pratique à base d'Android Automotive montre que les prochaines générations de véhicules connectés pourront être transformés en LM.

Pour assurer une bonne communication des entités du modèle S^2aaS, un modèle réseau basé sur la technologie SDN a été proposé. Le SDN a été utilisé pour l'implémentation de l'hyperviseur de passerelle/interface sans fil au niveau des véhicules, de même que la gestion de l'ensemble des infrastructures réseaux utilisées. Ainsi, le routage dans les VANET est géré par un contrôleur SDNVANET qui s'appuie sur la prédiction des positions des véhicules pour générer la table des flux. L'évaluation des performances (taux de délivrance des paquets, charge réseau) du SDNVANET a été faite dans un environnement de simulation pour des raisons de contraintes logistiques. Les résultats ont montré que le routage SDNVANET est adapté aux besoins de communication des LM.

Le plan de contrôle constitue la partie critique de l'architecture SDN. Ainsi, un backbone SDN a été proposé pour le transport du plan de contrôle RAN et VANET. Des schémas de routage basé sur le nombre de sauts et sur la bande passante libre ont été proposés. Leurs évaluations via un prototype de backbone constitué de switches SDN de type Zodiac FX ont montré que le routage basé sur la bande passante libre est le mieux adapté pour faire face aux montés en charge. Au niveau du RAN, plusieurs schémas SDN de mobilité WiFi pour le véhicule ont été proposés en se basant sur l'architecture du projet Wi-5. De même, au niveau de l'ordinateur de bord du véhicule, un contrôleur de mobilité est chargé de prendre le relais lorsque le contrôleur SDNRAN n'est plus en mesure de gérer la connectivité des véhicules. L'évaluation des performances des schémas de mobilité a été faite via un prototype de RAN constitué de points d'accès WiFi TP-LINK AC1750 v2, d'ordinateurs monocarte Raspberry Pi et de modem LTE Huawei E8372. Les résultats des tests montrent que le WiFi et la LTE peuvent être utilisés pour la transmission du plan de contrôle.

En perspectives, nous comptons dans un premier temps évaluer les performances du schéma de routage SDNVANET au niveau MAC. Il s'agira d'implémenter le protocole EDCA (Enhanced Distributed Channel Access) au niveau de la pile WAVE/802.11p. L'intégration des contraintes liées à la couche MAC nous permettra de mieux évaluer les performances du schéma de routage VANET en termes de délais, de débit et le taux de délivrance de paquets.

Dans un second temps, l'utilisation du simulateur de trafic routier SUMO (Simulation of Urban MObility) permettrait de générer des modèles de trafic beaucoup plus réalistes grâce notamment à l'utilisation de cartes réelles via la plateforme *OpenStreetMap*. En outre, les applications de simulation des VANET peuvent interagir facilement avec SUMO via la librairie TraCL. Cela nous permettra, dans nos travaux futurs, de se focaliser beaucoup plus sur l'implémentation de modèles d'accès au canal, de sécurité...

En outre, l'une des raisons du non succès de l'architecture actuelle des VANET réside sur son coup de déploiement et son manque d'évolutivité. L'avènement des NVG (Next Generation Vehicular Networks) s'annonce ainsi très prometteuse pour le déploiement à large échelle d'infrastructures réseau adaptées pour la communication véhiculaire. Les NVG regroupent la technologie NR-V2X (New Radio V2X) pour le C-V2X (Cellular-V2X) proposée dans le cadre de la 4G LTE/5G NR, et la technologie IEEE 802.11bd proposée dans le cadre de la DSRC. Nous comptons dans nos futurs travaux, s'inspirer des NVG pour proposer de nouveaux protocoles et services pour les véhicules connectés. Il est possible de prototyper un réseau C-V2X grâce à la virtualisation de fonctions réseau via la SDR (Software Defined Radio).

Enfin, pour la gestion du plan de données des *targets* (switches, cartes réseaux, routeurs, interfaces de communication des véhicules...), les solutions SDN, notamment OpenFlow, la laissent à la discrétion des équipementiers. Avec le paradigme du plan de données programmable via le langage P4 (Programming Protocol-Independent Packet Processors), l'efficacité (n'implémenter que les protocoles nécessaires sur les *targets*), la *target*-indépendance (code exécutable sur différentes plateformes telles que ASIC, FPGA, NPU, eBPF, Open vSwitch), la visibilité (création de tag sur les paquets), la simplicité (création de nouveaux protocoles, traitement d'entêtes non prédéfinis) ... sont pleinement prises en compte. Ainsi, gérer le plan de contrôle via OpenFlow et le plan de données via P4Runtime permettrait de concevoir un réseau véhiculaire entièrement programmable. Le but étant d'améliorer entre autres les performances, la sécurité, la flexibilité de l'infrastructure réseau véhiculaire et des unités de communication automobiles.

REFERENCES

[1] W. J. Fleming, "New Automotive Sensors - A Review," *IEEE Sensors Journal*, vol. 8, no. 11, pp. 1900–1921, Nov. 2008, doi: 10.1109/JSEN.2008.2006452.

[2] S. Abdelhamid, H. S. Hassanein, and G. Takahara, "Vehicle as a Mobile Sensor," *Procedia Computer Science*, vol. 34, pp. 286–295, Jan. 2014, doi: 10.1016/j.procs.2014.07.025.

[3] *Bosch Automotive Electrics and Automotive Electronics: Systems and Components, Networking and Hybrid Drive*, 5th ed. Springer Vieweg, 2014.

[4] M. Petracca, P. Pagano, R. Pelliccia, M. Ghibaudi, C. Salvadori, and C. Nastasi, "On-Board Unit Hardware and Software Design for Vehicular Ad-Hoc Networks," *Roadside Networks for Vehicular Communications: Architectures, Applications, and Test Fields*, pp. 38–56, 2013, doi: 10.4018/978-1-4666-2223-4.ch002.

[5] P. Papadimitratos, A. D. L. Fortelle, K. Evenssen, R. Brignolo, and S. Cosenza, "Vehicular communication systems: Enabling technologies, applications, and future outlook on intelligent transportation," *IEEE Communications Magazine*, vol. 47, no. 11, pp. 84–95, Nov. 2009, doi: 10.1109/MCOM.2009.5307471.

[6] G. Karagiannis *et al.*, "Vehicular Networking: A Survey and Tutorial on Requirements, Architectures, Challenges, Standards and Solutions," *IEEE Communications Surveys Tutorials*, vol. 13, no. 4, pp. 584–616, Fourth 2011, doi: 10.1109/SURV.2011.061411.00019.

[7] A. A. Khan, M. Abolhasan, and W. Ni, "5G next generation VANETs using SDN and fog computing framework," in *2018 15th IEEE Annual Consumer Communications Networking Conference (CCNC)*, 2018, pp. 1–6, doi: 10.1109/CCNC.2018.8319192.

[8] A. Kriel, "Investigating the sustainability of the freeway management system and feasibility of implementing a connected vehicle environment in the Western Cape : conceptual design of a connected vehicle environment in Cape Town," Thesis, Stellenbosch : Stellenbosch University, 2017.

[9] Gartner, Inc., "Predicts 2015: The Internet of Things." [Online]. Available: http://www.gartner.com/newsroom/id/2970017. [Accessed: 29-Dec-2016].

[10] T. Koslowski, "Forget the Internet of Things: Here Comes the 'Internet of Cars,'" *WIRED*. [Online]. Available: https://www.wired.com/2013/01/forget-the-internet-of-things-here-comes-the-internet-of-cars/. [Accessed: 29-Dec-2016].

[11] S. Jain, R. C. Shah, W. Brunette, G. Borriello, and S. Roy, "Exploiting Mobility for Energy Efficient Data Collection in Wireless Sensor Networks," *Mob. Netw. Appl.*, vol. 11, no. 3, pp. 327–339, Jun. 2006, doi: 10.1007/s11036-006-5186-9.

[12] U. Lee, E. Magistretti, B. Zhou, M. Gerla, P. Bellavista, and A. Corradi, "Efficient data harvesting in mobile sensor platforms," in *Fourth Annual IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOMW'06)*, 2006, pp. 5 pp.–356, doi: 10.1109/PERCOMW.2006.47.

[13] U. Lee, B. Zhou, M. Gerla, E. Magistretti, P. Bellavista, and A. Corradi, "Mobeyes: smart mobs for urban monitoring with a vehicular sensor network," *IEEE Wireless Communications*, vol. 13, no. 5, pp. 52–57, Oct. 2006, doi: 10.1109/WC-M.2006.250358.

[14] Liu Nanjie, "Internet of Vehicles : Your next connection," *Huawei WinWin*, vol. 11, pp. 23–28, 2011.

[15] F. Yang, S. Wang, J. Li, Z. Liu, and Q. Sun, "An overview of Internet of Vehicles," *China Communications*, vol. 11, no. 10, pp. 1–15, Oct. 2014, doi: 10.1109/CC.2014.6969789.

[16] W. Wu, Z. Yang, and K. Li, "Chapter 16 - Internet of Vehicles and applications A2 - Buyya, Rajkumar," in *Internet of Things*, A. V. Dastjerdi, Ed. Morgan Kaufmann, 2016, pp. 299–317.

[17] F. Hu and Q. Hao, Eds., *Intelligent Sensor Networks: The Integration of Sensor Networks, Signal Processing and Machine Learning*. Boca Raton, FL: CRC Press, 2012.

[18] S. M. Allen, M. J. Chorley, G. B. Colombo, C. B. Jones, V. Tanasescu, and R. M. Whitaker, "Collective spatial awareness," in *2013 IEEE International Conference on Communications Workshops (ICC)*, 2013, pp. 209–214, doi: 10.1109/ICCW.2013.6649230.

[19] M. Whaiduzzaman, M. Sookhak, A. Gani, and R. Buyya, “A survey on vehicular cloud computing,” *Journal of Network and Computer Applications*, vol. 40, pp. 325–344, Apr. 2014, doi: 10.1016/j.jnca.2013.08.004.

[20] J. Kang, D. Lin, E. Bertino, and O. Tonguz, “From Autonomous Vehicles to Vehicular Clouds: Challenges of Management, Security and Dependability,” in *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, 2019, pp. 1730–1741, doi: 10.1109/ICDCS.2019.00172.

[21] X. Sheng, J. Tang, X. Xiao, and G. Xue, “Sensing as a Service: Challenges, Solutions and Future Directions,” *IEEE Sensors Journal*, vol. 13, no. 10, pp. 3733–3741, Oct. 2013, doi: 10.1109/JSEN.2013.2262677.

[22] US Department of Transportation (USDOT), “Connected Vehicles - CV Pilot Deployment Program,” *Connected Vehicle Applications*. [Online]. Available: https://www.its.dot.gov/pilots/cv_pilot_apps.htm. [Accessed: 22-Jul-2018].

[23] M. McCarthy, M. Seidl, S. Mohan, J. Hopkin, A. Stevens, and F. Ognissanto, “Access to In-vehicle Data and Resources,” European Commission, 2017.

[24] N. B. Truong, G. M. Lee, and Y. Ghamri-Doudane, “Software defined networking-based vehicular Adhoc Network with Fog Computing,” in *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, 2015, pp. 1202–1207, doi: 10.1109/INM.2015.7140467.

[25] Z. He, J. Cao, and X. Liu, “SDVN: enabling rapid network innovation for heterogeneous vehicular communication,” *IEEE Network*, vol. 30, no. 4, pp. 10–15, Jul. 2016, doi: 10.1109/MNET.2016.7513858.

[26] A. Falchetti, C. Azurdia-Meza, and S. Cespedes, “Vehicular cloud computing in the dawn of 5G,” in *2015 CHILEAN Conference on Electrical, Electronics Engineering, Information and Communication Technologies (CHILECON)*, 2015, pp. 301–305, doi: 10.1109/Chilecon.2015.7400392.

[27] A. Bujari, O. Gaggi, C. E. Palazzi, and D. Ronzani, “Would Current Ad-Hoc Routing Protocols be Adequate for the Internet of Vehicles? A Comparative Study,” *IEEE Internet of Things Journal*, vol. 5, no. 5, pp. 3683–3691, Oct. 2018, doi: 10.1109/JIOT.2018.2812727.

[28] W. Liang, Z. Li, H. Zhang, S. Wang, and R. Bie, “Vehicular Ad Hoc Networks: Architectures, Research Issues, Methodologies, Challenges, and Trends,” *International Journal of Distributed Sensor Networks*, vol. 11, no. 8, p. 745303, Aug. 2015, doi: 10.1155/2015/745303.

[29] S. Sato, “Air quality in auto-cabin,” *R&D Review of Toyota CRDL*, vol. 39, no. 1, pp. 36–43, 2004.

[30] X. Cheng, “Air quality in transportation Cabins - Part I: How much do we know about it?,” *ASHRAE Transactions*, vol. 112, no. 2, pp. 505–517, 2006.

[31] K. Galatsis and W. Wlodarski, *Car cabin air quality sensors and systems*. AMERICAN SCIENTIFIC PUBLISHERS, 2006.

[32] R. Asadi and M. Ghatee, “A Rule-Based Decision Support System in Intelligent Hazmat Transportation System,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 5, pp. 2756–2764, Oct. 2015, doi: 10.1109/TITS.2015.2420993.

[33] P. Sallis, C. Dannheim, C. Icking, and M. Maeder, “Air Pollution and Fog Detection through Vehicular Sensors,” in *2014 8th Asia Modelling Symposium*, 2014, pp. 181–186, doi: 10.1109/AMS.2014.43.

[34] J. Ivanecký and S. Mehlhase, “An In-Car Speech Recognition System for Disabled Drivers,” in *Text, Speech and Dialogue*, Springer, Berlin, Heidelberg, 2012, pp. 505–512.

[35] C. Perera, A. Zaslavsky, C. H. Liu, M. Compton, P. Christen, and D. Georgakopoulos, “Sensor Search Techniques for Sensing as a Service Architecture for the Internet of Things,” *IEEE Sensors Journal*, vol. 14, no. 2, pp. 406–420, Feb. 2014, doi: 10.1109/JSEN.2013.2282292.

[36] Y. C. Hsu, C. H. Lin, and W. T. Chen, “Design of a Sensing Service Architecture for Internet of Things with Semantic Sensor Selection,” in *2014 IEEE 11th Intl Conf on Ubiquitous Intelligence and Computing and 2014 IEEE 11th Intl Conf on Autonomic and Trusted Computing and 2014 IEEE 14th Intl Conf on Scalable Computing and Communications and Its Associated Workshops*, 2014, pp. 290–298, doi: 10.1109/UIC-ATC-ScalCom.2014.85.

[37] J.-P. Calbimonte, H. Jeung, O. Corcho, and K. Aberer, "Semantic Sensor Data Search in a Large-Scale Federated Sensor Network," *Proceedings of the 4th International Workshop on Semantic Sensor Networks*, vol. 839, pp. 23–38, 2011.

[38] M. Compton *et al.*, "The SSN ontology of the W3C semantic sensor network incubator group," *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 17, pp. 25–32, Dec. 2012, doi: 10.1016/j.websem.2012.05.003.

[39] A. Haller, K. Janowicz, S. Cox, D. Le Phuoc, K. Taylor, and M. Lefrançois, "Semantic Sensor Network Ontology," 14-Jun-2018. [Online]. Available: https://www.w3.org/TR/2017/REC-vocab-ssn-20171019/. [Accessed: 14-Jun-2018].

[40] M. Faezipour, M. Nourani, A. Saeed, and S. Addepalli, "Progress and Challenges in Intelligent Vehicle Area Networks," *Commun. ACM*, vol. 55, no. 2, pp. 90–100, Feb. 2012, doi: 10.1145/2076450.2076470.

[41] S. Al-Sultan, M. M. Al-Doori, A. H. Al-Bayatti, and H. Zedan, "A comprehensive survey on vehicular Ad Hoc network," *Journal of Network and Computer Applications*, vol. 37, pp. 380–392, Jan. 2014, doi: 10.1016/j.jnca.2013.02.036.

[42] B. T. Sharef, R. A. Alsaqour, and M. Ismail, "Vehicular communication ad hoc routing protocols: A survey," *Journal of Network and Computer Applications*, vol. 40, pp. 363–396, Apr. 2014, doi: 10.1016/j.jnca.2013.09.008.

[43] J. Harri, F. Filali, and C. Bonnet, "Mobility models for vehicular ad hoc networks: a survey and taxonomy," *IEEE Communications Surveys Tutorials*, vol. 11, no. 4, pp. 19–41, 2009, doi: 10.1109/SURV.2009.090403.

[44] H. J. F. Qiu, I. W. H. Ho, C. K. Tse, and Y. Xie, "Technical Report: A Methodology for Studying 802.11p VANET Broadcasting Performance with Practical Vehicle Distribution," *CoRR*, vol. abs/1410.3978, 2014.

[45] S. K. Bhoi and P. M. Khilar, "Vehicular communication: a survey," *IET Networks*, vol. 3, no. 3, pp. 204–217, Sep. 2014, doi: 10.1049/iet-net.2013.0065.

[46] A. Dua, N. Kumar, and S. Bawa, "A systematic review on routing protocols for Vehicular Ad Hoc Networks," *Vehicular Communications*, vol. 1, no. 1, pp. 33–52, Jan. 2014, doi: 10.1016/j.vehcom.2014.01.001.

[47] I. Salhi, M. O. Cherif, and S. M. Senouci, "A New Architecture for Data Collection in Vehicular Networks," in *2009 IEEE International Conference on Communications*, 2009, pp. 1–6, doi: 10.1109/ICC.2009.5198637.

[48] R. Kadakia, A. Bhatt, and L. Kurup, "A Review Paper of Data Aggregation in VANET Architecture."

[49] S. C. Hu, Y. C. Wang, C. Y. Huang, and Y. C. Tseng, "A vehicular wireless sensor network for CO2 monitoring," in *2009 IEEE Sensors*, 2009, pp. 1498–1501, doi: 10.1109/ICSENS.2009.5398461.

[50] I. Salhi, M. O. Cherif, and S. M. Senouci, "A New Architecture for Data Collection in Vehicular Networks," in *2009 IEEE International Conference on Communications*, 2009, pp. 1–6, doi: 10.1109/ICC.2009.5198637.

[51] X. Yu, H. Zhao, L. Zhang, S. Wu, B. Krishnamachari, and V. O. K. Li, "Cooperative Sensing and Compression in Vehicular Sensor Networks for Urban Monitoring," presented at the Communications (ICC), 2010 IEEE International Conference, 2010, pp. 1–5, doi: 10.1109/ICC.2010.5502562.

[52] Z. He and H. Zhang, "Density Adaptive Urban Data Collection in Vehicular Sensor Networks," *Journal of Networks*, vol. 9, no. 8, 2014.

[53] R. Du, C. Chen, B. Yang, N. Lu, X. Guan, and X. Shen, "Effective Urban Traffic Monitoring by Vehicular Sensor Networks," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 1, pp. 273–286, Jan. 2015, doi: 10.1109/TVT.2014.2321010.

[54] H. Huang, L. Libman, and G. Geers, "An Agent Based Data Collection Scheme for Vehicular Sensor Networks," in *2015 24th International Conference on Computer Communication and Networks (ICCCN)*, 2015, pp. 1–9, doi: 10.1109/ICCCN.2015.7288374.

[55] P. A. Eskandarian, "Introduction to Intelligent Vehicles," in *Handbook of Intelligent Vehicles*, A. Eskandarian, Ed. Springer London, 2012, pp. 1–13.

[56] N. Lu, N. Cheng, N. Zhang, X. Shen, and J. W. Mark, "Connected Vehicles: Solutions and Challenges," *IEEE Internet of Things Journal*, vol. 1, no. 4, pp. 289–299, Aug. 2014, doi: 10.1109/JIOT.2014.2327587.

[57] J. A. Guerrero-ibanez, S. Zeadally, and J. Contreras-Castillo, "Integration challenges of intelligent transportation systems with connected vehicle, cloud computing, and internet of things technologies," *IEEE Wireless Communications*, vol. 22, no. 6, pp. 122–128, Dec. 2015, doi: 10.1109/MWC.2015.7368833.

[58] R. E. Sibaï, T. Atéchian, J. B. Abdo, R. Tawil, and J. Demerjian, "Connectivity-aware service provision in vehicular cloud," in *2015 International Conference on Cloud Technologies and Applications (CloudTech)*, 2015, pp. 1–5, doi: 10.1109/CloudTech.2015.7337017.

[59] R. Hussain, J. Son, H. Eun, S. Kim, and H. Oh, "Rethinking Vehicular Communications: Merging VANET with cloud computing," in *2012 IEEE 4th International Conference on Cloud Computing Technology and Science (CloudCom)*, 2012, pp. 606–609, doi: 10.1109/CloudCom.2012.6427481.

[60] J. Wan, J. Liu, Z. Shao, A. V. Vasilakos, M. Imran, and K. Zhou, "Mobile Crowd Sensing for Traffic Prediction in Internet of Vehicles," *Sensors*, vol. 16, no. 1, p. 88, Jan. 2016, doi: 10.3390/s16010088.

[61] B. Kantarci and H. T. Mouftah, "Sensing services in cloud-centric Internet of Things: A survey, taxonomy and challenges," in *2015 IEEE International Conference on Communication Workshop (ICCW)*, 2015, pp. 1865–1870, doi: 10.1109/ICCW.2015.7247452.

[62] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, "Sensing As a Service Model for Smart Cities Supported by Internet of Things," *Trans. Emerg. Telecommun. Technol.*, vol. 25, no. 1, pp. 81–93, Jan. 2014, doi: 10.1002/ett.2704.

[63] S. Alam, M. M. R. Chowdhury, and J. Noll, "SenaaS: An event-driven sensor virtualization approach for Internet of Things cloud," in *2010 IEEE International Conference on Networked Embedded Systems for Enterprise Applications (NESEA)*, 2010, pp. 1–6, doi: 10.1109/NESEA.2010.5678060.

[64] S. Abdelwahab, B. Hamdaoui, M. Guizani, and T. Znati, "Cloud of Things for Sensing-as-a-Service: Architecture, Algorithms, and Use Case," *IEEE Internet of Things Journal*, vol. PP, no. 99, pp. 1–1, 2016, doi: 10.1109/JIOT.2016.2557459.

[65] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog Computing and Its Role in the Internet of Things," in *Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing*, New York, NY, USA, 2012, pp. 13–16, doi: 10.1145/2342509.2342513.

[66] F. Bonomi, R. Milito, P. Natarajan, and J. Zhu, "Fog Computing: A Platform for Internet of Things and Analytics," in *Big Data and Internet of Things: A Roadmap for Smart Environments*, N. Bessis and C. Dobre, Eds. Springer International Publishing, 2014, pp. 169–186.

[67] K. Kai, W. Cong, and L. Tao, "Fog computing for vehicular Ad-hoc networks: paradigms, scenarios, and issues," *The Journal of China Universities of Posts and Telecommunications*, vol. 23, no. 2, pp. 56–96, Apr. 2016, doi: 10.1016/S1005-8885(16)60021-3.

[68] W. Zhang, Z. Zhang, and H. C. Chao, "Cooperative Fog Computing for Dealing with Big Data in the Internet of Vehicles: Architecture and Hierarchical Resource Management," *IEEE Communications Magazine*, vol. 55, no. 12, pp. 60–67, Dec. 2017, doi: 10.1109/MCOM.2017.1700208.

[69] K. Zhu, D. Niyato, P. Wang, E. Hossain, and D. In Kim, "Mobility and Handoff Management in Vehicular Networks: A Survey," *Wirel. Commun. Mob. Comput.*, vol. 11, no. 4, pp. 459–476, Apr. 2011, doi: 10.1002/wcm.853.

[70] J. Haerri, J. P. Jeong, M. I. University, S. Cespedes, and N. Benamar, "Survey on IP-based Vehicular Networking for Intelligent Transportation Systems," RFC Editor, Draft 01, Jul. 2016.

[71] Y. Chung, H. Lee, Y.-H. Choi, and C. Lee, "Proactive Caching and Forwarding Schemes for Seamless Handover in IEEE WAVE Networks," *International Journal of Distributed Sensor Networks*, vol. 9, no. 12, p. 627691, Dec. 2013, doi: 10.1155/2013/627691.

[72] R. Zagrouba, H. Hayouni, and F. Kamoun, “Handover optimization within vehicular networks,” in *2014 World Congress on Computer Applications and Information Systems (WCCAIS)*, 2014, pp. 1–5, doi: 10.1109/WCCAIS.2014.6916575.
[73] A. Böhm, M. Jonsson, A. Böhm, and M. Jonsson, “Handover in ieee 802.11p-based delay-sensitive vehicle-to-infrastructure communication,” Research Report IDE-0924,” Technical Report IDE-0924, 2007.
[74] J. Choi and H. Lee, “Supporting Handover in an IEEE 802.11P-based Wireless Access System,” in *Proceedings of the Seventh ACM International Workshop on VehiculAr InterNETworking*, New York, NY, USA, 2010, pp. 75–80, doi: 10.1145/1860058.1860073.
[75] J. M. Chung, M. Kim, Y. S. Park, M. Choi, S. Lee, and H. S. Oh, “Time Coordinated V2I Communications and Handover for WAVE Networks,” *IEEE Journal on Selected Areas in Communications*, vol. 29, no. 3, pp. 545–558, Mar. 2011, doi: 10.1109/JSAC.2011.110305.
[76] W. Cho, M. Kim, S. Lee, and H. S. Oh, “Implementation of handover under multi-channel operation in IEEE 802.11p based communication systems,” in *ICTC 2011*, 2011, pp. 151–155, doi: 10.1109/ICTC.2011.6082570.
[77] E. Baccelli, T. Clausen, and R. Wakikawa, “IPv6 operation for WAVE #x2014; Wireless Access in Vehicular Environments,” in *2010 IEEE Vehicular Networking Conference*, 2010, pp. 160–165, doi: 10.1109/VNC.2010.5698260.
[78] M. Fazio, C. E. Palazzi, S. Das, and M. Gerla, “Automatic IP Address Configuration in VANETs,” in *Proceedings of the 3rd International Workshop on Vehicular Ad Hoc Networks*, New York, NY, USA, 2006, pp. 100–101, doi: 10.1145/1161064.1161086.
[79] T. Kato, K. Kadowaki, T. Koita, and K. Sato, “Routing and Address Assignment Using Lane/Position Information in a Vehicular Ad Hoc Network,” in *2008 IEEE Asia-Pacific Services Computing Conference*, 2008, pp. 1600–1605, doi: 10.1109/APSCC.2008.298.
[80] R. Baldessari, C. J. Bernardos, and M. Calderon, “GeoSAC - Scalable address autoconfiguration for VANET using geographic networking concepts,” in *2008 IEEE 19th International Symposium on Personal, Indoor and Mobile Radio Communications*, 2008, pp. 1–7, doi: 10.1109/PIMRC.2008.4699949.
[81] S. Cespedes, N. Lu, and X. Shen, “VIP-WAVE: On the Feasibility of IP Communications in 802.11p Vehicular Networks,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 1, pp. 82–97, Mar. 2013, doi: 10.1109/TITS.2012.2206387.
[82] Y. Peng and J. M. Chang, “A Novel Mobility Management Scheme for Integration of Vehicular Ad Hoc Networks and Fixed IP Networks,” *Mobile Netw Appl*, vol. 15, no. 1, pp. 112–125, Feb. 2010, doi: 10.1007/s11036-009-0205-2.
[83] M. Mouton, G. Castignani, R. Frank, and T. Engel, “Enabling vehicular mobility in city-wide IEEE 802.11 networks through predictive handovers,” *Vehicular Communications*, vol. 2, no. 2, pp. 59–69, Apr. 2015, doi: 10.1016/j.vehcom.2015.02.001.
[84] M. Bechler and L. Wolf, “Mobility management for vehicular ad hoc networks,” in *2005 IEEE 61st Vehicular Technology Conference*, 2005, vol. 4, pp. 2294-2298 Vol. 4, doi: 10.1109/VETECS.2005.1543744.
[85] X. Wang, D. Le, and Y. Yao, “A cross-layer mobility handover scheme for IPv6-based vehicular networks,” *AEU - International Journal of Electronics and Communications*, vol. 69, no. 10, pp. 1514–1524, Oct. 2015, doi: 10.1016/j.aeue.2015.07.003.
[86] T. T. Nguyen and C. Bonnet, “A hybrid centralized-Distributed Mobility Management for supporting highly mobile users,” in *2015 IEEE International Conference on Communications (ICC)*, 2015, pp. 3945–3951, doi: 10.1109/ICC.2015.7248940.
[87] S. Cespedes, X. Shen, and C. Lazo, “IP mobility management for vehicular communication networks: challenges and solutions,” *IEEE Communications Magazine*, vol. 49, no. 5, pp. 187–194, May 2011, doi: 10.1109/MCOM.2011.5762817.
[88] I. Soto, C. J. Bernardos, M. Calderon, A. Banchs, and A. Azcorra, “Nemo-enabled localized mobility support for internet access in automotive scenarios,” *IEEE Communications Magazine*, vol. 47, no. 5, pp. 152–159, May 2009, doi: 10.1109/MCOM.2009.4939291.
[89] T. T. Nguyen and C. Bonnet, “A hybrid centralized-distributed mobility management architecture for Network Mobility,” in *2015 IEEE 16th International Symposium on A World*

of Wireless, Mobile and Multimedia Networks (WoWMoM), 2015, pp. 1–9, doi: 10.1109/WoWMoM.2015.7158125.

[90] J. H. Lee, T. Ernst, and N. Chilamkurti, "Performance Analysis of PMIPv6-Based NEtwork MObility for Intelligent Transportation Systems," *IEEE Transactions on Vehicular Technology*, vol. 61, no. 1, pp. 74–85, Jan. 2012, doi: 10.1109/TVT.2011.2157949.

[91] S. Ryu, J. W. Choi, and K. J. Park, "Performance evaluation of improved fast PMIPv6-based network mobility for intelligent transportation systems," *Journal of Communications and Networks*, vol. 15, no. 2, pp. 142–152, Apr. 2013, doi: 10.1109/JCN.2013.000027.

[92] R. Moskowitz and P. Nikander, "Host Identity Protocol (HIP) Architecture," IETF, RFC 4423, May 2006.

[93] R. Moskowitz, P. Nikander, P. Jokela, and T. Henderson, "Host Identity Protocol," IETF, RFC 5201, Apr. 2008.

[94] J. Laganier, T. Koponen, and L. Eggert, "Host Identity Protocol (HIP) Registration Extension," IETF, RFC 5203, Apr. 2008.

[95] P. Nikander, T. Henderson, C. Vogt, and J. Arkko, "End-Host Mobility and Multihoming with the Host Identity Protocol," IETF, RFC 5206, Apr. 2008.

[96] J. Melen, J. Ylitalo, P. Salmela, and T. Henderson, "Host Identity Protocol-based Mobile Router (HIPMR)," IETF, Draf 02, 2009.

[97] J. Ylitalo, J. Melén, P. Salmela, and H. Petander, "An Experimental Evaluation of a HIP Based Network Mobility Scheme," in *Wired/Wireless Internet Communications*, 2008, pp. 139–151, doi: 10.1007/978-3-540-68807-5_12.

[98] S. Nováczki, L. Bokor, G. Jeney, and S. Imre, *Design and Evaluation of a Novel HIP-Based Network Mobility Protocol*. .

[99] N. Toledo, J. M. Bonnin, M. Higuero, and E. Jacob, "Host Identity Protocol Based NEMO Solutions: An Evaluation of the Signaling Overhead," in *2011 IEEE 73rd Vehicular Technology Conference (VTC Spring)*, 2011, pp. 1–5, doi: 10.1109/VETECS.2011.5956231.

[100] N. Toledo, J. M. Bonnin, M. Higuero, and E. Jacob, "Fundamentals of NeMHIP: An enhanced HIP based NEMO protocol," in *2011 IEEE Consumer Communications and Networking Conference (CCNC)*, 2011, pp. 1132–1133, doi: 10.1109/CCNC.2011.5766353.

[101] S. Cespedes and X. Shen, "An Efficient Hybrid HIP-PMIPv6 Scheme for Seamless Internet Access in Urban Vehicular Scenarios," in *2010 IEEE Global Telecommunications Conference GLOBECOM 2010*, 2010, pp. 1–5, doi: 10.1109/GLOCOM.2010.5684018.

[102] A. Gurtov, *Host Identity Protocol*, 1 edition. Chichester, U.K.: Wiley, 2008.

[103] S. Céspedes and X. Shen, "On Achieving Seamless IP Communications in Heterogeneous Vehicular Networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 6, pp. 3223–3237, Dec. 2015, doi: 10.1109/TITS.2015.2442251.

[104] R. Mijumbi, J. Serrat, J. L. Gorricho, N. Bouten, F. D. Turck, and R. Boutaba, "Network Function Virtualization: State-of-the-Art and Research Challenges," *IEEE Communications Surveys Tutorials*, vol. 18, no. 1, pp. 236–262, Firstquarter 2016, doi: 10.1109/COMST.2015.2477041.

[105] Y. Li and M. Chen, "Software-Defined Network Function Virtualization: A Survey," *IEEE Access*, vol. 3, pp. 2542–2553, 2015, doi: 10.1109/ACCESS.2015.2499271.

[106] D. Kreutz, F. M. V. Ramos, P. E. Veríssimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-Defined Networking: A Comprehensive Survey," *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14–76, Jan. 2015, doi: 10.1109/JPROC.2014.2371999.

[107] L. I. B. López, Á. L. V. Caraguay, L. J. G. Villalba, and D. López, "Trends on virtualisation with software defined networking and network function virtualisation," *IET Networks*, vol. 4, no. 5, pp. 255–263, 2015, doi: 10.1049/iet-net.2014.0117.

[108] Open Networking Foundation, "SDN architecture," ONF, Technical Reference Issue 1, Jun. 2014.

[109] E. Haleplidis, K. Pentikousis, S. Denazis, J. H. Salim, D. Meyer, and O. Koufopavlou, "Software-Defined Networking (SDN): Layers and Architecture Terminology," IRTF, RFC 7426, Jan. 2015.

[110] International Telecommunications Union, "Framework of software-defined networking," ITU, Recommendation ITU-T Y.3300, Jun. 2014.

[111] Z. Bojovic, P. D. Bojović, and J. Šuh, "The implementation of Software Defined Networking in enterprise networks," *Journal of the Institute of Telecommunications Professionals*, vol. 12, no. 1, pp. 30–35, Mar. 2018.
[112] J. Durand, "Le SDN pour les nuls," presented at the Journées Réseaux de l'Enseignement et de la Recherche, 2015, pp. 1–12.
[113] F. Hu, Q. Hao, and K. Bao, "A Survey on Software-Defined Network and OpenFlow: From Concept to Implementation," *IEEE Communications Surveys Tutorials*, vol. 16, no. 4, pp. 2181–2206, Fourthquarter 2014, doi: 10.1109/COMST.2014.2326417.
[114] Open Networking Foundation, "OpenFlow Switch Specifcation," ONF, Specification Version 1.4.0, 2013.
[115] F. Yang, S. Wang, J. Li, Z. Liu, and Q. Sun, "An overview of Internet of Vehicles," *China Communications*, vol. 11, no. 10, pp. 1–15, Oct. 2014, doi: 10.1109/CC.2014.6969789.
[116] I. Ku, Y. Lu, M. Gerla, R. L. Gomes, F. Ongaro, and E. Cerqueira, "Towards software-defined VANET: Architecture and services," in *Ad Hoc Networking Workshop (MED-HOC-NET), 2014 13th Annual Mediterranean*, 2014, pp. 103–110, doi: 10.1109/MedHocNet.2014.6849111.
[117] A. Kazmi, M. A. Khan, and M. U. Akram, "DeVANET: Decentralized Software-Defined VANET Architecture," in *2016 IEEE International Conference on Cloud Engineering Workshop (IC2EW)*, 2016, pp. 42–47, doi: 10.1109/IC2EW.2016.12.
[118] M. A. Salahuddin, A. Al-Fuqaha, and M. Guizani, "Software-Defined Networking for RSU Clouds in Support of the Internet of Vehicles," *IEEE Internet of Things Journal*, vol. 2, no. 2, pp. 133–144, Apr. 2015, doi: 10.1109/JIOT.2014.2368356.
[119] H. Li, M. Dong, and K. Ota, "Control Plane Optimization in Software-Defined Vehicular Ad Hoc Networks," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 10, pp. 7895–7904, Oct. 2016, doi: 10.1109/TVT.2016.2563164.
[120] Q. Zheng, K. Zheng, H. Zhang, and V. C. M. Leung, "Delay-Optimal Virtualized Radio Resource Scheduling in Software-Defined Vehicular Networks via Stochastic Learning," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 10, pp. 7857–7867, Oct. 2016, doi: 10.1109/TVT.2016.2538461.
[121] K. Zheng, L. Hou, H. Meng, Q. Zheng, N. Lu, and L. Lei, "Soft-defined heterogeneous vehicular network: architecture and challenges," *IEEE Network*, vol. 30, no. 4, pp. 72–80, Jul. 2016, doi: 10.1109/MNET.2016.7513867.
[122] M. Zhu, J. Cao, D. Pang, Z. He, and M. Xu, "SDN-based routing for efficient message propagation in VANET," 2015, doi: 10.1007/978-3-319-21837-3_77.
[123] Y. C. Liu, C. Chen, and S. Chakraborty, "A Software Defined Network architecture for GeoBroadcast in VANETs," in *2015 IEEE International Conference on Communications (ICC)*, 2015, pp. 6559–6564, doi: 10.1109/ICC.2015.7249370.
[124] M. Wang *et al.*, "Use of a mobile laboratory to evaluate changes in on-road air pollutants during the Beijing 2008 Summer Olympics," *Atmospheric Chemistry and Physics*, vol. 9, no. 21, pp. 8247–8263, Nov. 2009, doi: https://doi.org/10.5194/acp-9-8247-2009.
[125] D. Franck, J. Bernière, D. Viltard, F. Parre, C. Challeton-de Vathaire, and M. Agarande, "Development of two mobile laboratories for a routine and accident monitoring of internal contamination," *Appl Radiat Isot*, vol. 70, no. 7, pp. 1095–1099, Jul. 2012, doi: 10.1016/j.apradiso.2012.03.022.
[126] M. C. Batistatos, G. V. Tsoulos, and G. E. Athanasiadou, "Mobile telemedicine for moving vehicle scenarios: Wireless technology options and challenges," *Journal of Network and Computer Applications*, vol. 35, no. 3, pp. 1140–1150, May 2012, doi: 10.1016/j.jnca.2012.01.003.
[127] P. Bolli *et al.*, "The Mobile Laboratory for Radio-Frequency Interference Monitoring at the Sardinia Radio Telescope," *IEEE Antennas and Propagation Magazine*, vol. 55, no. 5, pp. 19–24, Oct. 2013, doi: 10.1109/MAP.2013.6735468.
[128] E. Gustafsson and A. Jonsson, "Always best connected," *IEEE Wireless Communications*, vol. 10, no. 1, pp. 49–55, Feb. 2003, doi: 10.1109/MWC.2003.1182111.

[129] S. Novaczki, L. Bokor, and S. Imre, “Micromobility support in HIP: survey and extension of host identity protocol,” in *MELECON 2006 - 2006 IEEE Mediterranean Electrotechnical Conference*, 2006, pp. 651–654, doi: 10.1109/MELCON.2006.1653184.
[130] M. Muslam, H. A. Chan, and N. Ventura, “Inter-subnet localized mobility support for host identity protocol,” *J Wireless Com Network*, vol. 2011, no. 1, p. 55, Dec. 2011, doi: 10.1186/1687-1499-2011-55.
[131] J. Laganier and L. Eggert, “Host Identity Protocol (HIP) Rendezvous Extension,” IETF, RFC 5204, Apr. 2008.
[132] T. Aura, A. Nagarajan, and A. Gurtov, “Analysis of the HIP Base Exchange Protocol,” in *Information Security and Privacy*, C. Boyd and J. M. G. Nieto, Eds. Springer Berlin Heidelberg, 2005, pp. 481–493.
[133] P. Gustafson, “Meanings of Place: Everyday experience and theoretical conceptualizations,” *Journal of Environmental Psychology*, vol. 21, no. 1, pp. 5–16, Mar. 2001, doi: 10.1006/jevp.2000.0185.
[134] J. Rivera, M. Carrillo, M. Chacón, G. Herrera, and G. Bojorquez, “Self-Calibration and Optimal Response in Intelligent Sensors Design Based on Artificial Neural Networks,” *Sensors*, vol. 7, no. 8, pp. 1509–1529, Aug. 2007, doi: 10.3390/s7081509.
[135] C. Castello, J. Fan, A. Davari, and R. X. Chen, “Temperature Control Framework Using Wireless Sensor Networks and Geostatistical Analysis for Total Spatial Awareness,” in *2009 10th International Symposium on Pervasive Systems, Algorithms, and Networks*, 2009, pp. 717–721, doi: 10.1109/I-SPAN.2009.29.
[136] J. Zhou, L. Hu, F. Wang, H. Lu, and K. Zhao, “An efficient multidimensional fusion algorithm for IoT data based on partitioning,” *Tsinghua Science and Technology*, vol. 18, no. 4, pp. 369–378, Aug. 2013, doi: 10.1109/TST.2013.6574675.
[137] J. Radak, B. Ducourthial, V. Cherfaoui, and S. Bonnet, “Detecting Road Events Using Distributed Data Fusion: Experimental Evaluation for the Icy Roads Case,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 1, pp. 184–194, Jan. 2016, doi: 10.1109/TITS.2015.2464707.
[138] S. Gite and H. Agrawal, “On Context Awareness for Multisensor Data Fusion in IoT,” in *Proceedings of the Second International Conference on Computer and Communication Technologies*, S. C. Satapathy, K. S. Raju, J. K. Mandal, and V. Bhateja, Eds. Springer India, 2016, pp. 85–93.
[139] R. Marin-Perianu, M. Marin-Perianu, P. Havinga, and H. Scholten, “Movement-Based Group Awareness with Wireless Sensor Networks,” in *Pervasive Computing*, A. LaMarca, M. Langheinrich, and K. N. Truong, Eds. Springer Berlin Heidelberg, 2007, pp. 298–315.
[140] N. Kumar, K. Kaur, A. Jindal, and J. J. P. C. Rodrigues, “Providing healthcare services on-the-fly using multi-player cooperation game theory in Internet of Vehicles (IoV) environment,” *Digital Communications and Networks*, vol. 1, no. 3, pp. 191–203, Aug. 2015, doi: 10.1016/j.dcan.2015.05.001.
[141] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, “Context Aware Computing for The Internet of Things: A Survey,” *arXiv:1305.0982 [cs]*, May 2013.
[142] H. Vahdat-Nejad, A. Ramazani, T. Mohammadi, and W. Mansoor, “A survey on context-aware vehicular network applications,” *Vehicular Communications*, vol. 3, pp. 43–57, Jan. 2016, doi: 10.1016/j.vehcom.2016.01.002.
[143] Ö. Yürür, C. H. Liu, Z. Sheng, V. C. M. Leung, W. Moreno, and K. K. Leung, “Context-Awareness for Mobile Sensing: A Survey and Future Directions,” *IEEE Communications Surveys Tutorials*, vol. 18, no. 1, pp. 68–93, Firstquarter 2016, doi: 10.1109/COMST.2014.2381246.
[144] T. L. Saaty, “Decision making with the analytic hierarchy process,” *International Journal of Services Sciences*, vol. 1, no. 1, pp. 83–98, Jan. 2008, doi: 10.1504/IJSSci.2008.01759.
[145] J. A. Alonso and M. T. Lamata, “Consistency in the analytic hierarchy process: a new approach,” *Int. J. Unc. Fuzz. Knowl. Based Syst.*, vol. 14, no. 04, pp. 445–459, Aug. 2006, doi: 10.1142/S0218488506004114.
[146] World Health Organization, “WHO | Air pollution.” [Online]. Available: https://www.who.int/airpollution/en/. [Accessed: 19-Feb-2019].

[147] World Health Organization Regional Office for Europe, “Air quality guidelines for Europe,” *WHO Regional Office for Europe*, vol. 2nd ed. Copenhagen, 2000.

[148] S. van den Elshout, K. Léger, and F. Nussio, “Comparing urban air quality in Europe in real time: A review of existing air quality indices and the proposal of a common alternative,” *Environment International*, vol. 34, no. 5, pp. 720–726, Jul. 2008, doi: 10.1016/j.envint.2007.12.011.

[149] A. A. Abdel-Rahman, “On the dispersion models and atmospheric dispersion,” *International Journal of Global Warming*, vol. 3, no. 3, pp. 257–273, Jan. 2011, doi: 10.1504/IJGW.2011.043422.

[150] S. Krauss, “Microscopic Modeling of Traffic Flow: Investigation of Collision Free Vehicle Dynamics,” Doctoral degree, University of Cologne, Cologne, 1998.

[151] R. and M. ltd, “Global Automotive Cabin Air Quality Sensor Market: Focus on Sensor Types, Vehicle Types, Regional Study (15 Countries), Market Share, and Industry Insights - Analysis and Forecast, 2017-2021.” [Online]. Available: https://www.researchandmarkets.com/reports/4603997/global-automotive-cabin-air-quality-sensor. [Accessed: 15-Feb-2019].

[152] C. Castello, J. Fan, A. Davari, and R. X. Chen, “Temperature Control Framework Using Wireless Sensor Networks and Geostatistical Analysis for Total Spatial Awareness,” in *2009 10th International Symposium on Pervasive Systems, Algorithms, and Networks*, 2009, pp. 717–721, doi: 10.1109/I-SPAN.2009.29.

[153] “Android Automotive,” *Android Open Source Project*. [Online]. Available: https://source.android.com/devices/automotive. [Accessed: 28-Sep-2018].

[154] R. project team, *RYU SDN Framework - English Edition*, vol. Release 1.0. RYU project team, 2014.

[155] L. Sequeira, J. L. de la Cruz, J. Ruiz-Mas, J. Saldana, J. Fernandez-Navajas, and J. Almodovar, “Building an SDN Enterprise WLAN Based on Virtual APs,” *IEEE Communications Letters*, vol. 21, no. 2, pp. 374–377, Feb. 2017, doi: 10.1109/LCOMM.2016.2623602.

[156] L. Suresh, J. Schulz-Zander, R. Merz, A. Feldmann, and T. Vazao, “Towards Programmable Enterprise WLANS with Odin,” in *Proceedings of the First Workshop on Hot Topics in Software Defined Networks*, New York, NY, USA, 2012, pp. 115–120, doi: 10.1145/2342441.2342465.

[157] J. Y. Yen, “Finding the K Shortest Loopless Paths in a Network,” *Management Science*, vol. 17, no. 11, pp. 712–716, Jul. 1971, doi: 10.1287/mnsc.17.11.712.

[158] A. A. Hagberg, D. A. Schult, and P. J. Swart, “Exploring Network Structure, Dynamics, and Function using NetworkX,” in *Proceedings of the 7th Python in Science Conference*, Pasadena, CA, 2008, pp. 11–15.

[159] J. W. Guck and W. Kellerer, “Achieving end-to-end real-time Quality of Service with Software Defined Networking,” in *2014 IEEE 3rd International Conference on Cloud Networking (CloudNet)*, 2014, pp. 70–76, doi: 10.1109/CloudNet.2014.6968971.

[160] S. Tomovic and I. Radusinovic, “Fast and efficient bandwidth-delay constrained routing algorithm for SDN networks,” in *2016 IEEE NetSoft Conference and Workshops (NetSoft)*, 2016, pp. 303–311, doi: 10.1109/NETSOFT.2016.7502426.

[161] R. Kumar *et al.*, “End-to-End Network Delay Guarantees for Real-Time Systems Using SDN,” in *2017 IEEE Real-Time Systems Symposium (RTSS)*, 2017, pp. 231–242, doi: 10.1109/RTSS.2017.00029.

[162] A. Atary and A. Bremler-Barr, “Efficient Round-Trip Time monitoring in OpenFlow networks,” in *IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications*, 2016, pp. 1–9, doi: 10.1109/INFOCOM.2016.7524501.

[163] K. Phemius and M. Bouet, “Monitoring latency with OpenFlow,” in *Proceedings of the 9th International Conference on Network and Service Management (CNSM 2013)*, 2013, pp. 122–125, doi: 10.1109/CNSM.2013.6727820.

[164] J. Schulz-Zander, L. Suresh, N. Sarrar, A. Feldmann, T. Hühn, and R. Merz, “Programmatic Orchestration of WiFi Networks,” in *Proceedings of the 2014 USENIX Conference on USENIX Annual Technical Conference*, Berkeley, CA, USA, 2014, pp. 347–358.

[165] J. Song, Y. Wu, Z. Xu, and X. Lin, “Research on car-following model based on SUMO,” in *The 7th IEEE/International Conference on Advanced Infocomm Technology*, 2014, pp. 47–55, doi: 10.1109/ICAIT.2014.7019528.
[166] T. H. Clausen and P. Jacquet, “Optimized Link State Routing Protocol (OLSR),” RFC Editor, RFC 3626, Oct. 2003.
[167] B. Karp and H. T. Kung, “GPSR: Greedy Perimeter Stateless Routing for Wireless Networks,” in *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking*, New York, NY, USA, 2000, pp. 243–254, doi: 10.1145/345910.345953.

ANNEXES

Annexe 1 : Schémas de mobilité HIP proposés dans la littérature

1. Schéma de mobilité HIP natif

La procédure de mobilité du protocole HIP natif, voir *Figure A1.1*, est indifférente de l'architecture du domaine. Dès qu'il y a changement d'adresse IP, des paquets UPDATE sont échangés entre toutes les partenaires de communication (CN1, CN2) du nœud mobile (MN).

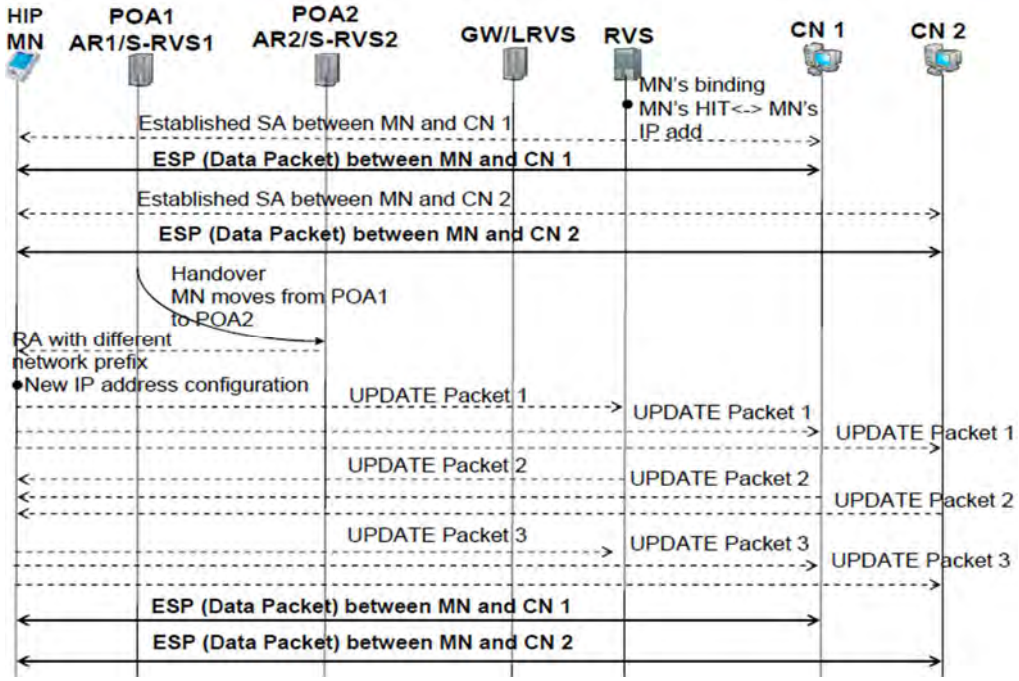


Figure A1.1 : Procédure de mobilité via HIP [130]

2. Schéma de mobilté de Muslam et al.

L'approche de Muslam et al. [130] est très simple : le MN devra toujours garder la même adresse IP en cas de handover. Pour ce faire, lorsque le MN quitte son ancien PoA pour se lier à un autre, le S-RVS2 le détecte et envoie un *UPDATE packet 1* au LRVS. Lorsque le LRVS reçoit ce paquet, il met à jour les enregistrements d'attachement du MN et envoie à son tour un paquet *UPDATE packet 2* au S-RVS2. En utilisant le contenu du *UPDATE packet 2*, le S-RVS2 envoie un RA au MN en y incluant le même préfixe réseau qu'avait ce dernier avant de changer de PoA. Cette procédure de mobilité est résumée par la *Figure A1.2*.

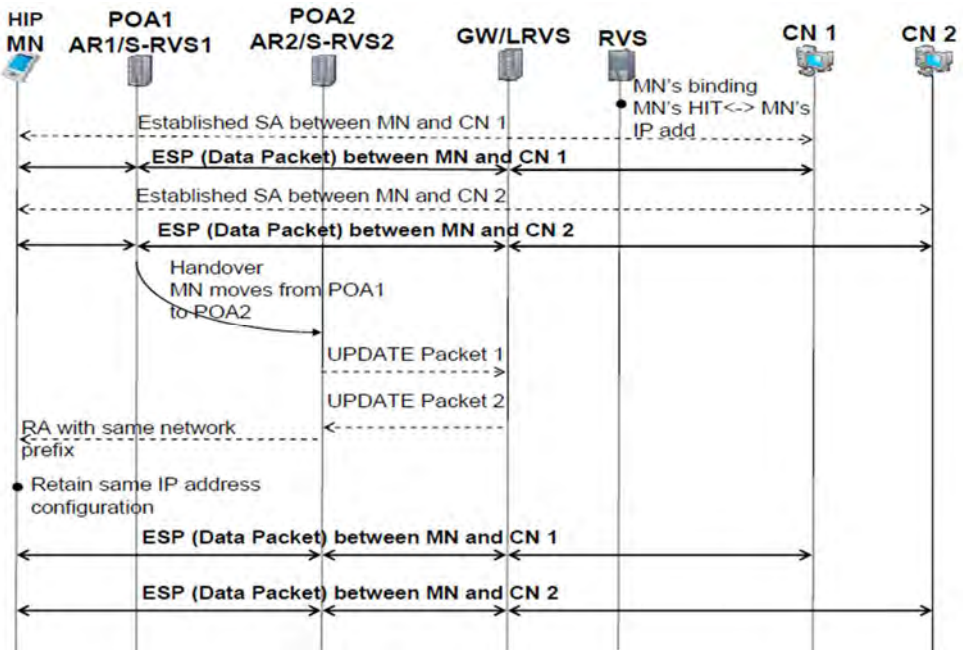
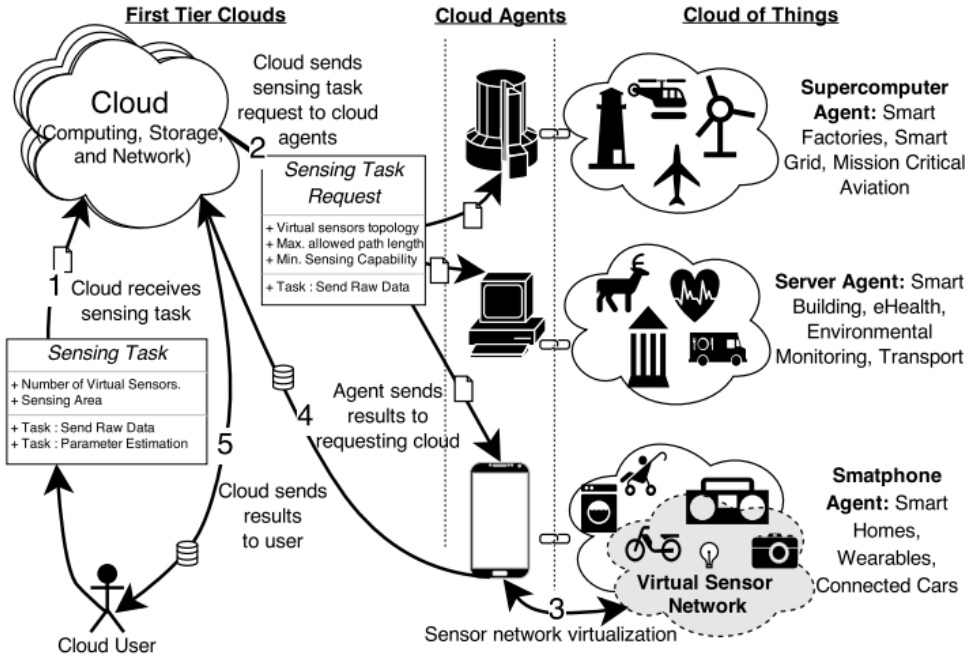


Figure A1.2 : Procédure de mobilité via HIP$_{MUSLAM}$ [130]

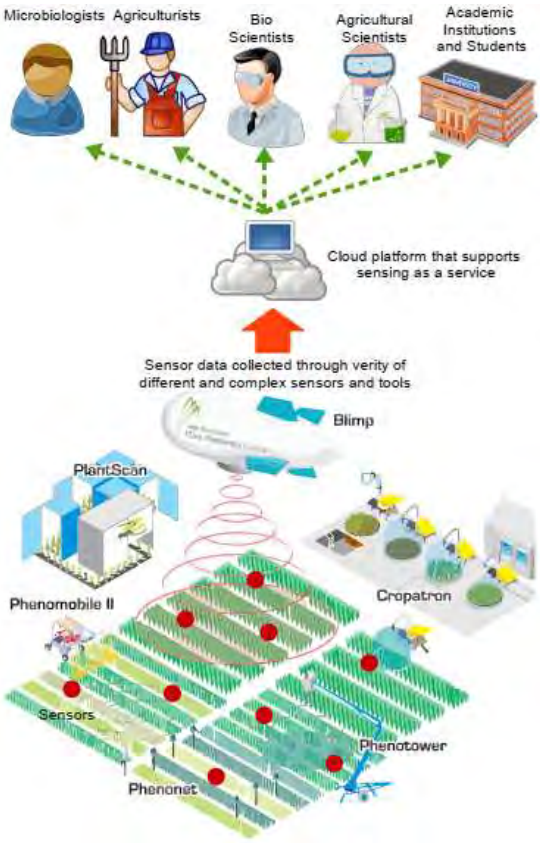
Annexe 2 : Architectures S^{2}aaS proposées dans la littérature

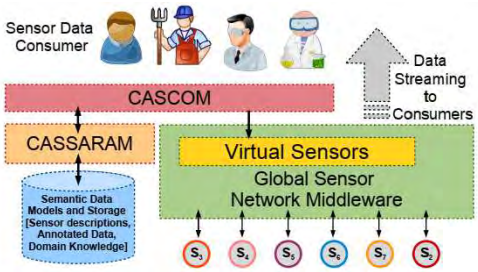
1. Modèle S^{2}aaS de Abdelwahab et al. [64]



2. Modèle S^{2}aaS de Hsu et al. [36]

3. Modèle S^{2}aaS de Perera et al. [35]





Annexe 3 : Principe de fonctionnement de Odin

Odin permet d'implémenter une politique de gestion de réseaux WiFi via la création de Points d'Accès Virtuels (LVAP, Light Virtual Access Point) qui virtualisent les associations entre l'AP physique et les stations clientes (STA). En d'autres termes, Odin permet de mettre en place une plateforme programmable en utilisant l'approche SDN permettant aux fonctionnalités et services réseaux d'être manipulés comme des applications. Le LVAP est un tuple constitué de l'adresse MAC de la station (STA_MAC), de l'adresse IP de la station (STA_IP), d'un BSSID virtuel de l'AP (AP_vBSSID) et d'un SSID virtuel de l'AP (AP_vSSID).

$$LVAP = (STA\_MAC, STA\_IP, AP\_vBSSID, AP\_vSSID)$$

Chaque station cliente reçoit un AP_vBSSID unique, ce qui lui donne l'illusion d'être le seul client de l'AP physique. L'architecture de Odin, *Figure A3.1*, est constituée de deux entités, à savoir :

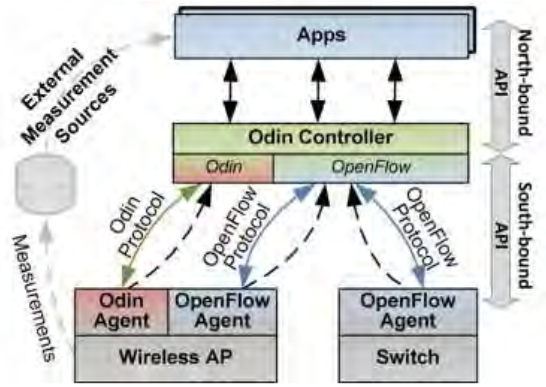


Figure A3.1 : Architecture de Odin

- *Odin Controller*

Ce contrôleur est déployé sur un serveur Linux et permet la programmation des applications de contrôle à travers une API northbound et leurs interactions avec les AP à travers une API southbound. Ces interactions se résument essentiellement à l'envoi de commandes de création et de suppressions de LVAP, de commandes de collectes d'information de monitoring…

- *Odin Agent*

Cet agent est déployé sur chaque AP et c'est à ce niveau où seront créés les LVAP ; ce qui permet la gestion des ressources via le contrôleur. L'agent permet également de collecter des informations de monitoring du réseaux WiFi et de les remonter vers le contrôleur.

Le protocole Openflow est utilisé comme API southbound et est responsable de la gestion des switches embarqués dans les AP. Le protocole Odin est une extension d'Openflow dédiée à la gestion des interfaces IEEE 802.11n/ac. Le

contrôleur SDN floodlight[6] est utilisé pour l'implémentation du Odin Controller. Au niveau de chaque AP, Open vSwitch[7] est installé afin de bénéficier virtuellement de fonctionnalités d'un commutateur. Click Modular Router[8] est également installé sur chaque AP afin de faire bénéficier à ce dernier des fonctionnalités de routage. Le projet Wi-5 offre des fonctionnalités supplémentaires, voir *Figure A3.2,* et met l'accent sur la gestion du spectre radio.

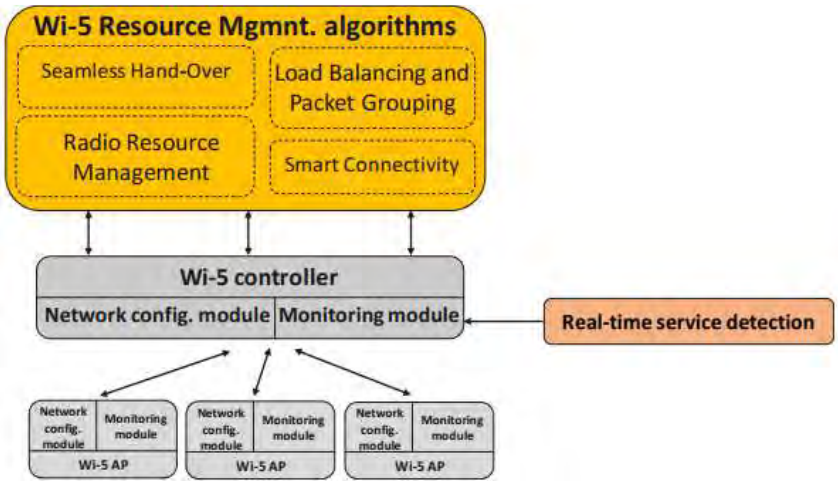


Figure A3.2 : fonctionnalités proposées dans le projet Wi-5

Dans l'architecture Wi-5, *Figure A3.3*, le plan de contrôle est subdivisé en commandes Openflow (ports TCP 6633, 6655) destinées à Open vSwitch et en commandes Odin (ports UDP 2819, TCP 6777) destinées à l'Odin Agent.Le plan de données des stations clientes (véhicules) transite via les LVAP créés au niveau de l'interface IEEE 802.11n/ac de l'AP. Le Odin Agent en s'aidant du routeur Click, route les paquets vers le commutateur Open vSwitch. Ce dernier se chargera de transférer ce plan de données vers le réseau WAN (Internet).

[6] http://www.projectfloodlight.org/floodlight/
[7] https://www.openvswitch.org
[8] https://github.com/kohler/click

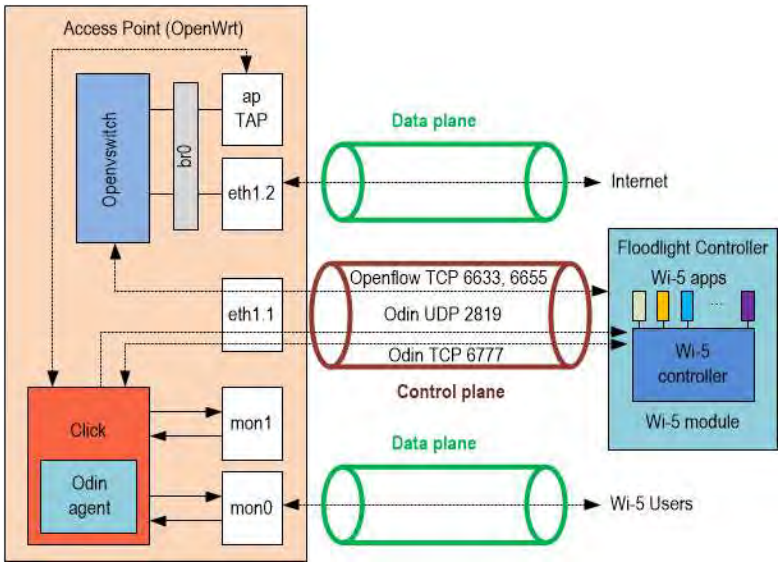


Figure A3.3 : Organisation des entités réseaux dans le projet Wi-5

Pour qu'un véhicule (STA) se connecte au réseau de test, un LVAP devra lui être assigné. Ce processus se résume comme suit :

- *Découverte*

Lorsque le véhicule effectue un scan actif sur l'ensemble des canaux WiFi, l'Odin Agent capte les trames balises générées (Probe Request) et le transfert au Wi-5 controler. Ce dernier génère alors un BSSID unique associé à ce véhicule et donne l'instruction à l'Odin Agent de générer une trame balise (Probe Response) adressée au véhicule.

- *Association*

Lorsque le véhicule envoie la trame d'association (Association Request), l'Odin Agent transfert cette trame au Wi-5 controler. Si aucun LVAP n'est associé au véhicule, le contrôleur créé un nouveau LVAP et envoie la commande de création du LVAP à l'agent. Ce dernier créé donc un LVAP pour le véhicule concerné et conclut l'association (Association Response).

- *DHCP et ARP*

Après l'association avec l'AP, le véhicule déclenche la transaction DHCP afin d'obtenir sa configuration IP. Lors de cette transaction, l'Odin Agent capture le message DHCPACK et le transfert au Wi-5 controler afin que ce dernier met à jour le *STA_IP* avec l'adresse IP contenu le DHCPACK. Ensuite le contrôleur envoie une commande Openflow à l'Odin Agent pour instruire Open vSwitch des règles de commutation à appliquer sur le flux relatif à cette adresse IP. La connaissance de l'adresse IP du véhicule permet également au Wi-5 controler de pouvoir gérer les requêtes ARP associées au LVAP du véhicule.

Annexe 4 : Dispositif expérimental

Le dispositif expérimental est constitué d'un ensemble de serveurs, de switches SDN et de dispositifs de communication sans fil.

- *SDNBACKBONE*

Le SDNBACKBONE est constitué d'un contrôleur SDN à base du framework Ryu. Le schéma de routage basé sur l'application PureSDN sera utilisé comme application SDN. Le contrôleur est installé sur une machine virtuelle (Processeur : 2 cœurs à 2.4Ghz, RAM : 2GB, OS : Ubuntu Server 14.04). Le backbone est constitué de switches SDN Zodiac FX, voir *Figure A4.1*, dont les caractéristiques sont données par le *Tableau A3.1*.

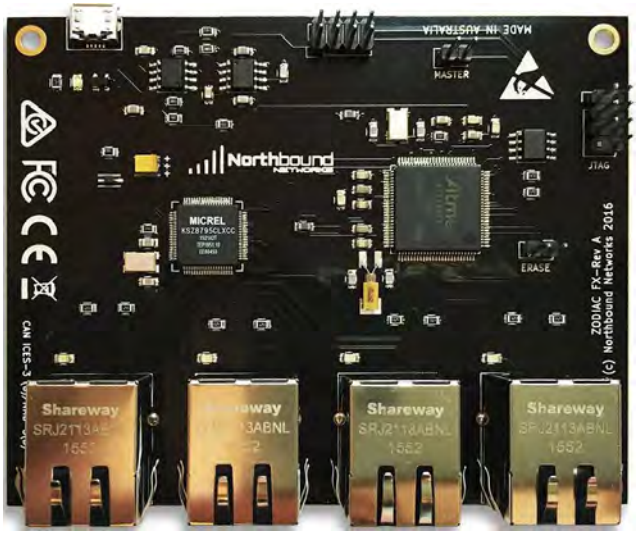


Figure A4.1: Zodiac FX

Tableau A4.1 : Caractéristiques du switch SDN Zodiac FX

Amtel ATSAM4E Cortex M4 processor	CPU : 120MHz
	RAM : 128KB
	FLASH : 1MB
Ethernet	3x 10/100Mbps ports OpenFlow
	1x 10/100Mbps port de contrôle
OpenFlow	Version 1.0, 1.3 & 1.4
	512 entrées maximale sur la table de flux
Configuration d'usine	Device Name: Zodiac3_FX IP Address: 10.0.1.99 Netmask: 255.255.255.0 Default Gateway: 10.0.1.1 OpenFlow Controller: 10.0.1.8 OpenFlow port: 6633

- *SDNRAN*

Le Contrôleur SDNRAN (W-i5 Odin Controller) est installé sur une machine virtuelle (Processeur : 2 cœurs à 2.4Ghz, RAM : 2GB, OS : Ubuntu Server 14.04). Les RAN sont constitués de deux Odin Agents installés sur des points d'accès TP-LINK AC1750 v2 (Archer C7), voir *Figure A3.2*, dont les caractéristiques sont fournies au niveau du *Tableau A3.2*. Ces deux points d'accès fonctionnent à base du système d'exploitation Openwrt 15.05 (Chaos Calmer).

Figure A4.2: TP-LINK AC1750 v2

Tableau A3.2 : Caractéristiques du TP-LINK AC1750 v2

SoC Qualcomm Atheros QCA9558	CPU : 720MHz
	RAM : 120MB
	FLASH : 16MB
WLAN (QCA9558 + QCA9880-BR4A)	802.11b/g/n 2.4GHz (450Mbps)
	802.11a/n/ac 5.0GHz (1300Mbps)
Ethernet	4x 10/100/1000Mbps LAN Ports
	1x 10/100/1000Mbps WAN Port
USB	2x USB 2.0 Port
Antenne	3x Antennes détachables Dual Band (RP-SMA)

- *Ordinateur de bord*

L'hyperviseur est constituée d'un ordinateur monocarte Raspberry Pi3, voir *Figure A3.3*. Les caractéristiques de la Raspberry sont fournies au niveau du *Tableau A3.3*. Le système d'exploitation *Raspbian* est installé sur la carte, ce système dérive de Debian et est optimisé pour fonctionner sur les processeurs ARM.

Figure A4.3 : Raspberry Pi3

Tableau A3.3 : Caractéristiques de la Raspberry Pi3

SoC Broadcom BCM2835	CPU 4× ARM Cortex-A53 x64, 1.2 GHz
	RAM 1GB LPDDR2 (900 MHz)
	GPU Broadcom VideoCore IV 1080p (Full HD)
Réseaux	WiFi 2.4GHz 802.11b/g/n
	Bluetooth 4.1 Low Energy (BLE) et classique
	Ethernet 10/100
Alimentation	5V, 2.5A (port USB)
	2 sources de tension de 3.3V & 2 sources de tension de 5V
Connectique	26 E/S à usage générale à 3.3V numérotée de GPIO0 à GPIO17
	4x port USB 2.0, 1 port micro SD, 1 port HDMI, 1 port Caméra
	1 interface UART, 1 interface SPI, 1 interface I^2C

L'hyperviseur est également doté d'un modem 4G LTE USB Huawei E8372 doté d'une antenne interne et de deux antennes externes détachables pour booster le signal, voir *Figure A4.4*.

Figure A4.4 : Modem 4G LTE USB Huawei E8372