UNIVERSITÉ DE MONTRÉAL

A DYNAMIC MESSAGING ARCHITECTURE FOR VEHICULAR SOCIAL NETWORKS

SAIDA MAAROUFI
DÉPARTEMENT DE GÉNIE INFORMATIQUE ET GÉNIE LOGICIEL
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

THÈSE PRÉSENTÉE EN VUE DE L'OBTENTION
DU DIPLÔME DE PHILOSOPHIÆ DOCTOR
(GÉNIE INFORMATIQUE)
JANVIER 2016

© Saida Maaroufi, 2016.

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Cette thèse intitulée:

A DYNAMIC MESSAGING ARCHITECTURE FOR VEHICULAR SOCIAL NETWORKS

présentée par: MAAROUFI Saida
en vue de l'obtention du diplôme de: Philosophiæ Doctor
a été dûment acceptée par le jury d'examen constitué de:

Mme BELLAÏCHE Martine, Ph. D., présidente
M. PIERRE Samuel, Ph. D., membre et directeur de recherche
M. QUINTERO Alejandro, Doctorat, membre
M. KARMOUCH Ahmed, Ph. D., membre externe

DEDICATION

Dedicated to my loving parents and to my advisor from whom I have learned so much...

ACKNOWLEDGMENTS

It is a great pleasure to gratefully acknowledge many people who greatly helped me during the difficult, challenging, and exciting journey towards a Ph.D. None of this work could have been possible without their input.

First and foremost, I would like to express my sincere and immense gratitude to my research director, Professor Samuel Pierre, for everything I learned from him and that will last forever.

I have been amazingly fortunate to have M. Pierre as my advisor and I would not have considered a graduate research career without his confidence in my abilities, his continuous support and motivation. Prof. Samuel Pierre made a big difference in my life and he became more of a father than a professor. He provided me with his wide and deep knowledge in many areas, his challenging supervision and his technical support while also allowing me to work freely and independently.

I am deeply thankful for the great opportunities he gave me that have considerably added to my research career. His guidance during these difficult years, his constant availability, his constructive advice when I was lost, his continuous encouragement to always reach higher, think bigger and try harder have strengthen my will to always keep moving forward.

M. Pierre, thank you very much for being such an inspirational person in my life.

I am also grateful to Dr. Adrian Holzer for the fruitful discussions that helped me develop innovative and great ideas, for his constructive criticisms at different stages of my research and for being challenging.

I also would like to thank the members of my Jury, Professors Ahmed Karmouch, Alejandro Quintero and Martine Bellaiche for the time spent reviewing my thesis and for the valuable input they provided.

I am very much grateful to Professor Haidar Safa, with whom I organized an international workshop on topics related to my thesis. I thank him for his enthusiastic support, his helpful advice and for the wonderful time I spent working with him.

I am thankful to my friends and colleagues at our LARIM Research Laboratory, especially my colleague, Ronald Jean Julien for his great help and assistance and for having sacrificed his valuable time to read my thesis and paper drafts and make relevant comments for which I am very grateful. Many thanks go to Grégory Charlot, Tarik Lakhbizi, Ahmed Salim Chekkouri, Valérie Justafort, Marième Diallo, Éric Fafolahan and Nasrin Taherkhani for being supportive during the past years and for having made every day of work an enjoyable experience.

I am grateful to the administrative staff of the Computer Engineering department who have helped me in several aspects of my academic life at École Polytechnique de Montréal. In particular, I would like to thank Sabine Kébreau, Micheline Lafrenière and Louise Longtin for their constant help and great support.

I have been fortunate to have met great friends throughout my Ph.D. journey. They were all supportive and helped me overcome setbacks and stay focused on my graduate studies.

Last but not least, I am forever indebted to my parents and the rest of my family. My parents have been a great source of inspiration and support to me. Their prayers have always accompanied me. None of this would have been possible without their sacrifices, patience and continuous encouragements. I deeply appreciate their belief in me and I dedicate this thesis to them.

RÉSUMÉ

La congestion routière et les longs trajets quotidiens sont deux grandes sources d'insatisfaction chez les voyageurs. Ces derniers partagent les mêmes besoins et intérêts au niveau de la route mais leur anonymité, leur manque de confiance en leur voisinage et la grande mobilité des véhicules les rendent incapables d'entamer et de maintenir une communication sure et stable entre eux, pendant les heures de pointe et les bouchons de circulation, afin de gérer ensemble l'état critique et imprévisible du trafic routier.

Ceci déclenche chez eux des sentiments de frustration, augmente leur niveau de stress et les pousse à s'envoyer des messages de façon aveugle, ce qui empire la situation du trafic, congestionne le réseau, augmente les délais d'attente de réception d'information utiles et affecte négativement la qualité de leur voyage et leur état psychologique.

Par ailleurs et en l'absence de congestion, certains voyageurs considèrent la longue durée de leur voyage comme du temps perdu à ne rien faire, d'autres utilisent les applications mobiles de géolocalisation et partagent leurs informations contextuelles avec leurs amis via les réseaux sociaux virtuels, ce qui est considéré comme un moyen de divertissement tout au long de leur voyage.

En fait, l'anonymité des voyageurs au niveau de la route est alimentée par leur hésitation à partager leurs intérêts avec un public inconnu, ce qui pourrait les exposer aux problèmes de fuite de données personnelles et mettre en péril leur identité et information personnelles.

Plusieurs travaux de recherche ont proposé des architectures véhiculaires qui favorisent et facilitent le développement de services et d'applications véhiculaires traditionnels orientés véhicule, qui visent principalement à améliorer la sécurité routière et à prévenir les accidents. D'autres travaux plus récents sont motivés par l'amélioration de la qualité du voyage et à offrir aux voyageurs des services de divertissement. Ces travaux proposent l'introduction du concept des réseaux sociaux dans les réseaux véhiculaires ad hoc afin de faciliter la formation de communautés véhiculaires sociales où les voyageurs sont regroupés en fonction de leur contexte et leurs intérêts sur la route. Pour ce faire, les auteurs de ces travaux ont procédé à la conception d'architectures véhiculaires sociales distribuées ou centralisées qui supportent le développement des applications véhiculaires orientées utilisateur.

Cependant, ces architectures proposées ne sont pas hybrides. De plus, elles ne tiennent pas compte de la coopération entre les couches supérieures de l'architecture OSI (services et applications) et les couches inférieures responsables du routage de l'information contextuelle

dynamique propagée à travers le réseau. En outre, ces dernières devraient aussi considérer la fragilité et l'instabilité des liens de communication entre les véhicules, ce qui empêche les voyageurs de maintenir une communication fiable et efficace sur la route et cause des collisions lors de la dissémination des messages dans des environnements véhiculaires denses. Par ailleurs, le coût d'accès à l'infrastructure induit une grande consommation de la bande passante dans le cas où la majorité des voyageurs se connectent à Internet simultanément dans un environnement véhiculaire hybride et dense.

Cette thèse vient combler le vide d'architecture véhiculaire hybride dans les réseaux véhiculaires sociaux et propose un nouveau système de messagerie dynamique hybride, fiable, stable et efficace pour ce type de réseaux, appelé DYMES. Une telle architecture permet de favoriser les interactions entre les voyageurs en temps-réel, tout en respectant leur anonymité sur la route et en tenant compte de la dynamicité de leurs informations contextuelles partagées à travers le réseau, en utilisant un ensemble d'abstractions de communication fiable, efficace, distribué et centralisé, dans un environnement véhiculaire hybride et dense.

Ce travail est subdivisé en trois volets importants qui ont fait l'objet d'articles scientifiques.

Le premier volet consiste à identifier une méta-stratégie qui guide la conception des abstractions de communication hybrides sur lesquelles repose DYMES. Nous proposons l'utilisation du modèle de publication et de souscription aux services qui concorde avec la nature dynamique des réseaux véhiculaires et qui répond aux requis et aux besoins des voyageurs au niveau de la route en terme du respect de l'anonymité de leur identité. Dans ce modèle, les récipiendaires des messages publiés sont identifiés par leur contexte et non par leur identité.

De ce fait, nous concevons et introduisons des abstractions dynamiques de publication et de souscription aux services qui visent à assurer une communication anonyme entre les voyageurs en leur permettant de publier leur information contextuelle dynamique et de souscrire en utilisant des filtres dynamiques sensibles au contexte des messages. Nous illustrons l'utilisation de DYMES et montrons son fonctionnement via deux applications véhiculaires sociales distribuées et centralisées. De plus, nous identifions et nous discutons nos choix d'implémentation des abstractions centralisées, distribuées et hybrides proposées qui guident la conception du système DYMES.

Le deuxième volet propose un nouvel ensemble d'abstractions de publication et de souscription dynamiques, hybrides, efficientes et fiables qui représente un module de l'architecture DYMES.

Notre première abstraction est une stratégie de publication et de souscription dynamique aux services de regroupement DPSCS (Dynamic Publish/Subscribe Clustering Strategy) qui

aborde les problématiques de l'isolation des voyageurs au niveau de la route et de l'instabilité de leur liens de communication.

DPSCS permet à chaque voyageur de former une communauté stable basée sur son propre contexte et intérêt, qui est capable de s'auto mettre à jour de façon efficiente et fiable tout en respectant l'anonymité des voyageurs sur la route. Pour ce faire, chaque voyageur qui désire communiquer avec son entourage en créant une communauté, publie une seule publication persistante, dans un espace déterminé, dont le contenu est dynamique. DPSCS repose sur un protocole de communication qui permet de propager cette publication de manière efficiente et fiable en sélectionnant les relais qui disposent d'un lien stable avec le nœud source (l'éditeur) de la publication et qui sont situés loin de ce nœud. Les voyageurs dont les souscriptions courantes concordent avec la publication dynamique du nœud source, joignent sa communauté. La persistance de la publication envoyée détermine la fin de la formation de la communauté.

Notre deuxième abstraction proposée est une stratégie de découverte et de sélection de relais mobiles appelée MGDSS (Mobile Gateway Discovery/Selection Strategy). Cette dernière aborde les problématiques de découverte et de sélection de relais mobiles dans les environnements véhiculaires hybrides qui donnent naissance à d'autres problèmes comme la grande consommation de bande passante lors de l'accès d'un grand nombre de véhicules à l'infrastructure, et le nombre exponentiel de messages envoyés entre les voyageurs dans les environnements véhiculaires denses.

MGDSS se base sur le résultat de DPSCS. Elle facilite aux voyageurs groupés dans des communautés la découverte de leur entourage en leur permettant de s'envoyer un seul message chacun, qui est propagé de façon fiable et efficiente en utilisant un nouveau protocole de diffusion nommé CoCo (Context-aware Coding). Ce dernier permet de réduire le nombre de retransmissions des messages à travers la communauté et d'en assurer la livraison par les voyageurs. Ces derniers procèdent à la sélection d'un nombre minimal de nœuds relais capables de les lier à l'infrastructure en se basant sur les informations qu'ils ont reçues à travers MGDS-CoCo. Notre stratégie de sélection nommée MGSS (Mobile Gateway Selection Strategy) permet aux voyageurs d'envoyer leurs publications et leurs souscriptions à l'infrastructure à travers les relais sélectionnés en utilisant les chemins les plus stables.

Une nouvelle stratégie de correspondance entre publication et souscription est implémentée au niveau de l'infrastructure et permet de comparer et de mettre à jour les souscriptions des voyageurs et de les filtrer selon leur contenu dynamique avec les publications reçues. Le résultat est renvoyé aux souscripteurs concernés via le relais sélectionné en utilisant MGSS.

L'évaluation de performance de cet ensemble d'abstractions dynamiques de publication et

de souscription de l'architecture DYMES prouve que la stratégie DPSCS est, en moyenne, 28% meilleure que les autres stratégies existantes en termes d'efficience et qu'elle est capable de former des communautés dans l'ordre des millisecondes. De plus, MGDSS dépasse les stratégies existantes par un facteur de 71% à 100% en termes d'efficience pour toute densité de noeuds comparée aux autres stratégies.

Finalement, le troisième volet de cette thèse aborde la problématique de la dissémination des messages dans les environnements véhiculaires distribués. Nous proposons une nouvelle stratégie de publication et de souscription dynamique aux services appelée SocialDrive-BroadTrip qui constitue un module important dans l'architecture DYMES.

Cette stratégie assure une communication fiable et efficiente entre les voyageurs regroupés en peloton. Elle leur permet de publier des mises à jour persistantes dont le contenu est dynamique et de souscrire en utilisant des filtres dynamiques sensibles au contexte des mises à jour publiées. La correspondance entre publication et souscription est effectuée par les souscripteurs qui sont intéressés à recevoir les dernières mises à jour publiées. La propagation de ces dernières est assurée par un nouveau protocole de diffusion nommé BroadTrip. Ce dernier est basé sur la localisation et sur le codage réseau afin de réduire le nombre de retransmissions des mises à jour envoyées à travers le peloton. L'évaluation de performance de BroadTrip montre qu'il est en moyenne 12% à 38% meilleur que les autres approches existantes. De plus, nos résultats de simulation montrent que SocialDrive-BroadTrip dépasse les autres stratégies de 26% à 58% en termes d'efficience et qu'elle est plus rapide que les autres stratégies en termes du nombre de correspondances de mises à jours effectuées.

Globalement, les abstractions de communication proposées dans l'architecture DYMES peuvent être utilisées comme base de développement de n'importe quelle application véhiculaire sociale. De plus, les résultats prouvent que l'architecture DYMES améliore la qualité d'interaction entre les voyageurs tout au long de leur voyage, en leur offrant différents types de services qui leur permettent de contrer leur isolement sur la route et de communiquer en temps réel de façon anonyme, efficiente, stable et efficace. Ceci permet aussi d'assurer leur confort pendant leurs navettes quotidiennes.

ABSTRACT

Spending time in a lengthy commute is unavoidable and is considered as one of the most painful parts of the commuters' daily routine.

As ubiquitous computing is increasingly revolutionizing the way people interact and socialize, there is a pressing need to showcase vehicular social applications and services that enable proximity-based social interactions among commuters during their daily commutes. These applications aim at improving the quality of the commuters' traveling experience since they share similar congestion issues and are connected through wireless links.

However, the major challenging issues that constraint their social interactions during their highway travels are 1) their anonymity on the road that does not encourage them to share their common interest which may reveal their identities and disclose their private information to an unknown public, and 2) the heterogeneous nature of vehicular environments and the unreliable connectivity of their wireless links which may drastically impact the quality of their social interactions.

The inclusion of social networks within vehicles has attracted many researchers to devise either distributed or centralized vehicular social frameworks that support the development of vehicular applications and promote social interactions among commuters on the road. However, and to the best of our knowledge, there is a lack of hybrid vehicular social architectures. Moreover, existing architectures do not ensure a cooperation between upper service layers and the physical vehicular communication layers and are only designed to satisfy a specific kind of commuters' requirements.

In this thesis, we tackle the lack of specialized hybrid vehicular social frameworks in the literature and we propose a novel, efficient, reliable, stable, hybrid and Dynamic Messaging System (DYMES) for vehicular social networks. Our proposed messaging system enables real-time social interactions among commuters based on their common interests, without revealing their identities, while taking into account the dynamic nature of their shared information, using a set of efficient, reliable, distributed and centralized communication abstractions.

More specifically, this work is subdivided into three main aspects, each of these aspects led to scientific publications.

The first aspect consists in the identification of a meta-strategy to guide building DYMES. We propose the use of the publish/subscribe model to design novel dynamic communication abstractions that match the dynamic nature of vehicular networks, as well as the anonymous

nature of commuters' on the road. Our proposed dynamic publish/subscribe abstractions aim at breaking the commuters' social isolation by allowing them to publish dynamic contextual information and to subscribe using online context-aware message filters, without revealing their identities. We show the DYMES usage via two typical centralized and distributed vehicular social applications. Furthermore, we identify and discuss implementation issues of our proposed hybrid, distributed and centralized publish/subscribe abstractions which guide the building of our DYMES architecture.

The second aspect introduces a set of novel, hybrid, efficient, reliable, stable and dynamic publish/subscribe abstractions that constitute an important building block of our hybrid DYMES architecture.

Our first proposed abstraction is a Dynamic Publish/Subscribe Clustering Strategy (DPSCS) that tackles the problems of the commuters' anonymity and the intermittent nature of their wireless links which limit their social communication during their road trips. DPSCS provides the opportunity for each commuter to build a stable, self-updated, efficient and reliable community based on its own interests, without disclosing its personal information and without flooding the network. Using DPSCS, each commuter publishes a single persistent and dynamic publication in a predetermined geographical range, leaving the dynamic matching process to subscribers. DPSCS relies on the two Shot Stable Routing Service (2S-SRS) that disseminates the commuter's publication using stable and farther nodes.

Our second proposed abstraction is a Mobile Gateway Discovery/Selection Strategy (MGDSS) that tackles the problems pertaining to mobile gateway discovery and selection in hybrid dense vehicular environments and that lead to other related challenges such as extensive bandwidth consumption and high message overhead.

MGDSS is based on the outcome of DPSCS and allows commuters clustered in interest-based communities to efficiently and reliably discover their neighborhood, using an efficient and reliable broadcasting protocol called CoCo (Context-aware Coding). This protocol uses location and network coding in order to reduce the number of message retransmissions throughout the network, during the discovery process. A minimum number of mobile gateways is then selected by each commuter, upon the end of the discovery process, in order to simultaneously send the commuters' dynamic publications and subscriptions to the infrastructure using stable links. An online matching strategy is executed at the infrastructure and aims at updating the commuters' subscriptions and at sending positive matches to the corresponding subscribers.

More specifically, DPSCS outperforms the next best comparable approach by 28% and succeeds to build social communities in the order of milliseconds. Furthermore, MGDSS shows

an improvement of 71% to 100% over existing discovery strategies for any node density.

Finally, the third aspect introduces a new dynamic publish/subscribe broadcasting abstraction called SocialDrive-BroadTrip that also constitutes an important building block of our hybrid DYMES architecture. SocialDrive-BroadTrip tackles the problem of updates dissemination in distributed vehicular environments. It aims at enabling real-time social interactions among commuters clustered in platoons. It allows them to publish dynamic and persistent updates and to subscribe using online context-aware update filters. Matching is executed by subscribers who are interested to receive the newest updates. The dissemination of persistent publications (updates) throughout the network is performed using a novel, efficient and reliable broadcasting protocol called BroadTrip. It leverages on location information and network coding in order to reduce the number of retransmissions needed to propagate the published persistent updates in platoons. The proposed abstraction is evaluated analytically and through simulations. We first evaluate the performance of BroadTrip protocol. The results show that it outperforms the next best comparable approach by 12% to 38% depending on settings. We then use BroadTrip as an underlying protocol in the DYMES architecture and evaluate the performance of SocialDrive-BroadTrip in a vehicular dense environment. The results show that our proposed strategy outperforms other existing approaches by 26% to 58% depending on settings.

Globally, the results prove that DYMES is the best suitable messaging system to improve the quality of real-time social interactions among commuters in hybrid and dense vehicular environments, that it is able to ensure efficiency and comfort during their daily commutes and that its dynamic, efficient and reliable communication abstractions can be used to build any kind of vehicular social applications.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS AND ABBREVIATIONS

ABBA	Area-Based Beaconless Algorithm
BS	Base Station
BTS	Base Station Transceiver
CAPS	Context-Aware Publish/Subscribe scheme
CBS	Counter-based Broadcasting Scheme
CH	Cluster Head
CMGM	Clustering based on adaptive Mobile Gateway Management
CoCo	Context-aware Coding
CPS	Context-aware Publish/Subscribe
DaS	Drive and Share
DPSS	Dynamic Publish/Subscribe Service
DPSCS	Dynamic Publish/Subscribe Clustering Strategy
DSRC	Dedicated Short Range Communication
DYMES	Dynamic Messaging System
FMBP	Fast Multi-Broadcast Protocol
ITS	Intelligent Transportation Systems
LBR	Lifetime-Based Routing
LET	Link Expiration Time
LMA	Location-based Multicast Addressing
LPSS	Location-based Publish/Subscribe Service
LS	Location Service
MANETs	Mobile Ad Hoc Networks
MCS	Message Centric Strategy
MGDM	Mobile Gateway Discovery Mechanism
MGDSS	Mobile Gateway Discovery/Selection Strategy
MGSM	Mobile Gateway Selection Mechanism
OFP	Optimized Flooding Protocol
OSNs	Online Social Networks
PAMPA	Power-Aware Message Propagation Algorithm
POI	Point Of Interest
Pub/Sub	Publish/Subscribe
RET	Route Expiration Time
6SB	Six-Shot Broadcast

SMS	Scoped Multicasting Service
SOA	Service Oriented Architecture
SOR	Social On the Road
STEAM	Scalable Timed Events And Mobility
TTL	Time To Live
2S-SRS	Two-Shot Stable Routing Service
VANETs	Vehicular Ad Hoc Networks
VCSs	Vehicular Communication Systems
VSNs	Vehicular Social Networks
V2I	Vehicle to Infrastructure
V2V	Vehicle to Vehicle
WAVE	Wireless Access in Vehicular Environment

CHAPTER 1 INTRODUCTION

Vehicular Communication Systems (VCSs) have recently gained growing interest from the research community fueled by the urgent need of reducing traffic congestion as well as improving the driving comfort during lengthy commutes. In the meantime, several automotive industries such as Volvo, General motors, Ford and Audi have moved toward the vision of connected vehicles. They have recently realized the potential of integrating context-based social features to the existing vehicular navigation systems and are ramping up efforts to further develop and facilitate the deployment of fully connected vehicles. However, the deployment is still far removed from this ideal and highly depends on the attractive services that should respond to the online commuters' requirements on the road.

In fact, the drudgery of daily long commutes is unavoidable. It is considered as one of the awful moments of urban life since it negatively impacts the drivers' well-being by increasing their stress levels and lowering the quality of their traveling experience during rush hours. According to the navigation and mapping company *TomTom*, 100 extra hours per year are wasted by an average commuter during the worst evening rush hour congestion. Unfortunately, this wasted time could have been used as an opportunity to create a fruitful break from the daily engagements and responsibilities.

Among traditional vehicular applications are collision warning, security distance warning, driver assistance, cooperative driving, cruise control, road information dissemination, automatic parking and automated driving. Most of these existing safety applications rely on GPS navigation systems and on contextual information in order to provide local traffic information to the drivers during their daily long commutes. However, these applications do not satisfy the shared real-time commuters' requirements during their road trips. In fact, commuters stuck in traffic face similar congestion issues, they share different kinds of online road trips needs and preferences (Maaroufi and Pierre, 2014), but they lack real-time social interactions among each other in order to control the current state of their congested commutes. Their need of online social interaction is constrained by their social isolation on the road and the lack of scalable and social trust mechanisms (Luan et al., 2015a) that facilitate their social communication while preserving the anonymity of their identities.

Nevertheless, discovering the commuters' interests on the road and sharing dynamic contextual updates in a highly dynamic and unreliable vehicular environment brings up many-to-many message dissemination patterns. This is due to the commuter's anonymity that highly exacerbates the risk of message collisions and broadcast storms (Wisitpongphan et al., 2007;

Ni et al., 1999) since commuters randomly broadcast their instantaneous updates without relying on a neighborhood detection mechanism. Furthermore, the commuters' intermittent connectivity makes them lack great and momentaneous opportunities that allow them to identify their best peers which match their common needs and interests. In fact, the commuters' urgent need to socialize requires the creation of proximity-aware vehicular communities based on their shared interests that are called *Vehicular Social Networks* (VSNs) (Maaroufi and Pierre, 2014; Smaldone et al., 2008). The latter will facilitate their online and physical interaction among each other and will give birth to a new kind of hybrid context-based vehicular applications and services that are human-centric.

Context-aware social services are the building blocks for any messaging system (Eugster et al., 2009; Jayaram et al., 2013; Holzer et al., 2013). Their efficiency depends mostly on the performance of the underlying context-based broadcasting and clustering protocols. Unfortunately, and in addition to the aforementioned challenges, there are several technical issues that still hinder the development of a fully deployable vehicular social system, such as the scarcity of the radio resources in hybrid vehicular networks and the heterogeneous nature of vehicular ad hoc networks.

In this thesis, we focus upon vehicular communication abstractions that constitute the main building block of hybrid vehicular social systems. We aim at promoting their proliferation by providing DYMES, a Dynamic Messaging System that supports the development of vehicular social applications.

This chapter presents the basic concepts related to vehicular ad hoc networks, vehicular social networks and context-based publish/subscribe abstractions. It gives a broad review of the subject and problem statements. Detailed description of existing solutions and proposals are deferred to the next chapter. The present chapter describes the specific challenges tackled in this thesis. Then, it presents the research objectives of the thesis, the methodology plan used to achieve them and the main original contributions of the thesis. This introductory chapter ends by presenting a detailed outline of the remaining chapters.

1.1 Concepts

1.1.1 Vehicular Ad hoc Networks (VANETs)

Vehicular Ad Hoc Networks (VANETs) are a special case of Mobile Ad Hoc Networks (MANETs). They inherit the MANETs' characteristics in terms of the nodes' movements and their self-organization. However, and contrary to Mobile Ad Hoc Networks, VANETs do not face energy consumption issues, due to the presence of an alternator device that converts

mechanical energy of a vehicle to electrical rechargeable energy. The main differentiating features of VANETs and MANETs are the high speeds of mobile nodes in highway scenarios and their highly dynamic nature that is limited by predefined roads and driving rules.

In VANETs, vehicles are equipped with wireless communication capabilities that allow them to connect together and to communicate with each other using distributed communications, called *Vehicle To Vehicle* (V2V). They can also communicate indirectly through a *Base Station* (BS), such communication is called *Vehicle to Infrastructure* (V2I) (Holzer et al., 2010). The Wireless Access in Vehicular Environment (WAVE) and IEEE 802.11p are at the core of the *Dedicated Short Range Communication* (DSRC) standards that allow both V2V and V2I communication. Three types of networks architectures can be built using these technologies, as illustrated in Figure 1.1.

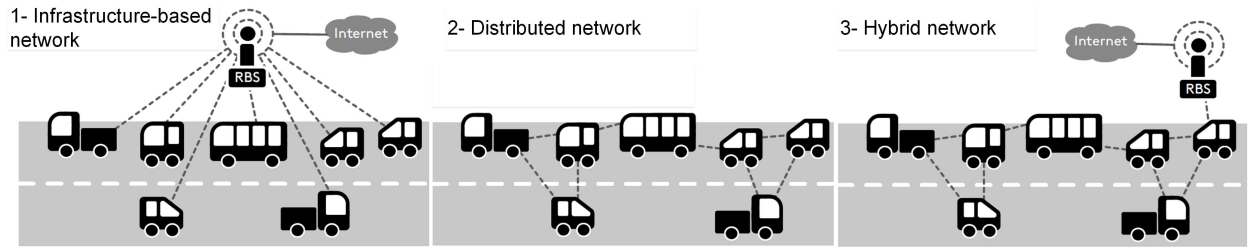

Figure 1.1 VANET architectures

The infrastructure-based vehicular network is depicted in Figure 1.1.1. Such network can currently be found on the road with drivers accessing the Internet through their smart phones. The distributed vehicular ad hoc network V2V is depicted in Figure 1.1.2 and the hybrid vehicular network that combines both distributed and centralized architectures is depicted in Figure 1.1.3.

1.1.2 Overview of VANETs Applications

Vehicular applications have recently emerged with the widespread diffusion of smart mobile devices with network capabilities and access to user location. These applications aim at improving the users' traveling experience by providing real time traffic information and also aim at ensuring comfortable and safe trips to the travelers. There are two main classes of VANET applications: safety and non-safety applications (Kaveh, 2012). The first class (i.e., safety applications) aims at increasing the drivers' safety awareness to accidents and emergency conditions located in close proximity. Safety applications include collision and security distance warning, driver assistance, cooperative driving, cruise control, road infor-

mation dissemination, automatic parking and automated driving (Holzer et al., 2010). These applications are used to help drivers avoid collisions and coordinate among themselves at highway entries and at intersections by broadcasting warning messages over the considered network. Such applications aim at saving lives by reducing the number of car accidents and the associated damages.

Non-safety applications aim at providing comfortable and entertaining services to the drivers. The latter range from real-time interactive games, weather information, restaurant locations, mobile e-commerce, music downloads and content delivery.

1.1.3 Vehicular Social Networks (VSNs)

Vehicular Ad hoc Networks (VANETs) can be considered as a vehicular social network for cars, since commuters can share contextual information in their physical neighborhood through V2V and V2I wireless capabilities (Vegni and Loscri, 2015).

Human behavior and social features largely influence VANETs which lead to Vehicular Social Networks (VSNs). A VSN is a social network of users who share a daily roadway commute (Smaldone et al., 2008). It provides the opportunity to create vehicular communities in which users share similar interests and are able to interact with each other using heterogeneous devices in order to enhance the driving experience.

In fact, people spend more and more time every day between home and work and use the same routes at specific time intervals (Litman, 2014; Force, 2012). Despite their anonymity on the road, they are eager to socialize and share their contextual updates as well as to develop momentaneous relationships inside vehicular communities based on their individual interests. Infotainment, entertainment, utility and emergency usages are one of many reasons that stay behind the creation of VSNs along the route.

Infotainment and entertainment usages aim for example at occupying commuters on long trips and who share common interests in playing the same online game, it may also concern people who discuss recent sport news or social and political issues.

Emergency usage aims at facilitating the driving experience in critical situations like assisting drivers in avoiding future collisions along the road through disseminating road information to vehicles in the vicinity of accidents.

Finally, utility usage aims at providing useful advices and ideas about local places already visited (restaurants, hotels etc ...), as well as to announce local events along highways which translate real customer experiences. These different usages will encourage developers to devise Location-Based Marketing (LBM) applications (Mediative, 2013) able to manage the

huge amount of the dynamic contextual data that is shared based on the consumers' activity. This kind of application will certainly unlock the future of Intelligent Transportation Systems (ITSs) (Maaroufi and Pierre, 2014).

The main difference between Online Social Networks (OSNs), that include Facebook, Twitter and LinkedIn and VSNs lies in the fact that OSNs form a virtual image of physical communities in which users are connected through friendship and professional relationships, while commuters on the road are anonymous to each other and are only connected using wireless links.

1.1.4 Broadcasting in Vehicular Networks

A broadcast is the dissemination of a message to all nodes in the network. Broadcasting is a very useful operation in vehicular networks and aims at transmitting data traffic information to all nodes in a determined geographical area for different purposes. The main problem of broadcasting in VANETs is flooding the network with redundant messages and updates, since vehicles move at high speeds and the content of the contextual information shared among users changes rapidly. This generates a huge number of collisions and also causes a huge number of retransmissions. The solution is to select forwarding node candidates able to retransmit the message to disseminate it in the network. In order to evaluate the performance of a given broadcast protocol, we focus on two measures:

- Reliability is defined by the delivery rate, i.e., the number of nodes that deliver a broadcast message m divided by the number of nodes in the network;
- Efficiency is defined by the message load, i.e., the number of retransmissions of a message m generated by one or multiple broadcasts. The higher the message load, the lower the efficiency.

Based on these measures, we can say that the broadcast problem is to maximize both reliability and efficiency.

1.1.5 Context-based Publish/Subscribe Services

Applications specifically adapted for vehicular networks are special cases of context-aware services. Location-based applications, such as GPS navigation systems, are examples of such services. Location is one of many environmental variables that might influence the behavior of an application. The notion of context encompasses these variables in the broad sense. In

traditional applications, the context information is not accessible to the application code. However, a context-aware application is a piece of software that can react to its context (Holzer et al., 2010).

Context-based applications refer to two types of context: the self-centered context and the social context. The self-centered context represents the commuter's context that is issued through its own vehicle, such as its location, its velocity and its direction. The social context represents the contextual information received from the neighboring vehicles, such as their location and acceleration during a specific time interval. The physical social contextual information (location) defines the social proximity of commuters from one another, whereas the temporal social contextual information (time) represents the life-time of the messages received from the neighboring vehicles.

The publish/subscribe paradigm (Eugster et al., 2003) has recently emerged as a promising solution for mobile computing due to its strong communication property that provides users with the ability to communicate with each other in an anonymous manner. Figure 1.2 illustrates the operating of such a paradigm. In fact, producers (publishers, sources or senders) post events (publications), whereas, consumers (subscribers or receivers) advertise their interests in certain published events and specify their desired content or topic through the creation of a subscription. The publish/subscribe service matches publications with subscriptions and notifies subscribers of matches that satisfy their needs.

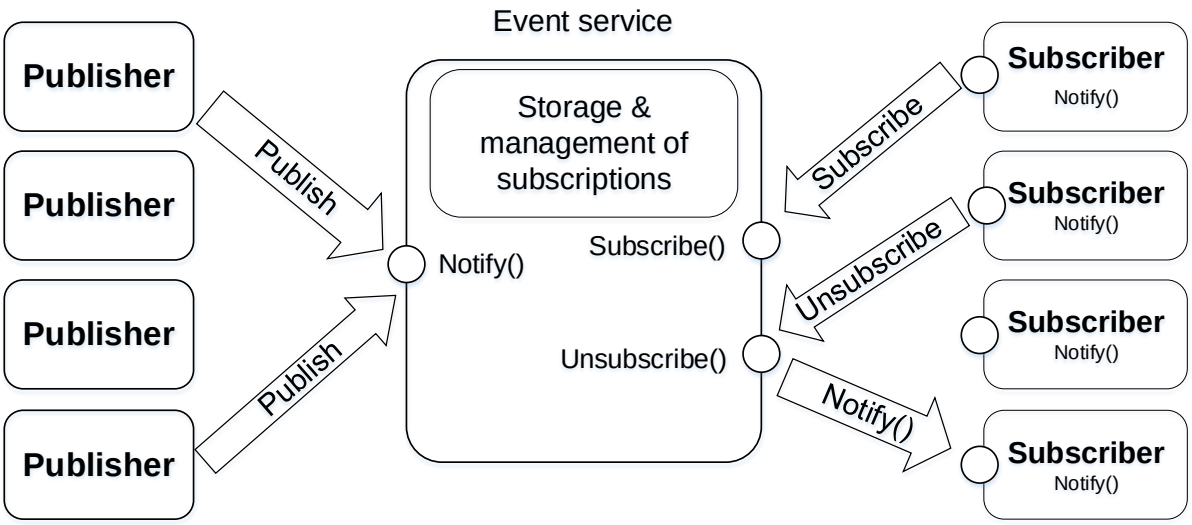

Figure 1.2 A simple example of the operating of the publish/subscribe model

In Context-based publish/subscribe services, publications and subscriptions are bound to a certain geographical range around the publisher and the subscriber respectively. Publications can be persistent to allow subscribers, located in that geographical space, to receive publications when they meet the matching condition even if these criteria were not met at the publication time (Holzer et al., 2010).

Among typical mobile location-aware publish/subscribe applications are *Foursquare* and dating applications like *MeetMoi*. In such applications, subscriptions depend on the user's GPS coordinates. Subscriptions change as long as the devices move. Notifications are sent to mobile users once their profiles match a desired surrounding profile.

Having defined the main concepts related to context-based communication abstractions and vehicular ad hoc networks, we tackle the major challenges that defy the use of these abstractions in vehicular social networks.

1.2 Problem Statement

Research highlights the fact that commuters share different and common needs during their daily commutes. However, they lack real-time social interaction among each other due to their anonymity on the road. The latter hugely impacts the network bandwidth in infrastructure-based vehicular social networks, especially when all vehicles have access to a Base Station Transceiver (BST), at the same time, in order to request different kinds of services. Moreover, the vehicular environment is highly dynamic and requires a short and strict communication delay. Furthermore, the current vehicular applications are only targeting a special kind of safety services and do not pay the required attention to the commuters'-oriented services that are mainly based on their online social interests.

We argue that vehicular applications require a specific type of communication abstraction which takes into account the highly dynamic nature of the shared information as well as the contextual information of senders and receivers.

Context-based communication abstractions form a foundation for building messaging systems (Eugster et al., 2009; Jayaram et al., 2013; Holzer et al., 2013). Their performance greatly depend on the underlying communication protocols that aim at disseminating contextual information efficiently and reliably, throughout the network.

As it will be reviewed in chapter 2, the design of dynamic context-aware communication abstractions specifically adapted to the highly dynamic nature of VSNs is a challenging endeavor. This is due to the following reasons.

- *Dynamic contextual updates*: VSNs context-awareness is tightly coupled with communication and mobility. In fact, the dynamic and rapidly changing nature of the location, the velocity and the driving habits of commuters in the road increases the dissemination rate of contextual updates and decreases both the efficiency and the reliability of context-aware communication abstractions by provoking message collisions that cause broadcast storms (Wisitpongphan et al., 2007; Ni et al., 1999). It also produces accuracy problems to the users that may receive outdated disseminated messages, which negatively impacts the information that they may share, later, over the entire network.

  The latter may drastically impact the quality of vehicular social applications in a way that a user may decide to join a vehicular social community based on an outdated sender's message and may be left isolated since he was not able to receive an accurate information.

- *Lack of specialized social abstractions for VSNs*: We recognize that vehicular social networks are still considered as a fertile research area. In fact, several research works (Smaldone et al., 2008; Han et al., 2010; Hu et al., 2011; Luan et al., 2015a) have addressed and focused on single aspects of vehicular social networking area that only include the design of services and software platforms for the development of a specific kind of vehicular social applications. In fact, Meier and Cahill (Meier and Cahill, 2010, 2002a) proposed STEAM, a Scalable Timed Events And Mobility publish/subscribe system that targets different kinds of location-based applications scenarios. However, they did not provide any detail about the implementation of the STEAM's communication protocol.

  Unfortunately, existing works do not support the full design and the development of comprehensive architectures that ensure a cooperation between the upper service layers and the physical vehicular communication layers. That is, the proposed context-aware vehicular social services should rely on scalable, efficient and reliable communication protocols, that perform under the heterogeneous vehicular conditions such as high mobility, limited bandwidth, and congested traffic in order to fill the lack of context-aware vehicular social abstractions and to push next generation vehicular social applications.

- *Bandwidth consumption and message dissemination*: V2I communications are generally constrained by the limited bandwidth resources. In fact, the dissemination of the commuters' dynamic contextual information in hybrid vehicular environments requires the communication among commuters and the infrastructure using a limited bandwidth. However, the scarcity of the network access radio resources requires nodes to discover

and to select a minimum number of suitable gateways able to link them with the BTS at low costs.

Existing mobile gateway discovery and selection mechanisms increase the message signaling overheads and require frequent changes in the routing tables of vehicles. In fact, a recent work (Benslimane et al., 2011a) proposed an architecture in which vehicles are dynamically clustered and a minimum number of mobile gateways is discovered and selected. These mobile gateways link VANET to the 3G wireless network. However, the proposed mechanisms, i.e. discovery and selection mechanisms, are centralized in a way that once a vehicle determines the adequate gateway, it floods the network in order to announce the presence of the mobile gateway in its neighborhood. The latter generates a huge amount of broadcast messages that flood the network and cause message collisions, which may leave some vehicles isolated during their congested commutes.

This is due to the vehicles' intermittent wireless connectivity and the unpredictable vehicular network topology that negatively impact the reliability of the hybrid vehicular communication by delaying the delivery time and increasing the packets loss especially in highly dense vehicular environments.

- *Commuters' anonymity*: contrarily to traditional online social networks that include *Facebook*, *LinkedIn* and *Twitter* and in which users are connected using friendships and professional relationships, commuters on the road are unknown and anonymous to each other. They are only connected through wireless links and their common road trips' needs and preferences. Moreover, commuters' anonymity makes them hesitant about disclosing their private information to the public which makes them lack real-time social communication among each other in order to enhance their driving experience during their lengthy commutes. Recent research works (Luan et al., 2015a,b) highlighted the need of specialized communication abstractions that help commuters break their social isolation on the road and that allow them to identify the adequate peers that match their similar interests and with whom they can share their personal contextual information in a private manner.

1.3 Research Objectives

In the light of the aforementioned challenges, this thesis aims at exploring how to benefit from the dynamic nature of the contextual shared information, the limited radio resources and the communication pattern predictions in order to design an efficient and reliable online and hybrid Dynamic Messaging System (DYMES) in which different distributed and centralized

communication abstractions can be used depending on the vehicular environment. DYMES allows to 1) facilitate the creation of online vehicular social communities and to 2) promote social interactions among commuters and the infrastructure in highly dense environments.

More specifically, this thesis aims at:

1. Designing adequate dynamic communication abstractions for VSNs.
2. Designing and implementing an efficient and reliable context-aware broadcasting algorithm specifically adapted for vehicles transiting in platoons.
3. Designing and implementing an online social communication abstraction for distributed vehicular social applications.
4. Designing and implementing a specialized dynamic clustering strategy based on a stable and efficient location-based routing service for VSNs.
5. Designing and implementing dynamic mobile gateway discovery/selection mechanisms based on efficient and reliable routing services.

1.4 Methodological Approach

Our research objectives are achieved through an ongoing literature review in all stages of this thesis. After a first complete literature review, scientific research papers are periodically seek in order to keep an up to date perspective on the state of the art.

The first research objective of our thesis is achieved by addressing the shortcomings identified while reviewing middleware solutions based on publish/subscribe abstractions aimed at context-aware applications. We focus on the publish/subscribe model because it matches the commuters requirements on the road, in a way that it allows them to communicate with each other in an anonymous manner and without revealing their identities. We then propose a dynamic publish/subscribe abstraction through DYMES specifically intended for VANETs.

We show how DYMES can be used in order to implement two publish/subscribe vehicular applications and we introduce the DYMES' layered architecture. Then, we detail the different DYMES' building blocks and discuss our implementation choices for each block that consists of a dynamic publish/subscribe abstraction based on specialized communication protocols that depend on underlying vehicular network topologies, i.e., distributed, centralized and hybrid.

Having defined the DYMES' global architecture in the first objective, we design the first building block of the DYMES' architecture in the second objective, that is a broadcasting protocol for a distributed vehicular topology (V2V).

To do so, we review existing protocols that tackle the problem of message dissemination in vehicular environments. Then, we design and implement a distributed broadcasting protocol for vehicles transiting in platoons called BroadTrip. Our proposed solution makes use of a location-aware wait-and-count mechanism called PAMPA (Power-aware Message Propagation Algorithm) combined with network coding, in order to reduce the number of retransmissions needed to broadcast messages in platoons.

The combination of PAMPA and network coding aims at reducing the message load of a single broadcast (by using PAMPA) and the message load of multiple broadcasts (by using network coding). We validate BroadTrip by analysis and simulation and we also compare it to other existing approaches.

Then, based on the outcome of the second objective, research objective 3 is achieved by extending BroadTrip and integrating it in an online social based publish/subscribe abstraction for vehicular social platoons called SocialDrive. It publishes dynamic and persistent contextual information and allows commuters to subscribe to messages using online context-based filters. SocialDrive's matching process is executed by subscribers who are interested at receiving the latest issued updates from all the members of the platoon. We validate the performance of SocialDrive by simulation and we also compare it to other similar strategies.

Having designed, implemented and evaluated SocialDrive, the next step is to tackle the fourth objective of our thesis that aims at designing another building block of the DYMES' architecture and that focuses on online social grouping strategies for vehicular social networks.

Our fourth objective is achieved as follows: We review vehicular social frameworks that aim at building vehicular social communities in vehicular social networks and we define the main challenges that stay behind the lack of specialized dynamic context-based grouping abstractions in VSNs. We then propose a novel Dynamic Publish/Subscribe Clustering Strategy called DPSCS based on a stable, efficient and reliable multicasting protocol, called 2S-SRS (Two Shot Stable Routing Service). We combine two important metrics in order to design the DPSCS's multicasting protocol (2S-SRS), namely the stability metrics and the proximity metrics. DPSCS allows commuters to create interest-based social communities efficiently and reliably while preserving the privacy of the commuters' shared information. Moreover, published and persistent messages are disseminated throughout the network using stable forwarders who are located far from the publishers and closer to two targets. We validate our proposed solution by mathematical analysis and simulation and we compare it to exiting

strategies.

Finally, based on the outcome of objective 4, research objective 5 tackles the building of another important block of the DYMES' architecture specifically adapted for hybrid and dense vehicular environments.

Objective 5 is achieved as follows: We design an efficient, reliable and stable Mobile Gateway Discovery Strategy (MGDS) based on a broadcasting protocol called CoCo (Context-aware Coding), that allows commuters clustered in communities to get knowledge and discover their neighborhood. We then design a Mobile Gateway Selection Strategy (MGSS) that allows nodes to select a minimum number of mobile gateways based on the information they gathered during the discovery strategy (MGDS-CoCo). Selected gateways link commuters to the infrastructure by sending their dynamic publications/subscriptions to the cloud where an online matching strategy is performed. Positive matches are sent back to the commuting subscribers using the selected mobile gateway which reverses the paths through which it received the commuters' subscriptions. We evaluate the performance of the proposed strategies by analysis and simulation and we compare their performance to existing similar approaches.

1.5 Contributions & Originalities

This thesis comprises five main contributions which are presented in the following.

The first original contribution brought by this thesis is a novel hybrid and dynamic messaging system (Maaroufi and Pierre, 2015a) for VSNs, called DYMES that aims at enabling and facilitating real-time social interactions among commuters during their daily and lengthy commutes using a set of efficient, dynamic, distributed and centralized communication abstractions. Our proposed system is mainly fueled by the lack of specialized vehicular social communication abstractions able to 1) break the commuters' social isolation on the road while preserving the anonymity of their identities, 2) facilitate the creation of stable and reliable proximity and interest-based vehicular social communities while ensuring efficient and reliable updates dissemination and 3) ensure a scalable communication with the infrastructure in highly dense networks while lowering the bandwidth consumption.

To the best of our knowledge, this is the first original and rich messaging system that provides one-fits-all solution which spans the whole spectrum of vehicular social networks and which can be used as a building block for the development of different kinds of vehicular social applications in order to meet the online commuters' needs on the road.

The originality of our proposed dynamic messaging system lies in the fact that it relies on hybrid dynamic publish/subscribe abstractions that enable commuters to efficiently and

reliably communicate with each other based on their interests, without disclosing their private information, while taking into account the highly dynamic nature of the shared information.

A set of other contributions is related to DYMES, an architecture that tackles different salient challenges in hybrid vehicular social networks.

The second contribution is a Dynamic Publish/Subscribe Clustering Strategy (DPSCS) (Maaroufi and Pierre, 2015a,b) that tackles the problems of the commuters' anonymity, their intermittent connectivity and their strong need of efficient and reliable real-time communication abstractions that help them identify their best peers with similar interests. DPSCS solves this problem by allowing each commuter to build a stable, self-updated, efficient and reliable community based on its own interests, while preserving the commuters' personal information and without flooding the network with signalling messages. Our proposed solution beats other existing strategies in terms of the efficiency and reliability by 28% by building interest-based communities in a very short time.

The third contribution is a Mobile Gateway Discovery/Selection Strategy (MGDSS) (Maaroufi and Pierre, 2015a) for hybrid VSNs. This strategy tackles the problems related to mobile gateway discovery and selection in hybrid dense vehicular environments and that lead to other related issues such as huge bandwidth consumption and high message overhead.

MGDSS is based on the outcome of our first contribution (DPSCS) and allows commuters clustered in interest-based social communities to individually discover their neighborhood and be up-to-date about real-time traffic conditions using an efficient and reliable context-aware broadcasting protocol. They then select a minimum number of appropriate mobile gateways in order to ensure an efficient and reliable connectivity with the infrastructure using minimal bandwidth resources. Moreover, MGDSS allows commuters to satisfy their common needs in a dense infrastructure-based vehicular environment using a Dynamic Publish/Subcribe Strategy (DPSS). The latter allows commuters to simultaneously request infrastructure-based dynamic services and receive the latest updated information that match their online needs in a timely and reliable manner.

Our proposed MGDSS strategy that is based on the DPSS service shows an improvement of 71% to 100% over existing approaches for any node density.

The fourth contribution is a distributed, efficient and reliable location-based broadcasting algorithm for vehicles transiting in platoons called BroadTrip (Holzer et al., 2011). It tackles the problem of disseminating messages in distributed vehicular environments and is specifically intended for automated driving technology where cars follow each other in platoons and exchange messages in order to coordinate their movements, inform each other of emergencies

and share relevant contextual social information.

BroadTrip combines a proximity-based mechanism to increase the efficiency of a single message broadcast, and network coding mechanism that further increases the efficiency by performing wise coding decisions without requiring knowledge about the nodes' neighborhood. BroadTrip outperforms the next best comparable approach by 12% to 38% depending on settings.

The fifth contribution of this thesis is an online publish/subscribe strategy for vehicles transiting in platoons called SocialDrive (Maaroufi and Pierre, 2014). It tackles the problem of disseminating dynamic contextual updates in highly dense and distributed platoons, through extending BroadTrip and using it as an underlying protocol in the DYMES's architecture. The latter improves the quality of social communication among commuters clustered in dense platoons and increases their driving comfort by allowing them to receive the latest issued dynamic contextual update that match their interests. SocialDrive outperforms the next best comparable approaches by 26% to 58% depending on settings.

Globally, the results prove that DYMES is the best suitable messaging system able to improve the quality of real-time interactions among commuters in hybrid and dense vehicular environments and to also help them create a positive commuting time during their highway travels which strengthen the originality of our contributions.

1.6 Organization of the Thesis

The reminder of this thesis is organized as follows:

Chapter 2 - Literature Review: This chapter discusses the existing works related to our research areas, it first presents the state-of-the-art of vehicular social architectures. Then, it shows the most relevant research contributions to mobile location-aware publish/subscribe abstractions and context-aware group communication abstractions. The chapter also enumerates existing context-aware broadcasting algorithms. Finally, this chapter presents related works on dynamic clustering and mobile gateway management algorithms.

Chapter 3 - A Dynamic Messaging Service for VANETs: This chapter describes, in detail, the general architecture of DYMES. It also shows how DYMES can be used to program two typical vehicular applications.

Chapter 4 - A Dynamic Publish/Subscribe Abstraction for Distributed VSNs: This chapter presents, in detail, the operating of a dynamic publish/subscribe strategy for distributed VSNs, called SocialDrive, that is based on a location-based broadcasting protocol for vehicles transiting in platoons called BroadTrip. This chapter also shows the obtained

results and their discussion.

Chapter 5 - Discovering the Architecture of a Dynamic Messaging System for VSNs: This chapter presents the the architecture of DYMES. It describes its operating and details the algorithms associated with the DYMES's proposed strategies. It also presents the validation of the overall solution which is performed through simulations.

Chapter 6 - Conclusion and Future Work: Finally, this chapter concludes this thesis by discussing its major contributions, its main limitations, and well as a selection of future research directions.

CHAPTER 2 LITERATURE REVIEW

The main focus of this thesis is to propose an efficient, reliable and hybrid dynamic messaging system for VSNs, in which specialized distributed and centralized communication abstractions can be used depending on the vehicular environment.

Therefore, in this chapter we investigate the main proposed systems in the literature that have been designed and that rely on hybrid context-aware communication abstractions for vehicular social networks. First, in section 2.1 we review existing vehicular social architectures. Second, in section 2.2, we review and discuss mobile location-aware publish/subscribe abstractions and context-aware group communication abstractions. Third, we review and discuss context-aware broadcasting algorithms in section 2.3. Finally, we provide a review and a discussion of dynamic clustering and mobile gateway management algorithms in hybrid vehicular networks, in section 2.4.

2.1 Vehicular Social Architectures

Hereafter, we classify vehicular social architectures into two categories, the centralized architectures and the distributed architectures.

2.1.1 Centralized VSNs

A seminal work (Smaldone et al., 2008; Han et al., 2010) has first introduced the concept of vehicular social networks through a client/server framework called RoadSpeak. It aims at building a specific type of vehicular social communities in order to facilitate the communication between commuters physically present at the same location. This framework allows a driver to automatically join a VSN and to communicate with other commuters trough voice chat messages.

Another piece of work (Fei et al., 2011) proposed an effective and efficient vehicular social system based on social centrality as well as on physical network operational mechanisms.

Centrality is an important concept that has been widely used in social networks to highlight the relation between individuals and communities. According to the graph theory and network analysis, centrality is a quantitative measure of the importance of a vertex within a graph, i.e., how influential a person is within a social network (Freeman, 1978).

There exist three centrality measures: degree centrality, closeness centrality and betweenness

centrality. Degree centrality deals with the central actors that are the most active in the network graph. Closeness centrality is based on closeness or distance, i.e., the central actor is able to interact as quickly as possible with all others since it is close to them. In betweenness centrality, most central nodes are the ones that are on many shortest paths of any other nodes.

Fei et al. (2011) have considered degree centrality as an important aspect that has a great impact on dynamic bandwidth allocation in terms of average queuing delay and throughput. In fact, they used coaches as the best message carriers and/or consumers since it usually has more passengers which lead to a prominent social role and possess more resources than a vehicle.

More in line with the idea of designing robust vehicular social frameworks, Lequerica et al. (Lequerica et al., 2010) have analyzed the challenges faced when integrating social networks in vehicular scenarios. The authors proposed *Drive and Share* (DaS) architecture that provides social services in vehicles based on IP multimedia subsystem and Machine to Machine capabilities. DaS also uses cloud computing systems to interconnect mobile users.

Another interesting work similar to RoadSpeak (Smaldone et al., 2008) is proposed in (Sha et al., 2013). In this paper, a shared driving experience in form of voice tweets and drivers' preferences are integrated in a social vehicular navigation system allowing users to take benefits of tweet digest in order to calculate a personalized route.

2.1.2 Distributed VSNs

The main characteristic of the distributed VSNs is the absence of roadside infrastructures. Vehicles communicate independently and collaborate directly with each others in an ad hoc manner. Therefore, drivers have to store and disseminate contextual and social information until the final destination is found.

Verse (Luan et al., 2015b) is a vehicular social software that facilitates social communications among vehicles on highways. It is based on a friend recommendation function to identify potential social friends with both common interests and scalable wireless connections. The use of Verse is only restricted to distributed vehicular communications and single-hop V2V connections. Moreover, there is no details about the vehicular communication protocol used to link vehicle to each other and that should reduce the cost of messages and updates exchanged through the distributed considered network.

Social on the Road (SOR) (Luan et al., 2015a) is a decentralized system that enables social messaging and information exchange among vehicles present in physical proximity. The

social communication and exchange of messages are conducted through the onboard wireless transceivers among vehicles. SOR relies on an interest matching block that exploits potential social friends with shared interests. It also relies on a connection time prediction block that recommends users with relatively longer connection times for social interactions. It is worth noting that the interest matching block on which is based SOR requires users to publicly indicate their social preferences for evaluating the similarity of interests. This may disclose sensitive personal information if inappropriately managed, thus violating important user privacy.

SOR employed a privacy preserving mechanism that avoids the direct transmission of the users' social interests and that allows them to convert their social interests to binary variables before sending them to other neighboring vehicles. The interest matching block compares the similarity of social interests based on the social interest profiles posted by each user and makes recommendations accordingly.

In an approach by Abbani et al. (2011) introducing a framework for building and managing vehicular social networks; the authors used trust conditions to allow users to join social groups and control the interactions between members. Furthermore, they clarified the role of each component of the proposed system and proved through mathematical analysis and simulation results that the dynamic trust capabilities help reduce the influence of malicious behaviors in the social networks. Nevertheless, the authors have just focused on the trusting strategies and did not pay the required attention to the communication protocols used to disseminate information and updates throughout the network.

Hu et al. (2012) proposed a software platform called VSSA for transportation applications. VSSA fully supports Service Oriented Architecture (SOA) based mobile applications working independently without roadside infrastructures in popular mobile platforms such as Android, and in an ad-hoc manner with low resource overhead. It aims at facilitating the developer's task and helps build diverse and useful mobile applications for the transportation use. Moreover, it supports dynamic web service collaboration at run-time to enable people in transportation environment to effectively collaborate with others in their vicinity through conventional ways of social networking which they are already familiar with, so as to improve transportation efficiency.

More in line with the idea of software platforms for VSNs. S-Aframe (Hu et al., 2011) is a semantic based multi-agent framework for vehicular social networks. It is based on a multilayer architecture built on existing operating systems and implemented using Java. Likewise to VSSA, S-Aframe also supports easy and effective application development for vehicular social networks. Moreover, S-Aframe supports dynamically changing environment.

In fact, mobile agents created by applications can migrate automatically and dynamically around vehicular social networks and use different application services provided by resident agents on local nodes to accomplish specific application tasks in a coordinated manner and at the same time.

An intervehicular communication architecture is presented in (Palazzi et al., 2010, 2007). This architecture is based on a Fast Multi-Broadcast Protocol (FMBP) and is devised for gaming and safety applications. It aims at propagating and delivering messages as quick as possible to all engaged vehicles in a considered car platoon while guarantying a less-resource consuming over the distributed considered network.

Discussion

From the regarded vehicular social architectures, we can argue that there are several reasons behind the lack of successful vehicular social systems in vehicular environments: first of all, centralized architectures like RoadSpeak (Smaldone et al., 2008; Sha et al., 2013) rely on the use of a server that acts as a main component of the proposed social application. However, the dynamic topology of the vehicular environment and the frequent cut of connectivity status in VANETs do not encourage the use of a stable server that ensures the major tasks of such social application. Although RoadSpeak may be an inspiring social application to the existing vehicular applications, it just provides a voice chat service which may not satisfy the diverse needs of commuters in vehicular social networks.

Second, the centrality metric is an interesting social factor that has been used in the design of a centralized vehicular social system (Fei et al., 2011). Although the authors have used it in order to select the best relay, there exists a probability that relaying is approximately executed by the same coaches in a highly dense network and there is still a need to consider the negative social behavior of drivers on the road such as selfishness, which can stop or slow the forwarding process in a hybrid and collaborative vehicular network. Moreover, the social behavior of selected actors and the highly dynamic mobility patterns may introduce several updates and transformations in the community structure which may lead to some damages during the relaying process.

Third, SOR (Luan et al., 2015a) relies on a centralized similarity mechanism that takes into account the global users' social interest profiles physically present in the same location. The latter may consume much bandwidth resources and lead to a high communication complexity in a dense environment, since it requires knowledge about the social interest of each user in order to recommend those with long-lasting connection time for social communications. Moreover, SOR does not allow users to share their current dynamic social interests since they

acquire their social interest profiles from existing online social networks such as *Facebook* or from an existing static blog information. The latter requires them to be connected to the internet which does not respect the SOR's distributed architecture.

Furthermore, SOR is still in its initial step toward a distributed vehicular social system. In fact, the SOR's vehicular context-based communication abstraction that helps users post their social interest profile is still unknown.

We contribute to the existing body of research by providing DYMES, a hybrid and dynamic context-aware messaging architecture that relies on scalable, efficient and reliable context-aware publish/subscribe abstractions and that aims at enabling real-time social interactions among commuters in hybrid vehicular environments (Chapter 3 and 5).

2.2 Location-based Publish/Subscribe Abstractions

Location and velocity are the main contextual information that are relevant to vehicular ad hoc applications. We advocate that location-aware communication abstractions are a fundamental need of vehicular social systems and that it should be provided as a dynamic context-based publish/subscribe service.

Hereafter, we provide a literature review of location-based publish/subscribe abstractions aimed at context-aware applications.

Eugster et al. (2005) introduced the concept of Location-based Publish/Subscribe Service (LPSS) that allows mobile ad hoc applications to transparently and anonymously communicate with each other using a subscription and a publication, based on their location. With LPSS, publications as well as subscriptions are distributed and bound to a geographical range around the publisher and the subscriber respectively. These ranges define the publication space and the subscription space. Publications are persistent, i.e. the published event is distributed to all interested subscribers located or entering the publication space before the event is unpublished (Eugster et al., 2005). A persistent publication is unpublished by its producer or by the service once its determined period known as *Time-To-Live* (TTL) is elapsed. A match occurs when a publication/subscription couple meets two conditions:

1. ***Content match***: the published event contains at least all attributes specified in the subscribed event.

2. ***Location match***: both subscriber and publisher are located in the intersection of both spaces (see Figure 2.1).

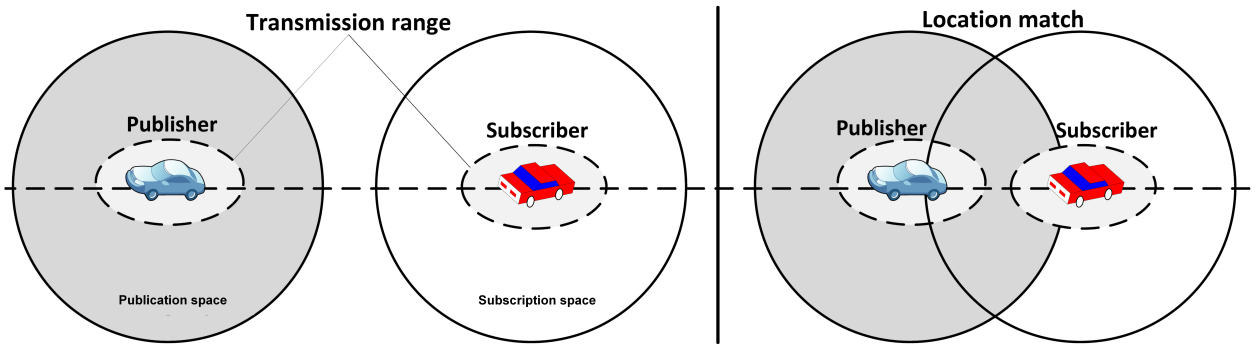

Figure 2.1 Location-based publish/subscribe matching

LPSS is implemented in a client/server platform for developing and testing mobile ad hoc applications (Eugster et al., 2009) called PERVAHO/LPSS. The LPSS client is responsible for sending persistent publications and subscriptions to the LPSS server that executes the matching operation and sends the matches (publication-subscription) to the corresponding subscriber clients. A positive match occurs when both the subscriber and the publisher are located in the intersection of the publication and the subscription spaces.

Nevertheless, PERVAHO does not publish a dynamic context. Moreover, the location of the sender is only used for matching and is not considered as dynamic contextual information for the receiver. Besides, PERVAHO/LPSS does not address the problem of gateway selection; each client sends its message/query to the infrastructure which may lead to a huge bandwidth consumption, overload the LPSS server and more importantly, disrupt the matching operation. This may also impact the users' satisfaction who are waiting to receive the matching pair (publication-subscription).

More in line with the extensions of the publish/subscribe paradigm that accommodate context-aware social applications; CPS is a Context-aware Publish/Subscribe service (Frey and Roman, 2007) in which both publications and subscriptions have a contextual component used for matching and both determine an area of relevance. This area must have a non-empty intersection for the matching to occur. Furthermore, publications and subscriptions can require the subscriber respectively the publisher to be located in a certain area, called domain, in order for the match to occur. However and contrary to our proposed messaging system, the matching areas are fixed in CPS and there is no support for publishing dynamic messages.

Another Context-Aware Publish/Subscribe scheme (CAPS) is proposed in (Cugola et al., 2009; Cugola and Migliavacca, 2008). In CAPS context is not encoded into the event, but

treated separately as a first class object very much like in LPSS (Eugster et al., 2005). The implementation of CAPS proposes three message routing techniques, publisher forwarding, subscriber forwarding and context-forwarding. The idea of the last approach is to broadcast contextual information in the network to build contextual routes which will be used to route messages and subscriptions.

Another line of work is proposed by Leontiadis et al. (Leontiadis, 2007; Leontiadis et al., 2009; Leontiadis and Mascolo, 2007a). They present a publish subscribe service for VANETs called VAPS. The idea is that a publication can be assigned to a publication *Area*, and a topic. Topics can contain a Point Of Interest (POI), which can be a geographical location. Subscribers subscribe with a similar topic. POIs can include routes, points, city, and road segments. Location is only used to restrain publication to a geographical region. VAPS considers a hybrid VANET for its implementation. However, VAPS does not allow to publish dynamic messages and subscription filters are not dynamic.

Message Centric Strategy (MCS) is one of three context-aware group communication strategies that are proposed in (Holzer et al., 2013). It is mainly based on the publish/subscribe model and aims at multicasting messages in a predefined geographical range around the senders. Nodes that are interested to join the multicast groups create a query and perform the location and the content matching processes, i.e., each node verifies if the content of the message matches with its query and also if it is located inside the intersection of the sender's message space and the receiver's query space. However, the proposed strategy does not allow to publish dynamic context. Moreover, it periodically broadcasts newly created messages that carry updated location information, which decreases the efficiency and the reliability of the proposed strategy, since these messages are forwarded throughout the network using the counter-based broadcast scheme (CBS) that is less reliable than a location-aware broadcast scheme (Garbinato et al., 2008).

Scalable Timed Events And Mobility middleware (STEAM) (Meier and Cahill, 2002b,a, 2010; Hughes et al., 2004) is an event-based middleware service that incorporates location sensitivity to the publish/subscribe paradigm. It is another message centric strategy (Holzer et al., 2013) that aims at propagating nonpersistent messages in a predefined area. It introduces a proximity filter in addition to the topic and content filters. The proximity filter supports the event-based communication models and enables the discovery of peers sharing the same geographical area and willing to interact with each other's based on functional and geographical aspects. Events are propagated within a restricted geographical range around the publisher, however, messages are not persistent, subscribers do not use dynamic filters and the underlying communication protocol is treated as a blackbox.

Research work on social networks has recently focused on publish/subscribe frameworks for delay tolerant networks (Costa et al., 2008; Zhao and Wu, 2010). SocialCast (Costa et al., 2008) is an interest-aware routing framework for Pub/Sub model and it relies on utility function to select the best message carriers in a social community. It assumes a previous knowledge of all the social interaction between mobile users and the connectivity structures before starting the dissemination process. Conversely to DYMES, SocialCast does not support the forwarding of recent and updated messages.

Zhao and Wu (2010) proposed a socially-aware Publish/Subscribe system for wireless communication carried by humans in Human Networks (HUNETs). Their system is based on a socially-aware broker allocation algorithm called social election. It dynamically controls the popularity of the nodes in order to select brokers or forwarders in the network, and also to find a balance between the efficiency and overhead. Contrary to DYMES, social election uses a local contact history to determine each node's role and also to select brokers.

Discussion

We share with the above-mentioned works the idea that location-based publish/subscribe abstractions are an important requirement for building vehicular applications that are user-oriented. However, existing publish/subscribe abstractions do not support the use of dynamic filters (Meier and Cahill, 2002b,a, 2010; Hughes et al., 2004) in their proposed matching mechanisms. In addition the location is only used for matching but is not available as a dynamic contextual information for the subscriber (Eugster et al., 2005; Holzer et al., 2013; Cugola et al., 2009; Cugola and Migliavacca, 2008). Moreover, location-based publish/subscribe abstractions must rely on underlying context-based message diffusion algorithms. These algorithms should propagate published messages and/or queries with low overhead (maximize the efficiency) while ensuring a high delivery ratio (maximize the reliability) over the considered network. Unfortunately, the major missing part of existing architectures is the absence (Meier and Cahill, 2002b,a, 2010; Hughes et al., 2004) and the lack of efficient and reliable communication protocols that if integrated to existing communication abstractions, can greatly improve the performance of vehicular social systems and push next generation vehicular applications.

We contribute to the existing body of research by providing novel and dynamic context-based publish/subscribe abstractions that rely on efficient and reliable broadcasting and multicasting protocols for different kinds of vehicular applications (Chapters 3, 4 and 5).

2.3 Context-aware Broadcasting Algorithms

In order to implement a broadcast abstraction in an ad hoc network, different protocols exist. Their main task is to determine which nodes forward messages so that nodes located outside of a sender's transmission range have a chance to deliver them. We call this task the forwarding decision. In the following we classify three kinds of protocols according to the nature of the information they use in their forwarding decisions, namely, probabilistic protocols, counter-based protocols and location-based protocols. Then we look at protocols that make use of the network coding mechanism (Ahlswede et al., 2006) in their forwarding decisions.

2.3.1 Probabilistic broadcasting protocols

The simplest protocol, called FLOOD, requires nodes to forward a message m upon its first reception. This protocol can be useful in settings with low message throughput or low densities, but can become problematic in other settings because of its elevated message load. In order to reduce this load, a forwarding probability can be added. For example, the simple gossiping protocol (Ni et al., 1999) requires nodes to throw a dice when they receive a message for the first time. If the resulting value is higher than a threshold p, the node forwards m, otherwise it remains silent. If p is too low, the efficiency of this protocol is still very low, but while increasing p allows to reduce the message load, it tends to decrease reliability rapidly.

2.3.2 Counter-based broadcasting protocols

Other approaches depart from these simple probabilistic mechanisms and use additional information in their forwarding decision. Many protocols use a variant of the *wait and count* mechanism, where nodes set a waiting time before retransmitting a message and cancel their scheduled retransmission if they hear some of their neighbors retransmitting the message before their own retransmission is due. The idea of such protocols is that message retransmission is delegated to nodes that scheduled the shortest waiting time. This allows to increase efficiency without jeopardizing reliability. The Counter-based Broadcasting Scheme (Haas et al., 2006) (CBS) is an example of such a protocol. In its forwarding decision, nodes set the waiting time randomly and cancel their scheduled retransmission when they hear a number of retransmissions that exceeds a threshold k.

2.3.3 Location-based broadcasting protocols

The wait and count mechanism offers a good starting point to build more sophisticated protocols. This is what some authors did by adding location-awareness to this mechanism (Ellis et al., 2009; Garbinato et al., 2008; Miranda et al., 2006; Paruchuri et al., 2003). The rationale behind this idea is that it is more useful if nodes located at the outskirts of a sender's neighborhood forward messages instead of nodes located close to the sender, since the former can reach a greater uncovered area than the latter. In order to achieve this, protocols such as the Power-Aware Message Propagation Algorithm (PAMPA) set the time according to the proximity with the sender (Ellis et al., 2009; Miranda et al., 2006), the closer the sender the longer the waiting time. In PAMPA, the distance between the sender and the receiver is computed based on the intensity of the signal with which the message is received.

The Six-Shot Broadcast (6SB) algorithm (Garbinato et al., 2010a, 2008) also uses location to adjust the waiting time in order to select the best forwarder. In fact, message forwarding is assigned to nodes located near six positions called targets. These targets are spread on the boundaries of the sender's transmission range; the goal is to compute a waiting delay that depends on the proximity of the receiving nodes to these targets. The closer the target the shorter is the waiting delay.

The Optimized Flooding Protocol (OFP) (Paruchuri et al., 2003) is quite similar to the 6SB. It is also based on the wait and count concept, which computes the waiting delay based on the proximity to the closest target. The main difference between 6SB and OFP is the number of the targets. In fact, 6SB is based on six targets, whereas OFP only uses three targets for the first hop and then two targets for the following hops.

The Area-Based Beaconless Algorithm (ABBA) (Ovalle;Martínez et al., 2006; Garbinato et al., 2010b) is somewhat different from the aforementioned location-aware broadcasting approaches. In fact, ABBA uses location in order to adjust the counting phase. It enhances the efficiency and the reliability in distributed mobile networks in the case when the neighborhood of a node n is fully covered by the neighborhoods of nodes that have previously sent a particular message. In this case there is no need for n to forward the message, since no additional node can be reached. ABBA's aim is the following: a node must forward the message after a certain waiting delay, unless its entire neighborhood is covered by the neighborhood of nodes which have already forwarded the message (Garbinato et al., 2010b).

Location-based approaches increase both efficiency and reliability compared to the previously presented approaches. However, as they rely on positioning systems, their performance can prove to be correlated to the accuracy of these systems.

2.3.4 Network coding aided broadcasting protocols

Whereas the previous mechanism aimed at reducing the message load produced by the dissemination of a single message, some authors have proposed to address the problem of reducing the message load of multiple message disseminations in the network. One such technique is called network coding (Ahlswede et al., 2006) and has originally been introduced to reduce message load and to achieve multicast capacity in wired networks. Its basic idea is that messages can be mixed in a single message using an *exclusive* or operation ($\oplus$), which allows to transmit several messages for the transmission cost of one message, as the size of the coded message is the same as the size of the biggest original message. Some authors have proposed to enhance broadcasting protocols using this technique (Asterjadhi et al., 2010), (Fragouli et al., 2008).

In (Fragouli et al., 2008) different broadcasting protocols are described based on network coding for a ring network topology and a square grid topology, as well as a generic protocol for random networks that involves neighborhood detection and multiple message transmissions to perform wise coding decisions.

Discussion

We share with the previous works the fact that commuters are socially isolated on the road and that they are only connected through wireless links. Moreover, they face similar traffic issues and lack efficient and reliable communication abstractions to help them discover their neighborhood and identify the best peers that match their common interests.

We also share with the previous works the fact that blindly disseminating real-time information in a highly dynamic vehicular environment hugely increases the amount of updates and messages' retransmissions over the network. The latter exacerbates the risk of message collisions and broadcast storms.

We contribute to the existing body of research by providing an efficient and reliable context-aware broadcasting protocol called BroadTrip that takes advantage of the performance of network coding and location-based wait-and-count approaches in order to reduce both the message load of single and multiple broadcasts in different vehicular node densities (Chapter 4).

2.4 Dynamic clustering and mobile gateway management algorithms

Different Clustering approaches were introduced in order to construct stable vehicle groups based on different metrics (Ucar et al., 2013; Zhang et al., 2011; Basu et al., 2001). In

fact, vehicles clustered in interest-based social communities can either share their common interests through distributed context-aware communication abstractions or be connected to the infrastructure through the selection of a small number of mobile gateways that ensure a smooth and reliable access to the infrastructure. In the following, we review existing approaches closest to our research work. That is works that make use of hybrid gateway management mechanisms in their context-aware communication abstractions.

In the proposed schemes (Zhang et al., 2011; Basu et al., 2001; Ucar et al., 2013), vehicle nodes broadcast beacon messages periodically. Once receiving two beacon messages, the nodes determine the packet delivery delay or the received power levels (Basu et al., 2001) and calculate the relative mobility metric with other vehicle nodes in their N-hop neighborhood. The vehicle node can then compute the aggregate mobility value based on the relative mobility metric and broadcasts its aggregate mobility value in the N-hop neighborhood. The vehicle node which has the smallest aggregate mobility value is selected as the cluster head node and broadcasts vehicle cluster information message in N hops. Other nodes work as cluster member nodes. This multi-hop clustering algorithm (Ucar et al., 2013) uses the changes in the packet delivery delay to compute the mobility metric among the nodes. However, the considered metric requires a precise synchronization among the vehicles. In addition, an efficient and reliable forwarding scheme that takes into account the dynamic parameters of VANET such as the location and the direction of vehicle nodes is highly needed.

Another stable grouping mechanism is devised in (Taleb et al., 2007) to support ITS services. The aim of this scheme is to group vehicles according to their moving directions in order to ensure that vehicles belonging to the same cluster are moving together using stable single and multi-hop paths. Communication stability within the same clusters are handled by selecting the most stable route using the ROMSGP scheme. In this scheme, each node inputs its mobility information before forwarding the RREP packet which results in a high number of forwarders that may decrease the performance of the grouping mechanism since there is no specific technique in ROMSGP that selects the best forwarders.

A seminal work (Benslimane et al., 2011a) has first introduced the concept of dynamic clustering in VANETs. In this work, a dynamic Clustering based on adaptive Mobile Gateway Management Mechanism (CMGM) is introduced. Vehicles are clustered based on the direction of movement, the received signal strength and the transmission range. Mobile gateway selection and discovery mechanisms have been fully analyzed in this work (Benslimane et al., 2011a) and in other recent works (Javaid et al., 2008; R. et al., 2009).

In fact, the CMGM (Benslimane et al., 2011a) is partially based on the mechanisms used in these works (Javaid et al., 2008; R. et al., 2009). It allows a minimum number of clustered

vehicles to link VANET to the 3G network through the selection of a Cluster Head (CH) that serves as a gateway. Once the serving gateway starts losing its performance, the handover operation is triggered in order to ensure a reliable migration between gateways. Gateway candidates closest to the CH are always aware of its information and act on its behalf in order to respond to the vehicles' solicitation messages during the gateway discovery and advertisement mechanisms. Although this work introduces the concept of mobile gateway and presents a solution to reduce the cost to access the infrastructure, it does not take into account the case where the majority of source vehicles desire to discover the gateway simultaneously, during a rush hour, in order to be connected to the infrastructure. This case may overload the network with a huge signaling overhead during the different phases of the proposed solution such as the clustering, the selection of the CH and the maintaining of the dynamic clusters. An efficient and cooperative mechanism that considers this kind of complex use cases and that optimizes the message forwarding process can greatly improve the performance of the proposed architecture. Moreover, the serving gateway in the mobile gateway handover mechanism always has a list of gateway-elected and decides about the new gateway-elects without checking their current state inside the cluster. These vehicles may have left the cluster before the start of the handover process which could negatively disturb the different multi-metric mobile gateway procedures. Furthermore, the serving gateway forwards the new incoming transactions to the new gateway-elects and notifies the sources about the current gateway-elects. This may generate a significant overhead in dense vehicular networks and may delay the sending of the users' requests to the infrastructure. In addition, some source vehicles may not receive the serving gateway notification and may be left isolated.

More in line with the idea of devising efficient routing strategies for connecting vehicles to the infrastructure, Benslimane et al. (Benslimane et al., 2011b) proposed a Lifetime-Based Routing (LBR) protocol that links vehicles to the Internet (Barghi et al., 2009), and that is partially based on the Power-Aware Message Propagation Algorithm (PAMPA) (Garbinato et al., 2010b). LBR aims at advertising the presence of a stationary gateway in a proactive manner through the broadcast of several control packets that fly across the vehicular network while maximizing the reliability over the network. The routing protocol used for the message forwarding process aims at selecting a route with the longest lifetime and a largest amount of progress (Garbinato et al., 2010b).

Authors have also studied the case of reactive gateway discovery. However and contrary to DYMES, these research works (Benslimane et al., 2011b; Barghi et al., 2009) have only focused on the gateway selection/advertisement and handover mechanisms in order to connect vehicles to the Internet through stable links, while neglecting the message retransmission cost over the network. Moreover, they did not pay the required attention to the impact of the

high mobility and the unpredictable behavior of vehicles on the selection of stable routes. In fact, they calculated a factor of driver behavior that predicts the stability of the drivers' behavior over time. However, this prediction depends on the history of the driving behavior and not on the current and real-time drivers' behavior at the moment of the relay selection.

Discussion

- *Dynamic clustering in hybrid VSNs*: We share with the previous research works the idea that clustering is an important network management task for vehicular networks since it reduces broadcast storms and improves the relaying of messages. However the main challenging issue in vehicle clustering is the frequent topology changes. Hence communication abstractions should consider the dynamic nature of the shared information in the design of efficient, stable and dynamic clustering strategies.

  Moreover, existing research works (Luan et al., 2015a,b) only tackled the problem of commuters' anonymity and their need to socialize without detailing the operating of their underlying communication protocols. The latter should take into account the heterogeneous nature of vehicular networks and should also ensure a private information sharing among users on the road. We contribute to the existing body of research by providing a novel online publish/subscribe grouping strategy that relies on an efficient, reliable and stable multicasting protocol and that aims at building stable and self-updated vehicular social communities at low costs while preserving the anonymity of the commuters' identities and the privacy of their shared information (Chapter 5).

- *Mobile gateway management algorithms in hybrid VSNs*: Existing research works introduced a VANET-3G architecture (Benslimane et al., 2011a) and a middleware for mobile context-aware applications, called Pervaho (Eugster et al., 2009). Messages are blindly forwarded among drivers during the CH selection and the gateway discovery/advertisement mechanisms (Benslimane et al., 2011a). This operation may flood the network when multiple source vehicles desire to communicate with the infrastructure at the same time.

  On the other hand, Pervaho (Eugster et al., 2009) does not consider the cost of client/server communications while sending publications and subscriptions to the server.

  Since, this cost induces a huge bandwidth consumption compared with the cost of ad hoc communications, publishers and subscribers would rather have to combine their messages in an ad hoc fashion and send them by a minimum number of users to the infrastructure.

We believe that the proposed approaches (Benslimane et al., 2011a; Eugster et al., 2009) need to be incorporated in a comprehensive middleware that allows to switch between networking strategies based on network architecture and on driver behavior predictions for particular online social application scenarios.

To the best of our knowledge, it still seems impossible to provide one-fits-all solution able of spanning the whole spectrum of vehicular social systems. In fact, existing vehicular social frameworks (Luan et al., 2015b,a) are either designed for distributed vehicular networks or centralized ones. Hybrid vehicular social frameworks are not yet available in the literature.

We contribute to the existing body of research by providing DYMES, a hybrid and dynamic context-aware messaging architecture that aims at enabling real-time social interactions among commuters in hybrid and dense environments. We focus on maximizing the efficiency (minimize the message overhead) and the message delivery (reliability), through distributed and centralized vehicular communication abstractions (Chapter 5).

CHAPTER 3 A DYNAMIC MESSAGING SERVICE FOR VANETs

As presented during the literature review in chapter 2, several research works advocate the design of a dynamic messaging system able to satisfy the online commuters' requirement in terms of enabling social interaction on the road by promoting the development of user-oriented vehicular social applications and services.

In fact, commuters are unknown and anonymous to each other and are sometimes hesitant about revealing their private information to an unknown public. The latter is one of many other reasons that constrains and limits their social interaction on the road (Luan et al., 2015b,a).

The publish/subscribe model is attractive for VSNs due to its natural match of anonymous communication (Vo and Bellovin; Eugster et al., 2003). In this model, messages are published based on the nature of their content rather than addressed to specific receivers. Using the Pub/Sub paradigm, commuters can publish their messages and all interested receivers subscribe to the published messages without either knowing each other.

Existing frameworks, Pervaho (Eugster et al., 2009) and LMA (Holzer et al., 2013) have exploited location information in their proposed publish/subscribe abstractions in order to build several context-aware applications. They typically add the possibility to limit the diffusion of messages in a certain geographical area. However, they do not allow to publish dynamic content and only allow some kinds of dynamic location-based filtering. Moreover, they lack efficient and reliable communication protocols able to minimize the overhead of the disseminated publications/subscriptions.

We argue that these works provide a foundation for building a particular type of publish/subscribe abstractions able to integrate the dynamic nature of the shared information as well as the contextual information of senders and receivers in order to ease the development of context-aware vehicular social applications that are commuters'-oriented. To overcome the latter shortcomings, we introduce a dynamic messaging abstraction through the Dynamic Publish/Subscribe Service (DPSS) that is at the center of the Dynamic Messaging System (DYMES). It allows to break the commuter's social isolation on the road by allowing them to publish dynamic context and to subscribe using online context-aware message filters.

This chapter is organized as followed. First, in section 3.1 we present DPSS, we discuss its API and we present a set of related APIs offered by DYMES. We illustrate and show their usage via two typical vehicular applications in section 3.2. Then, we identify and discuss

implementation issues that guide the architectural choices of DYMES in section 3.3. Finally, section 3.4 summarizes this chapter by highlighting its main contributions.

3.1 Dynamic Publish/Subscribe Service

The core idea behind the dynamic publish/subscribe service is that publisher can publish messages containing dynamic contextual information. Furthermore, these publications can be restricted to subscribers in a certain geographical location. In addition, subscriptions can use contextual information from the subscriber in their message filters, making these filters dynamic. Finally, subscriptions can also filter messages based on the geographical location of their publisher. As illustrated in Figure 3.1, the DPSS API proposes four core methods and relies directly on several classes outlined below.

1. *Message and Filters*: Messages are objects containing a list of key-value entries. Each entry in the list can be a static property added via the `addProperty` method or a dynamic context, added via the `addContext` method. MessageFilters contain a list of SQL-like queries used to select matching messages. These queries can either be static, as available mainstream messaging services such as JMS, or they can be dynamic, which means that messages can be evaluated against a contextual information. For these operations, the `addPropertyFilter` and the `addContextFilter` methods are used respectively.

2. *Publishing*: Messages are published via the publish method. Once published, dynamic context information contained in a message is updated when it changes. This allows developer to avoid having to track changes in contextual information and republish messages consequently. Publications are persisted until they are removed via the `unpublish` method. When publishing a message, the developer can define a scope. The notion of scope is useful in architecture with decentralized components in order to limit the unnecessary physical spread of information. A scope can either be defined as a mobile range around the publisher or as a fixed landmark, if the value of the scope is `null` no restriction is set. Both of these notions of scope have been proposed in the litterature. LPSS (cit, 2005) and STEAM (Meier and Cahill, 2002b,a, 2010) propose to limit publications to a range around the publisher, whereas CPS (Frey and Roman, 2007), CAPS (Cugola et al., 2009; Cugola and Migliavacca, 2008) and VAPS (Leontiadis, 2007; Leontiadis et al., 2009) propose fixed area where publications are propagated. We believe that both approaches should be incorporated in a comprehensive middleware.

```
public interface DPSS extends ContextListener{
  public Publication publish(Message m, Scope s);
  public Subscription subscribe(MessageListener l,
       MessageFilter f, Scope s);
  public void unpublish(Publication pub);
  public void unsubscribe(Subscription sub);
}

public interface MessageListener {
  public void onMessage(Message m);
}

public interface Message {
  public void addProperty(String name, String value);
  public void addContext(String name, Context c);
}

public interface MessageFilter {
  public void addPropertyFilter(String filter);
  public void addContextFilter(String filter);
}

public interface Scope {
  public Scope(int range);
  public Scope(Landmark area);
}
```

Figure 3.1 DPSS core APIs for Java (excerpt)

3. *Subscribing*: Subscriptions are created via the `subscribe` method. A subscription contains a filter used to discriminate messages of interest. The fact that this filter can use contextual information from the subscriber to do this job means that for example one can subscribe to messages relative nearby restaurants, or ask for warnings when a car with a much higher speed approaches, or subscribe to traffic jam information for cars going in the same direction. When a matching message is found, the `onMessage` method is triggered on the message listener contained in the subscription. The subscription also contains a scope that allows to limiting its spread in the network.

4. *Event Matching*: In order for a publication to match a subscription, every query of the

message filter must be met. It should be noted that certain subscription/publication pairs might match several time during the execution of an application, or that during a match, the value of the published message changes. When this happens, the `onMessage` method is called again, with updated values. Note that since both publications and subscriptions are subject to periodical contextual updates, there cannot be a strong consistency guarantee, meaning that depending on the update frequencies of the publisher and the subscriber, it may happen that a positive context-based match is computed on outdated information. In order to limit this effect one can add timestamps to contextual information.

3.1.1 Context

In order to manage context, DYMES provides several interfaces described in Figure 3.2.

```
public interface ContextProvider {
  public Context getLocation();
  public Context getVelocity();
  public Context getDirection();
}

public interface ContextListener {
  public void contextUpdate(Context c);
}

public interface Context {
  String name;
  String value;
  public void setContextListener(ContextListener l);
}

public interface Location implements Context {
  public int proximity(Location l);
}

public interface Direction implements Context {
  public float compare(Direction d);
}

public interface Speed implements Context {
  public float compare(Speed s);
}

public interface Route implements Context{
  public boolean onRoute(Location l);
  public boolean onRoute(Location l, Direction d);
}
```

Figure 3.2 DPSS Context related APIs for Java (excerpt)

The idea is that the DYMES implementation itself takes care of packaging the contextual information provided by different context sensors and offers it through one coherent set of APIs. Central to this set is the `ContextProvider` interface that gives direct access to the different contextual information available.

1. *Contextual Variables*: Contextual variables contain a name and a value and allow an external object to register as a listener via the `setContextListener` method. This allows them to be notified when the value of the contextual information changes via the `contextUpdate` method. An important class used in the context of vehicular applications is the `Route` class, which encompasses a set of coordinates representing a path from a point A to a destination D. It allows queries on whether or not a location or a moving object is on the path through the `onRoute` methods. It is a contextual information since the point A can be a moving entity.

2. *Query Language Extension*: Each `Context` object has particular methods that can be used in message filters through a query language extension. The extension of the language allows to trigger method call and test its outcome. For example in order to determine if a vehicle represented by a location `loc` and a direction `dir` is on route `myRoute`, the condition written in java code would be: `myRoute.onRoute(loc,dir)==true`, in the query language this would be translated into the following expression: `"myRoute ONROUTE loc dir = true"`.

3.2 DYMES Usage

In this section, we present *TrafficCheck* and *EasyCruise*, two typical vehicular applications and we show how DYMES can be used to program these applications.

3.2.1 TrafficCheck

In TrafficCheck, the idea is that every driver communicates his/her position, direction and speed to the driving community, so that other drivers can see the traffic on their route.

1. *TrafficCheck in action*: Figure 3.3 shows a use case of TrafficCheck. It should be noted that for such an application we assume that participants must somehow rely on an infrastructure in order to relay information to all interested parties. Figure 3.3.1 shows the situation on the ground where Alice and Bob are not in each other's vicinity and rely on a gateway to the Internet to communicate. In this use case, Bob has switched

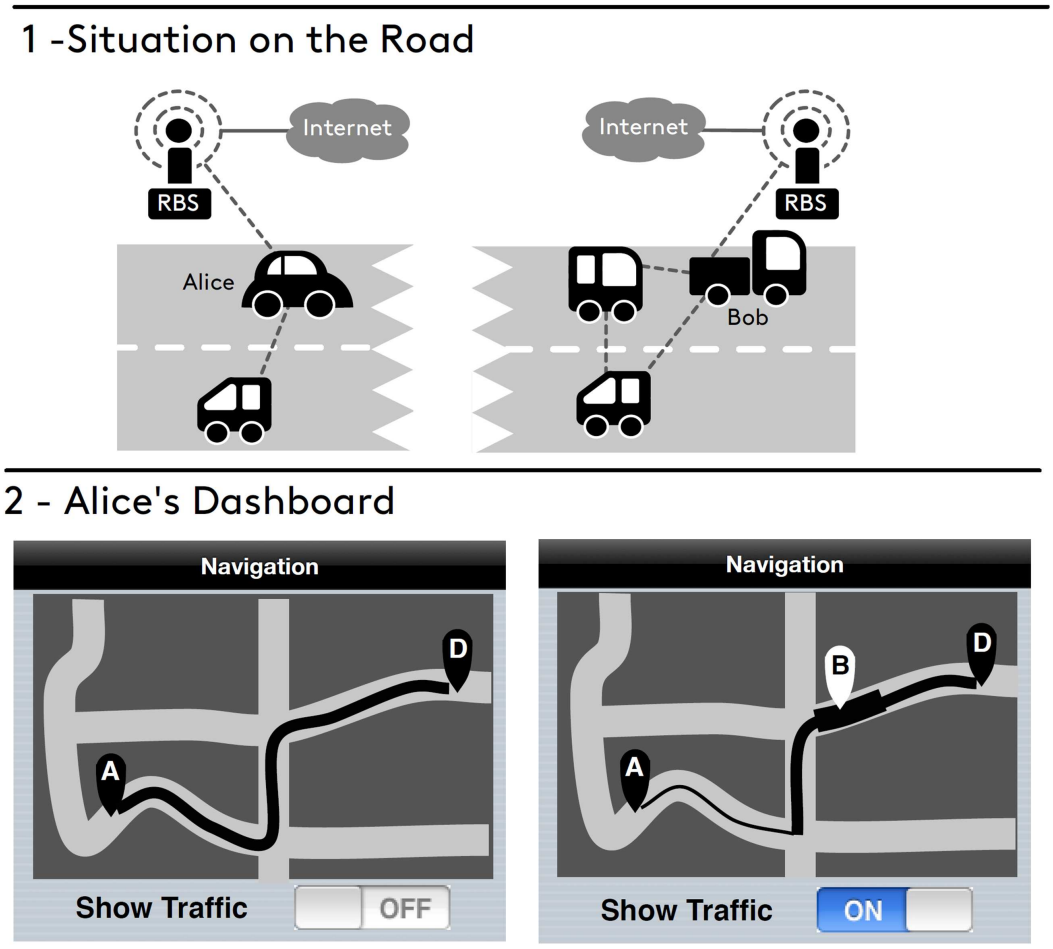

Figure 3.3 TrafficCheck in action

on his Navigation application and has allowed it to periodically send his location, speed and direction. On her Navigation user interface, depicted in Figure 3.3.2, Alice can see the directions she has to follow to reach her destination D from her current location A, when she slides the *Show Traffic toggle* in the *ON* position, her display will show the route with different thicknesses, the thicker the line the slower the traffic. In this example, Bob is currently where the letter B is shown.

2. *TrafficCheck under the hood*: An excerpt of the code of the TrafficCheck application is shown in Figure 3.4. When the application starts, `init` is called and a message containing the dynamic location, speed and direction of the publisher is published with a global scope, i.e., imposing no restriction by setting the scope to `null`. When Alice slides the *Show Traffic toggle* to the *ON* position, the `showTraffic` method is called with Alice's route as parameter. This generates a subscription to all messages where

the location and direction are on Alice's route. The subscription is created once per route. If the route is changed, the `reset` method is called and the subscription is removed. When Alice exits the application, both the publication and the subscription are removed via the `exit` method.

```
class TrafficCheck{
  DPSS dpss;
  Publication pub = null;
  Subscription sub = null;
  ConnectionFactory factory = new ConnectionFactory();

  public void TrafficCheck(){
    dpss = (DPSS) factory.createConnection();
  }

  public void init(){
    Message m = new MessageImpl();
    m.addStaticProperty("app", "TrafficCheck");
    m.addStaticProperty("CarID", MyNickName);
    m.addDynamicProperty("carLocation", myLocation);
    m.addDynamicProperty("carSpeed", mySpeed);
    m.addDynamicProperty("carDirection", myDirection);
    pub = dpss.publish(m, null)
  }

  public void showTraffic(Route myRoute) {
    if(sub==null){
      MessageListener l = new MessageListener(){
        public void onMessage(Message m){
          GUI.updateDisplay(m);
        }
      };
      MessageFilter f = new MessageFilter();
      f.addPropertyFilter("app LIKE 'TrafficCheck'");
      f.addContextFilter("myRoute ONROUTE location
           direction = true");
      dpss.subscribe(f, l, null);
    }
  }

  public void reset{
    dpss.unsubscribe(sub);
    sub=null;
  }

  public void exit{
    dpss.unpublish(pub);
    dpss.unsubscribe(sub);
  }
}
```

Figure 3.4 TrafficCheck with DYMES

3.2.2 EasyCruise

One of the promising application that we can envision for VSNs is automated driving, where drivers select their destination and the car's navigation system takes care of the rest. Such an advanced system will have to use input from roadside and mobile sensors as well as input from other cars in order for them to be coordinated. EasyCruise can be seen as a first step in this direction.

1. *EasyCruise in action*: Figure 3.5 shows a use case of EasyCruise. Alice drives in her pick up truck on a highway and decides to turn on her cruise control via the EasyCruise user interface. Her current speed is 119 km/h so that is the speed her cruise control will be set to. Then, Bob approaches Alice at a slightly higher speed. He receives a message asking him if he wants to join the platoon. He replies yes and his cruise control system sets his cruising speed to 119 km/h. Later as other cars approach and accept to join Alice and Bob, the platoon becomes bigger. Note that for this application a pure V2V communication can be considered sufficient since by definition platooning vehicles must be in each other's vicinity. As with traditional cruise control, when a car brakes, the cruise control is switched off.

2. *EasyCruise under the hood*: An excerpt of the code of the EasyCruise application is shown in Figure 3.6. When the application is started, typically when the driver enters a highway and presses the initialization button, `init` is called. This method creates a subscription. The idea of this subscription is to filter all messages about existing platoons in front of the driver with a similar speed to the driver, if such a message exists, it is delivered through the `onMessage` method and a notification is displayed on the drivers dashboard that allows her to join the group. If the driver decides to join the group, the system will set her speed to the group speed and the application will trigger the `startCruise` method with the same ID as the joined cruise. Note that EasyCruise limits the diffusion of messages to a 50 meter range around publishers and subscribers. When a car slows down or accelerates, the platooning mode is switched off and the `stopCruise` method is called in order to unpublish the message. When the application is switched off, the `exit` method is called in order to remove publications and subscriptions.

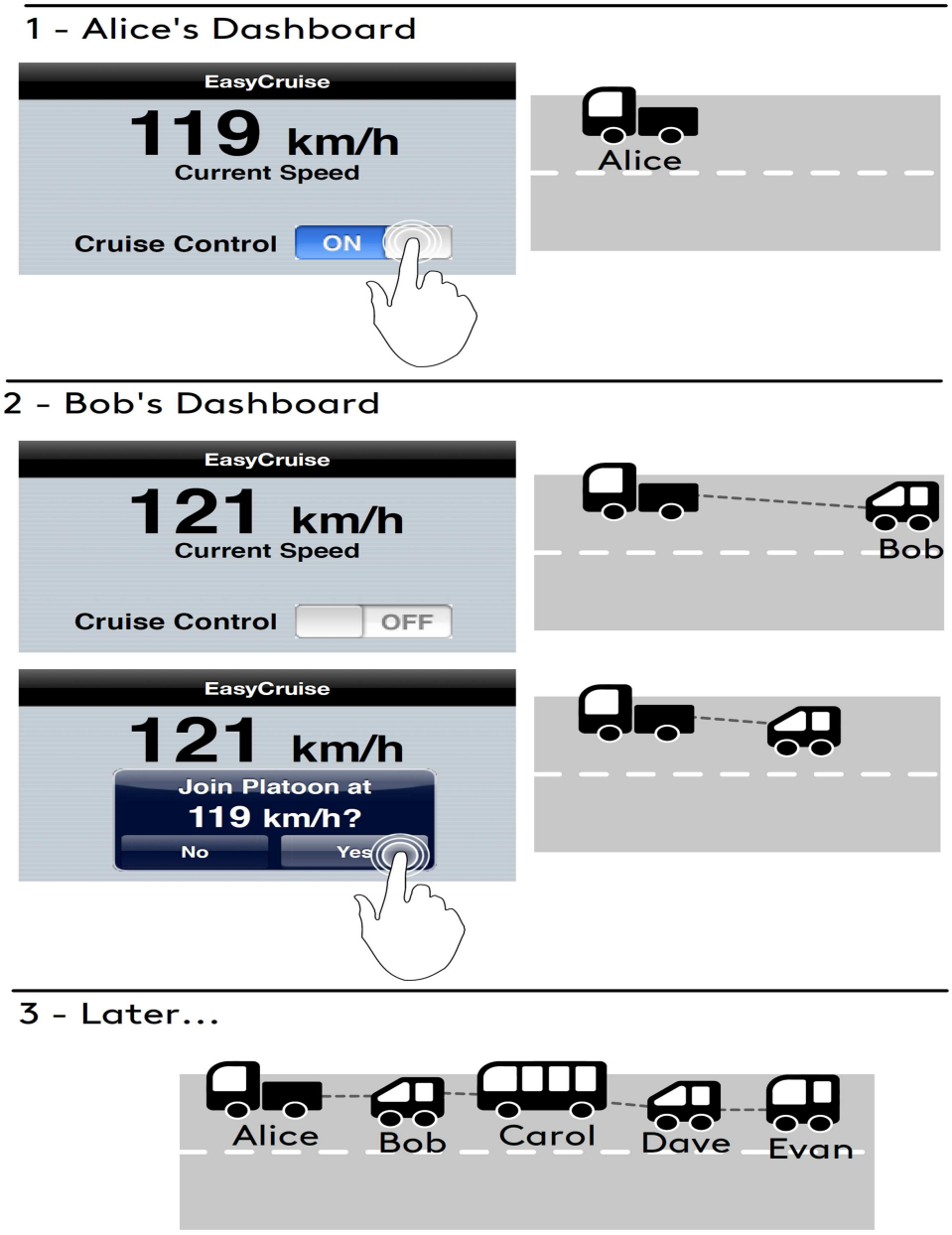

Figure 3.5 EasyCruise in action

```
class EasyCruise{
  DPSS dpss;
  Publication pub = null;
  Subscription sub = null;
  ConnectionFactory factory = new ConnectionFactory();
  Context myLocation;
  Context mySpeed;
  Route myRoute;

  public void EasyCruise(){
    dpss = (DPSS) factory.createConnection();
    myLocation=ContextProvider.getLocation();
    mySpeed=ContextProvider.getSpeed();
  }
  public void init(){
    MessageListener l = new MessageListener(){
      public void onMessage(Message m){
        display invitation to join platoon;
      }
    };
    MessageFilter f = new MessageFilter();
    f.addPropertyFilter("application LIKE 'EasyCruise'");
    f.addContextFilter("mySpeed COMPARE speed < 2");
    f.addContextFilter("myLocation PROXIMITY location <
        1000");
    f.addContextFilter("myRoute ONROUTE location direction
        = true");
    dpss.subscribe(f, l, new Scope(50));
  }

  public void startCruise(String ID){
    Message m = new Message();
    m.addProperty("application", "EasyCruise");
    m.addContext("speed", mySpeed);
    m.addContext("location", myLocation);
    m.addContext("direction", myDirection);
    pub = dpss.publish(m, new Scope(50));
  }

  public void setRoute(Route route){
      myRoute = route;
  }

  public void stopCruise(){
      dpss.unpublish(pub);
      pub=null;
  }

  public void exit(){
    dpss.unpublish(pub);
    dpss.unsubscribe(sub);
  }
}
```

Figure 3.6 EasyCruise with DYMES

3.3 Implementation Choices and Issues

In this section, we present the different implementation issues that have to be faced when it comes to implement a context-aware middleware for VANETs and present our implementation choices for DYMES. The general architecture of the DYMES framework can be captured by the layered architecture presented in Figure 3.7.

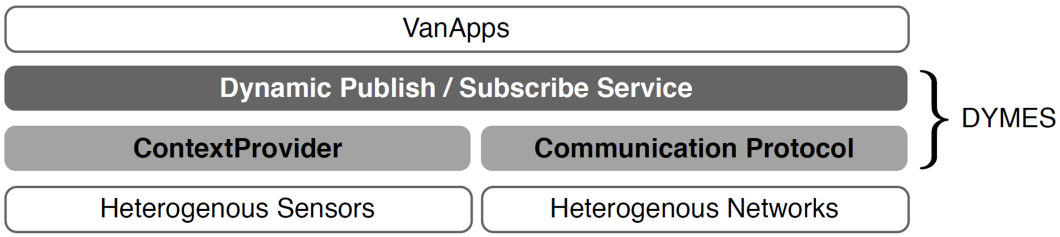

Figure 3.7 DYMES layered architecture

DYMES groups together a high-level communication layer, i.e., DPSS, with low-level communication protocols and a context provider. DPSS relies on a context provider layer to access different contextual information, and on a location-based communication protocol to disseminate information through the network. Using these two layers, the DPSS must deliver matching publications to subscribers. We point out the different implementation issues posed by these protocols and then discuss how we intend to address them in DYMES.

3.3.1 Providing context

Implementing a mechanism for transparently gathering contextual information is a challenging task when addressing heterogeneous devices. Tools such as the Location API in Java ME, or the Middlewhere (Ranganathan et al., 2004) platform address this issue by providing an API encapsulating the access to location. DYMES aims at building on these existing frameworks and provide an encapsulation for several different contextual inputs, i.e., location, speed, direction.

3.3.2 Communication and matching

Implementing a matching mechanism is coupled to the used communication protocol which depends on the underlying network topology, i.e., centralized, distributed, or hybrid.

1. *Centralized network*: In a mobile network, matching can be done on the cloud, i.e., on some remote server cluster. Publications, subscriptions and context updates are sent to

the cloud via a roadside base station. When the cloud find a match it multicasts it to the interested subscribers. The matching process on the cloud can leverage on an existing industrial publish subscribe engine, such as the Java messaging service (JMS), with an extra layer used to match and update dynamic contextual information. Typically, JMS can be used as a first filter to distinguish publication-subscription pairs that will never match, i.e., pairs having incompatible static content and filter, from pairs that can match at some stage, i.e, with compatible static content and filter (Eugster et al., 2009). The extra layer will then be in charge of matching and updating the second group of pairs. This layer obviously becomes the bottleneck of the matching process and thus should be a central preoccupation in the implementation. For these types of networks DYMES aims at providing a lightweight layer that allows context updates and dynamic matching.

2. *Distributed network*: In a fully decentralized architecture, matching and communication can be done in three different ways. First, matching can be done by subscribers, implying that only publications and the contextual information of publishers must be disseminated throughout the network.

   Second, matching can be done by publishers, meaning that subscriptions and the subscriber's contextual updates must be disseminated through the network and subsequently, matching publications can be routed to interested subscribers.

   Third, some intermediate node can do the matching, which implies that publications subscriptions and contextual updates of both subscribers and publishers must be disseminated and matches must then be routed. Each of these approaches can yield advantages depending on the application scenario and its communication pattern (Eugster et al., 2008; Holzer et al., 2013).

   These strategies must rely on underlying location-based message diffusion algorithms, and these algorithms can be improved to generate less traffic, since traffic congestion in ad hoc networks is the source of the broadcast storm problem (Ni et al., 1999). One approach is to reduce the number of relays need to broadcast or multicast a message to a group of nodes. In algorithms such as the counter-based broadcasting scheme (Ni et al., 1999), the goal is to minimize the number of relays for one message. Another approach consists of grouping several messages together and sent them simultaneously through the network. The network coding technique (Ahlswede et al., 2006; Ahmed and Kanhere, 2006; Frangiadakis et al., 2007) uses this approach and allows to encode and transmits several messages in a single transmission and decode them upon reception. DYMES investigates broadcasting and routing algorithms, which take advantage of

both approaches to reduce network traffic and increase message throughput.

3. *Hybrid network*: There are two ways to look at a hybrid network. First one can make the assumption that all vehicles have both an access to V2I communication as well as V2V capabilities. With such an assumption, matching can be expected to take place on the cloud. However, since the cost of V2I communication is greater for vehicles than the cost of V2V communication, publishers and subscribers should not individually send their messages and updates to the cloud, but rather message should be combined in an ad hoc fashion and sent by a reduced number of vehicles to the roadside base station. DYMES aims at providing a message diffusion algorithm for that purpose. Second, if we relax the assumption that all vehicles can access a roadside base station we fall into the research area of delay tolerant and opportunistic networks, where information must be stored on moving vehicles in order to get to a certain physical place, either to be handed over to a base station or to be brought to a group of vehicles. Protocols such as GeoOpps (Leontiadis and Mascolo, 2007b) address such issues.

3.3.3 Protocol modularity

Network heterogeneity, devices heterogeneity and application scenario communication pattern specifics advocate for a middleware where different protocols can be used depending on the environment. Middleware such as MANETKit (Ramdhany et al., 2009) tackle this issue by providing a framework based on protocol composition that allows to switch between different routing protocols at runtime. DYMES intends to provide such a mechanism and investigates the possibility of switching between protocols based on network architecture prediction for particular routes, and on communication pattern predictions for particular application scenarios.

3.4 Conclusion

Devising adequate programming tools for emerging distributed and centralized architectures is a challenging endeavor.

In this chapter, we presented a dynamic publish/subscribe abstraction for VSNs that is at the center of the DYMES system. Its particularity lies in the fact that it enables real-time social communication among commuters on the road based on their shared dynamic interests rather than their identities. The proposed abstraction allows to publish dynamic contextual information and to subscribe using dynamic context-based message filters whose content changes according to the dynamic nature of the commuters' shared information. We presented

the core APIs provided by DYMES and the query language extension used for dynamic filtering. We also showed how DYMES can be used to program distributed and centralized vehicular applications that are typical to vehicular social networks. Moreover, we sketched the DYMES' general architecture and identified and discussed DYMES' implementation choices and issues for different vehicular topologies.

Our next step is to fully implement DYMES for different network architectures and evaluate its performance with other existing messaging architectures.

CHAPTER 4 A DYNAMIC PUBLISH/SUBSCRIBE ABSTRACTION FOR DISTRIBUTED VSNs

Commuters stuck in traffic need to exchange real-time messages in order to control the current state of their congested commutes, they may also need to discuss recent political news or to play online games on the road for entertainment purposes.

A promising technology that matches the urgent commuters' needs on the road is the assisted and automated driving, where cars follow each other and play in turn the role of sender, relay or receiver in order to coordinate by message passing. It is essential to build such system on top of solid communication building blocks, such as reliable and efficient publish/subscribe abstractions that rely on scalable broadcasting protocols.

Nevertheless and as mentioned in chapter 2, devising an appropriate vehicular communication abstraction is a challenging endeavor due to the rapidly changing nature of the commuters' contextual information that increases the dissemination rate of contextual updates and decreases the efficiency and the reliability over vehicular dense networks.

This chapter presents a dynamic publish/subscribe abstraction for vehicles transiting in platoons called SocialDrive, that relies on an efficient, reliable and scalable broadcasting protocol called BroadTrip.

Our proposed strategy enables online social interactions among commuters clustered in platoons. It allows them to publish dynamic updates and to subscribe to the latest published updates using online context-aware filters. The updates dissemination is ensured using BroadTrip that aims at optimizing the number of updates retransmissions over the considered platoon and at providing high efficiency and reliability. BroadTrip combines a mechanism to reduce both the message load of a single broadcast (by using a location-based wait-and-count mechanism) as well as the message load of multiple broadcasts (by using network coding), and it does not require any neighborhood detection mechanism.

The remainder of this chapter is as follows. We first describe the operating of BroadTrip and detail its algorithm in section 4.1. We then present its simulation settings in section 4.3 followed by its performance evaluation in terms of efficiency and reliability in section 4.4. Then, we present and explain the algorithm associated with our proposed strategy SocialDrive in section 4.5. Section 4.6 presents the simulation results obtained in terms of efficiency and reliability. Finally, section 4.7 summarizes this chapter by highlighting its main contributions.

4.1 BroadTrip Protocol

Hereafter, we present the operating of BroadTrip. We first discuss and illustrate its concepts, before we present its architecture and finally we detail its algorithm.

4.1.1 Concepts

The idea of BroadTrip is to provide an efficient broadcast protocol specifically targeting vehicles transiting in a platoon. In order to do so, BroadTrip relies on two mechanisms. First, it uses a wait-and-count mechanism based on node proximity to increase the efficiency of a single message broadcast. Second, it uses a location-based network coding mechanism that allows to further increase efficiency by wisely combining messages without the need to acquire knowledge about its neighborhood. The idea is that a node n can code any message m_f coming from a node n_f located in front of it with any message m_b coming from a node n_b located behind it, since all node in n's neighborhood will normally either have delivered m_f or m_b before n sends the combination $m_f \oplus m_b$. [1]

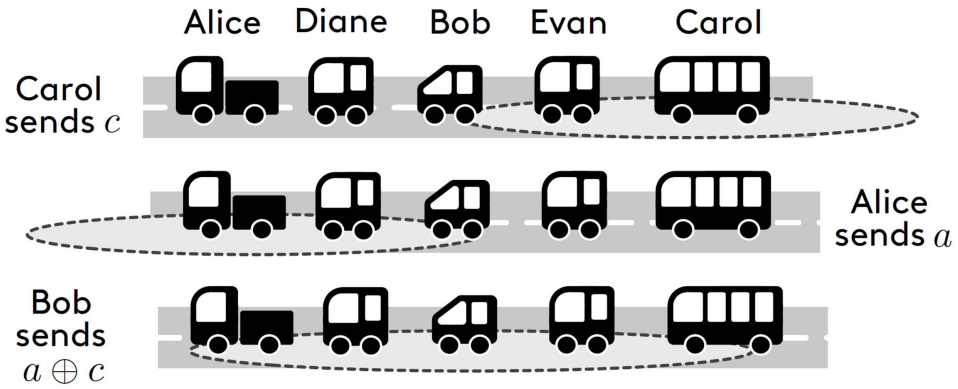

Figure 4.1 BROADTRIP illustration

Figure 4.1 illustrates the concept of BroadTrip on a small five car platoon with two senders. First, Carol broadcast a message c in her transmission range. Her neighbors, Bob and Evan, deliver c and set a schedule to retransmit c. Bob's waiting time will be shorter than Evan's since he is located further away from Carol. Then, Alice broadcasts the message a. Diane and Bob deliver a. Again, Bob will set a shorter waiting time than Diane since he is further from Alice. Finally, when Bob's waiting time expires he can combine both a and c into a message $a \oplus c$ and broadcasts it to his neighbors. Alice and Diane will retrieve c by performing $a \oplus c \oplus a$ and Diane will cancel her scheduled retransmission of a. During this time Carol and Evan will

[1] We make the assumption that all nodes have the same transmission range.

retrieve message a by performing a⊕c⊕c and Evan will cancel his scheduled retransmission of c.

4.1.2 Architecture

In terms of the OSI model, BroadTrip is part of the network layer as shown in the architecture depicted in Figure 4.2.

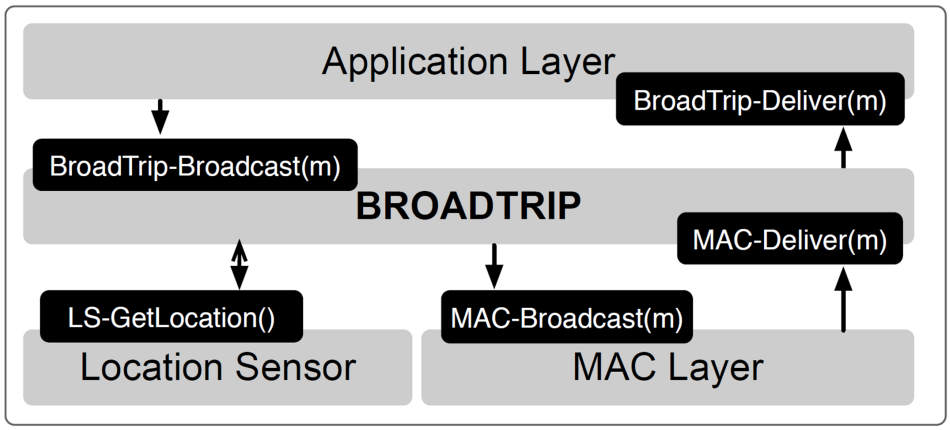

Figure 4.2 BROADTRIP's layered architecture

In terms of specifications, broadcast abstractions provide two primitives:

- broadcast(m): broadcasts a message m to every node in the network.
- deliver(m): works as a callback when a message m is received.

The BroadTrip protocol is accessible to the application layer through the Broadcast and Deliver primitives, called BroadTrip-broadcast and BroadTrip-deliver from now on to avoid confusion with their MAC layer counterparts. The MAC layer is a data link layer used to communicate with neighboring nodes. We model its communication capabilities via the following primitives:

- MAC-broadcast(m): broadcasts a message m to every neighboring node, i.e., every node located in the node's transmission range.
- MAC-deliver(m): works as a callback when a message m is received.

A central component of the BroadTrip architecture is the location sensor (LS). Its specification allows a node to retrieve its location via the following primitive:

- LS-getLocation(): returns the location of the localhost.

4.1.3 Algorithm

Algorithm 1 presents the details of the BroadTrip protocol. It can be decomposed in three elements: the broadcast and the deliver primitives as well as the refresh task.

- The broadcast primitive;
- The deliver primitive;
- The refresh task.

Algorithm 1: BroadTrip

```
1: uses MAC, LS
2: init
3:     schedule ← ⟨0, 0, ...0⟩                    {set of schedules indexed by message id}
4:     counter ← ⟨0, 0, ...0⟩                     {set of counters indexed by message id}
5:     M ← ∅                                      {set of messages to handle}

6: To execute BroadTrip-broadcast(m) :
7:     m.loc ← GetLocation()                      {sets location of m's sender}
8:     MAC-broadcast(m)                           {broadcasts m on the MAC layer}
9:     counter_m ← counter_m + 1

10: BroadTrip-deliver(m) occurs as follows:
11:    upon MAC-deliver(msg) do                   {when MAC delivers a message msg}
12:        msgs ← decode(msg, counter)            {decodes msg}
13:        for all m ∈ msg do                     {for all decoded messages}
14:            if counter_m == 0 then             {if m has not been handled}
15:                schedule_m ← GetSchedule(m)    {adds m}
16:                BroadTrip-deliver(m)           {delivers m}
17:                M ← M ∪ m                      {adds m}
18:                counter_m ← counter_m + 1      {adds m}
19:                if counter_m > k then          {if retransmissions exceed threshold k}
20:                    M ← M \ m                  {removes m from the handled messages}

21: task Refresh
22:    repeat every Δτ
23:    t ← GetCurrentMillis()                     {gets the time}
24:    for all m, n ∈ M|m ≠ n ∧ schedule_m < t ∧ schedule_n < t do
25:        l_m ← m.loc                            {gets message m's origin location}
26:        l_n ← n.loc                            {gets message n's origin location}
27:        if GetLocation() ∈ overline(l_m l_n) then   {if localhost between l_i and l_j}
28:            msg ← m ⊕ n                        {combines messages m and n}
29:            m.loc ← GetLocation()              {sets location of m's sender}
30:            m.loc ← GetLocation()              {sets location of n's sender}
31:            MAC-broadcast(msg)                 {forwards the coded message}
32:            M ← M \ m, n                       {removes messages m and n}
33:    for all m ∈ M/schedule_m < t do            {for all ripe messages}
34:        m.loc ← GetLocation()                  {sets the location of m's sender}
35:        MAC-broadcast(m)                       {forwards the message}
36:        M ← M \ m                              {removes m}
```

1. *Broadcasting (lines 6-9)*: Message broadcasts are initiated via the BroadTrip-Broadcast primitive with a message m as parameter. The location of the sender is added to m before it is broadcast over the MAC layer. Then, m's counter is incremented to avoid retransmitting it again.

2. *Delivering (lines 10-20)*: When the MAC layer delivers a message *msg* through the MAC-DELIVER primitive, *msg* is decoded via the DECODE function, which produces the set *msgs* of decoded messages contained in *msg*. Then, for each of the messages m in *msgs*, if m was received for the first time, m is delivered, added to the set of messages to retransmit M, and a schedule for its retransmission is set using the GETSCHEDULE function. This function computes a schedule using the location information contained in m. The greater the distance between the localhost and location of m's sender, the shorter the waiting time before m's scheduled retransmission. Finally, the counter of m is incremented and if its counter exceeds the threshold k, m is removed from M.

3. *Refreshing (lines 21-36)*: The refresh task is in charge of retransmitting messages. This task is executed every $\triangle\tau$. It goes through all the messages in the set M and for each pair it checks if the messages are ripe for retransmission. If they are, the task checks if the messages originated from opposite sides of the localhost. If this is the case, it combines the messages and retransmits the resulting message on the MAC layer after it added the localhost's location to the messages. Then, it removes the messages from the set of messages M. Next, it retransmits ripe messages that could not be paired with others. Again it adds the localhost's location to the messages and removes them from M after they are broadcast. In a all-to-all broadcast setting, we optimize the refreshing task in that we do not allow head and queue vehicles to retransmit others' messages. To do so without prior topology knowledge, nodes consider themselves to be at the extremity of the platoon until they receive a message from the front and from the back.

4.2 Analysis

In this section, we analyze the theoretical costs of BROADTRIP compared to the FLOOD protocol in terms of message load. In order to do so we consider a platoon of N vehicles (nodes), each one broadcasting a message m and following each other at a distance equal to their transmission range. These settings are especially harsh on the expected performance of BROADTRIP since it is expected to further reduce its message load in denser settings thanks to its location-based wait and count mechanism. We denote C_{FLOOD}, the cost of FLOOD. Trivially, it is equal to N^2 since all nodes retransmit each of the broadcast messages.

$$C_{\text{FLOOD}}(N) = N^2 \tag{4.1}$$

We denote $C_{\text{BROADTRIP}}(N)$ the cost of BROADTRIP. This cost is composed of the cost of flooding (N^2) minus the messages saved from the extremity optimization, minus the messages saved from network coding. Thanks to the extremity optimization -2N+2 messages are saved since the two nodes at the extremities of the platoon do not transmit any message except their own. For each node in the platoon, the number of messages it can code corresponds to the number of nodes in front of it or behind it, which ever number is smaller. Thus, the total number of messages that can be coded and saved can be expressed as $\Sigma_{i=1}^{N} \lfloor \frac{i-1}{2} \rfloor$.

$$C_{\text{BROADTRIP}}(N) = N^2 - 2N + 2 - \Sigma_{i=1}^{N} \lfloor \frac{i-1}{2} \rfloor \tag{4.2}$$

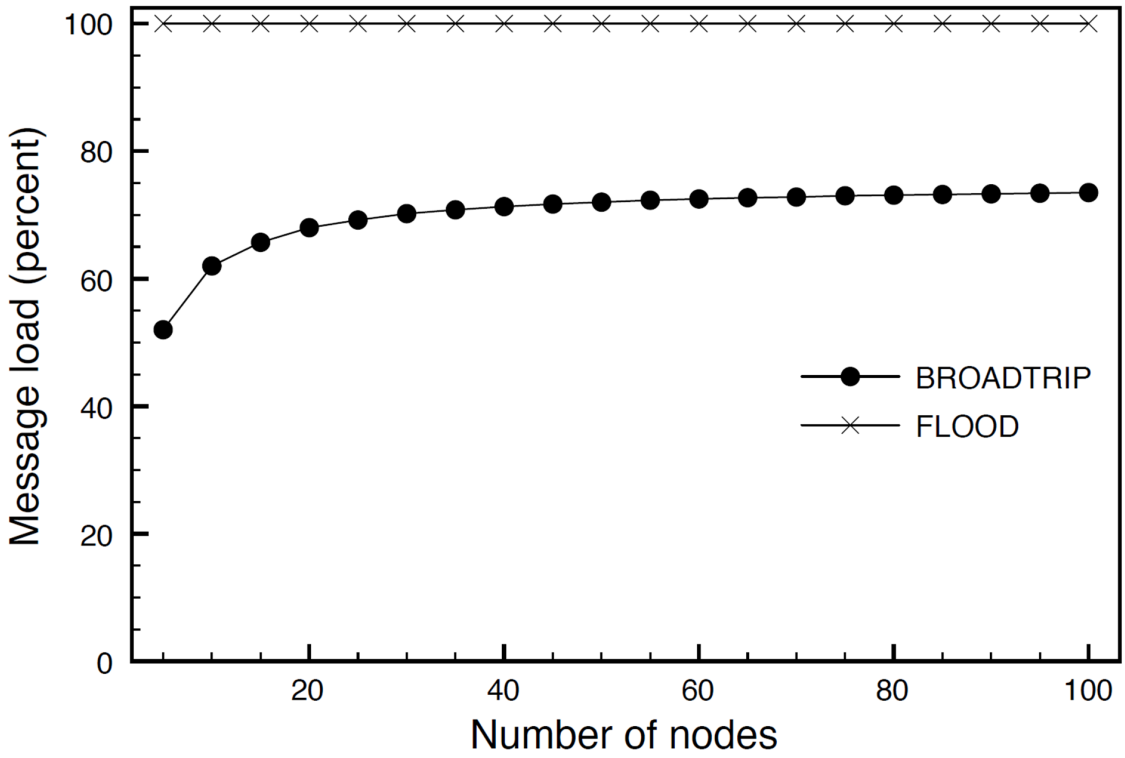

Figure 4.3 Message load prediction (FLOOD = 100%)

Figure 4.3 shows the expected cost of BROADTRIP compared to FLOOD according to N. The ratio between the cost of BROADTRIP compared to the cost of FLOOD tends towards 75% as the network grows.

4.3 Simulation Settings

In our simulation settings we consider a platoon of vehicles connected in a VANET with IEEE 802.11 communication capabilities and access to a positioning system, such as a GPS.

4.3.1 General parameters

We simulate this network on the Sinalgo[2] simulator, specifically aimed at simulating communication protocols in wireless networks. Table 4.1 summarizes the general simulations parameters. We deploy a platoon of vehicles represented by network nodes on a 12 *km* road. Each node has a communication capability with a transmission range between 80-100 *m* based on the Quasi Unit Disk Graph connectivity model (QUDG)[3] and send a total number of 50 messages each, at a frequency of 2 messages per second. Node mobility is set uniformly to 100 *km/h* to mimic a coordinated platoon, where nodes follow each other at constant speed. The message transmission time represents the delay between a BROADCAST call on one peer and the DELIVER callback on one of its neighbors.

Table 4.1 General Simulation Parameters

Parameters	Values
Connectivity model	QUDG
Transmission range	80-100 *m*
QUDG probability	0.8
Number of messages sent	50 msgs / node
Sending frequency	2 msgs / sec
Road length	12 *km*
Mobility	Uniform Platoon
Nodes speed	100 *km/h*
Message trans. time	0.1 *s*
Number of simulations	20

4.3.2 Protocol parameters

We compare BROADTRIP to the three closest broadcast protocols, namely, FLOOD, CBS, and PAMPA. Table 4.2 presents the parameters specific to each protocol, i.e., the message

[2]http://dcg.ethz.ch/projects/sinalgo/

[3]Nodes closer than 80 meters are always connected, whereas nodes between 80 and 100 meters have an 80% probability of being connected.

threshold k and the maximal waiting delay in the scheduling task.

Table 4.2 Parameters Of Evaluated Protocols

Parameters	FLOOD	CBS	PAMPA	BROADTRIP
threshold k	N/A	1	1	1
max delay	N/A	1 second	1 second	1 second

4.3.3 Variables

Table 4.3 presents the variables of our simulations. We simulate three platoon sizes, 10 nodes, 40 nodes and 100 nodes. Different platoon densities are simulated by varying the distance between nodes from 10 m - 80 m. In order to account for location measurement inaccuracies, we simulate maximal location errors from 0 m - 50 m.

Table 4.3 Variables

Variables	Values
Number of nodes	10, 40, 100
Distance between nodes (m)	10 - 80
Maximal location error (m)	0 - 50

4.4 Simulation Results

Hereafter, we present the results of our performance evaluation in terms of efficiency (message load) and reliability (delivery rate). Each data point represents the mean of 20 simulations, the standard deviations are minor. We first present a protocol comparison for different platoon sizes and densities assuming that location sensors provide accurate information. Then, we relax this assumption and introduce noise to the location sensor to evaluate the resilience to location errors.

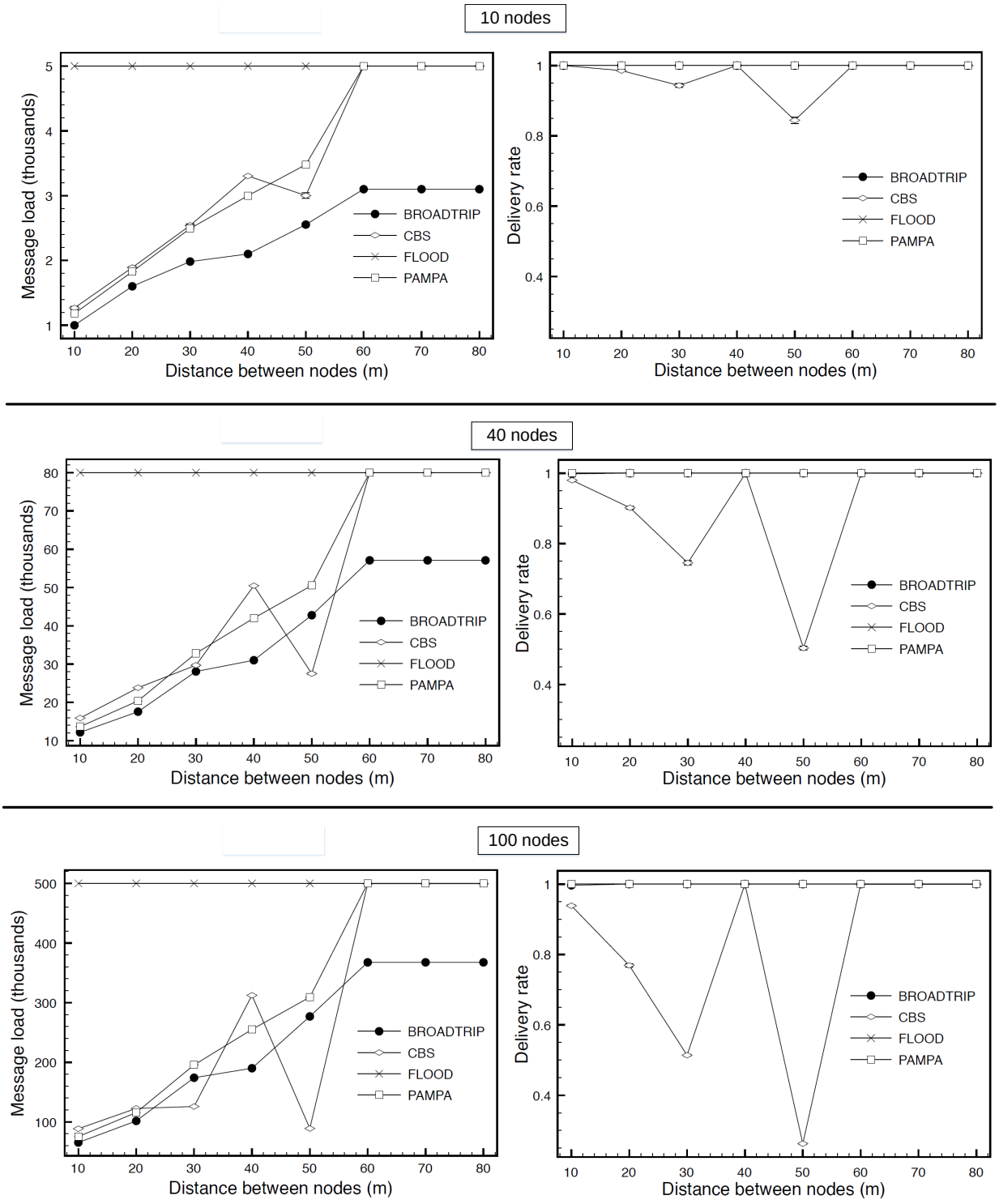

Figure 4.4 Protocol comparison - message load and delivery rate.

4.4.1 Protocol comparison

We compare BroadTrip with CBS, PAMPA, and Flood. The results show efficiency and reliability for different network sizes (10, 40 and 100 nodes) according to the distance between nodes in the platoon from 10 to 80 meters.

1) Results (Figure 4.4): We see that the trends are similar for small and large platoons. In terms of reliability, all protocols reach a 100% delivery rate or are very close, except CBS for inter-node distances below 20 meters and for a 50 meters distance. In these conditions, CBS exhibits a poor delivery rate as low as 25% in large platoons for settings with an inter-node distances of 50 meters. In terms of efficiency, BroadTrip outperforms all other protocols for any network size and any density. In dense platoons (10 meters between nodes) it beats PAMPA by 12% to 18%, it beats CBS by 27% to 31% and it beats Flood by a factor of 5 to 8. In sparse platoons (60 to 80 meters between nodes), where each node can only reach the next node in front and the next node behind it, Flood, CBS and PAMPA all perform equally and BroadTrip outperforms them by 26.5% to 38%.

2) Findings: The results of BroadTrip compared to Flood confirm the predictions presented in Section 4.2. That is, the cost of BroadTrip was expected to represent about 62% of the cost of Flood in sparse 10-node platoons, about 71.3% in sparse 40-node platoons, and about 73.5% in sparse 100-node platoons. The results also show that network density heavily penalizes CBS in settings where the distance between nodes is 50 meters and to a lesser, but still significant, extent for distances of 30 meters and below. The reason for this result is that in those settings, a node n_1 can decide to forward a message m it received from a node n_0 without reaching any new node, which means that the propagation of m will end. This problem is due to the variability of the transmission range of our simulation settings. If the transmission range is set to be constant (e.g. exactly 80 m, or 100 m) this performance drop would disappear.

4.4.2 Resilience to location error

We test the effect of location measurement error on BroadTrip and PAMPA, both relying on location information in their forwarding decisions. We simulate the behavior of these protocols for a 100-node platoon for different node densities (10, 50 and 80 meters between nodes) in function of the maximal location measurement error e expressed in meters. More precisely, each time a node accesses its location, a random number between and e is added to the returned value to represent the measurement inaccuracy.

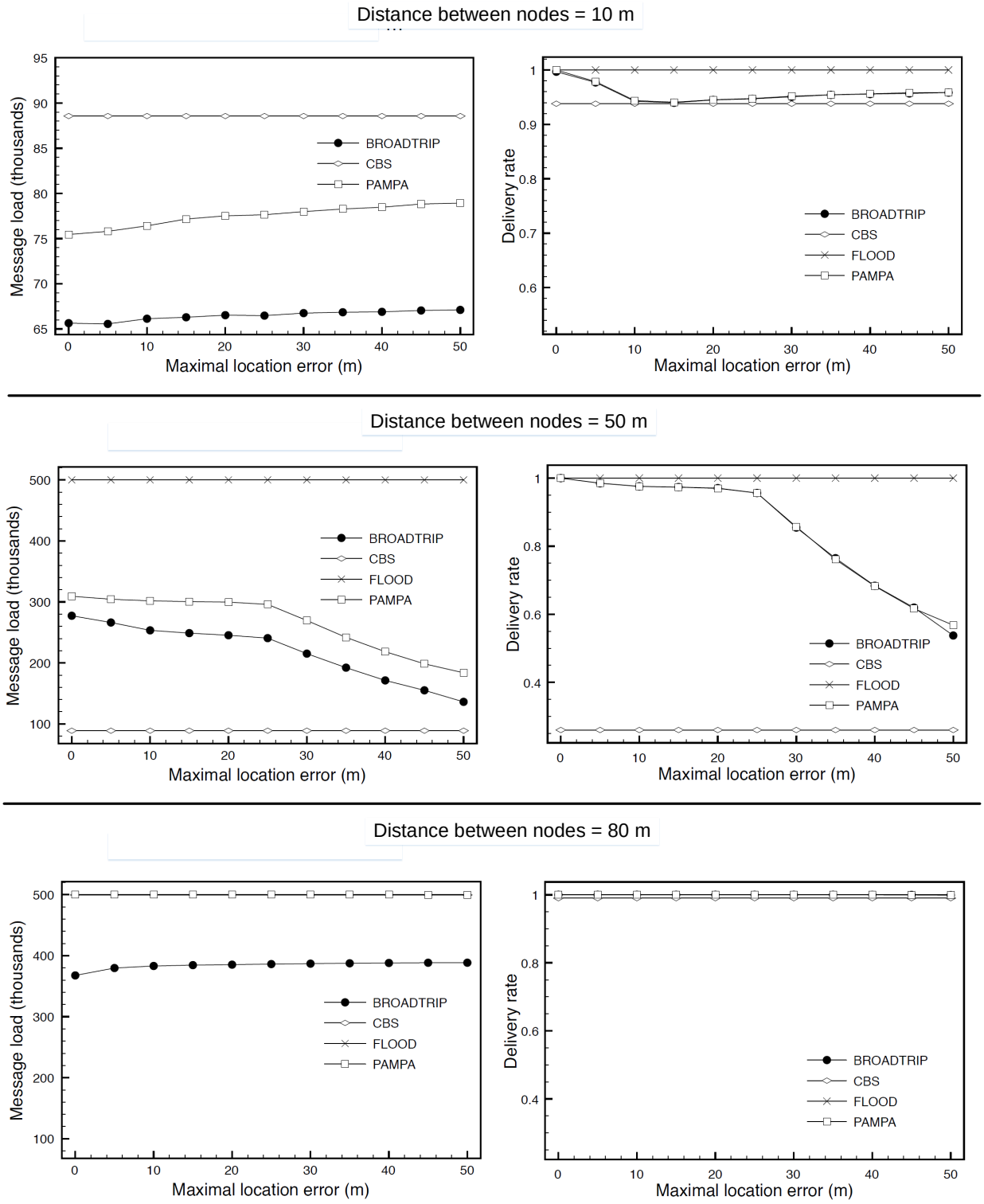

Figure 4.5 Resilience to location error – message load and delivery rate. 100 nodes.

1) Results (Figure 4.5): The results show that location measurement errors can diminish the reliability of BroadTrip and PAMPA in networks with several neighbors (10 and 50 meters between nodes). In dense networks, the effect on reliability is moderate, as it stays around 95% even when the maximal inaccuracy reaches 50 meters. In networks with 50 meters between nodes, the reliability stays also around 95% until the maximal inaccuracy reaches 25 meters and drops sharply after. In sparse networks with at most one neighbor (80 meters between nodes), reliability is not affected. However, the message load of BroadTrip in this setting increases with the location error to reach about 5% when the location error reaches 50 meters.

2) Findings: These results show that BroadTrip is resilient to location measurement errors in dense and sparse settings. In certain settings, however, such as in the 50-meter-distance-between-nodes setting, it tends to suffer from the same ailment as CBS. That is, as the transmission range is variable, some nodes might decide to forward a message and thereby cancel the forwarding schedule of their neighbors without reaching new nodes.

4.5 SocialDrive Strategy

Hereafter, we present the operating of SocialDrive. We first discuss and describe its concept, then we present its architecture and finally we detail its algorithm.

4.5.1 Concept

SocialDrive is an online social based publish/subscribe strategy for vehicles transiting in platoons. It aims at publishing dynamic and persistent contextual information and at subscribing to all persistent messages using an online context-based filter. Matching is executed by subscribers who filter all received messages based on the time at which they were created. SocialDrive relies on BroadTrip in order to efficiently and reliably disseminate the published persistent updates throughout the network.

Figure 4.6 illustrates the functioning of SocialDrive. When Mike publishes a persistent publication m, Tom and Helen deliver this publication and schedule each a waiting time that depends on the distance between them and Mike. The farther is the distance, the shorter is the waiting time. Then, Kevin publishes its persistent publication k. Marc and Helen deliver k and set a schedule to retransmit this publication, Helen's waiting time is shorter because she is farther than Marc. Finally, when Helen's waiting time expires, she combines m and k in one message $m \oplus k$ and broadcasts it to her neighbors. Mike and Tom will retrieve k by performing $m \oplus k \oplus m$, and Tom will cancel his scheduled retransmission of m. In the

meantime, Kevin and Marc will retrieve m by performing $m \oplus k \oplus k$, and Marc will cancel his scheduled retransmission of k.

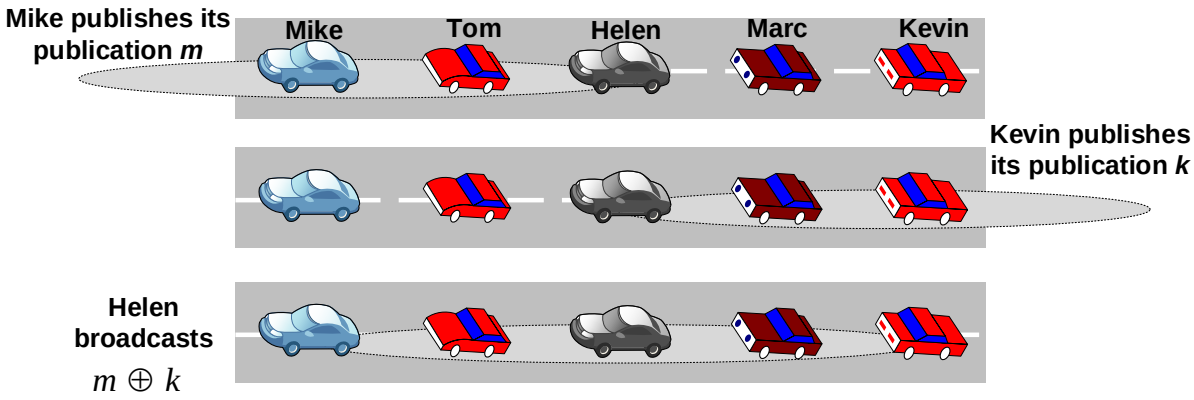

Figure 4.6 SocialDrive example

Once messages are delivered, subscribers execute the matching process illustrated in Algorithm 2. They are interested in latest published updates and create a subscription that contains a filter to compare the time at which publications were issued with the current time of their subscriptions. If the match is positive, the matching publication is delivered to the subscriber.

Algorithm 2: The matching process

1: **if** $sub.GetTime() - 1000 < pub.GetTime() < sub.GetTime() + 1000$ **then**
2: $\quad match \leftarrow match \cup \{< p, s >\}$ $\{The\ match\ is\ positive\}$

4.5.2 Architecture

SocialDrive's architecture is illustrated in Figure 4.7.

Each vehicle publishes its contextual information for a determined duration through the SocialDrive-publish primitive. The propagation of persistent publications throughout the network is based on the BroadTrip-deliver and the BroadTrip-broadcast primitives previously presented. Once the message is decoded and delivered, it is matched against the current subscription of the receiver via the SocialDrive-deliver primitive. The match is positive if and only if, the time at which the update was issued is approximately close to the subscription time.

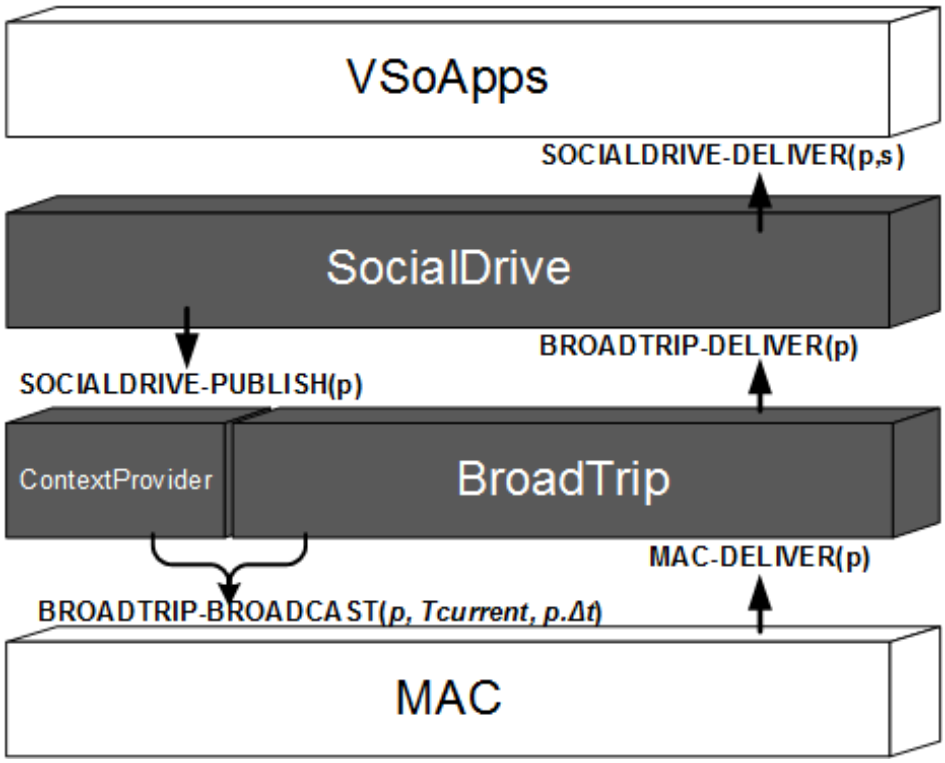

Figure 4.7 SocialDrive's architecture

4.5.3 Algorithm

Algorithm 3 presents the details of the SocialDrive strategy, it can be decomposed in four elements: Publishing, Subscribing, Matching and Refreshing.

Algorithm 3: SocialDrive

```
1: uses BroadTrip
2: INIT
3:     pubs ← ∅
4:     match ← ∅
5:     subs ← ∅

6: To execute SocialDrive-publish(p) :
7:     pubs ← pubs ∪ {p}                                  {p is added to the list of publications}

8: To execute SocialDrive-subscribe(s) :
9:     subs ← subs ∪ {s}                                  {s is added to the list of subscriptions}

10: SocialDrive-deliver(p, s) occurs as follows:
11:     upon BroadTrip-deliver(p) do                      {after decoding}
12:         for all s ∈ subs do                           {for all created queries}
13:             if ContentMatch(p, s) then
14:                 if ⟨p, s⟩ ∉ matches then              {if the match is new}
15:                     SocialDrive-deliver(p, s)         {delivers the matching msg}
16:                     match ← match ∪ {< p, s >}        {adds the match}

17: task Refresh
18:     repeat every Δt
19:     for all p ∈ pubs do                               {for all created publications}
20:         BroadTrip-broadcast(p, t_current, p.Δt)
21:         if p.Δt ≠ ∞ then
22:             p.Δt ← p.Δt − Δτ                          {decreases the lifetime}
23:             if p.Δt ≤ 0 then                          {if p is obsolete}
24:                 pubs ← pubs \ p                       {removes p}
```

1. *Publishing (lines 6-7)*: A persistent publication p is added to *pubs* in order to be published via SocialDrive-publish(p) primitive. This update or message is removed by the refresh process once its lifetime expires.

2. *Subscribing (lines 8-9)*: Each vehicle creates a subscription s in order to filter instantaneous messages.

3. *Refreshing (lines 17-24)*: The refresh task is executed every Δt and is in charge of sending newly created updates using BroadTrip-broadcast($p, t_{current}, p.\Delta t$) *(line 20)* primitive. These carry current and persistent updates, whose Time To Live *(TTL)* decreases with time.

4. *Matching (lines 10-16)*: The matching process is initiated at the SocialDrive layer, once a message is received for the first time by a subscriber using the BroadTrip-Deliver primitive. The message is decoded using the Decode function, at the BroadTrip level, in which two messages are extracted on both sides at the same time. The matching process compares the creation time of the published persistent update p with the creation time of the subscription. Once the match is positive, the publication is delivered using the SocialDrive-deliver callback, and the match is added to the content matches.

4.6 Performance Evaluation

In this section, we present the simulation parameters and results obtained using the Sinalgo simulator, which is specifically aimed at simulating wireless network protocols. We consider a platoon of vehicles connected in a VANET with IEEE 802.11 communication capabilities and access to a positioning system, such as a GPS.

4.6.1 Simulation Settings

Table 4.4 represents the general parameters of our simulations. We simulate different platoon sizes while varying the distance between vehicles from 2 m - 20 m, which represents a high node density.

Each node publishes 10 persistent publications that last for 10 seconds. Matching is performed by subscribers who are interested at receiving the latest issued updates, within a short time interval.

Table 4.4 Simulation Parameters

Parameters	Values
Connectivity model	QUDG[4]
Transmission range	80-100 m
QUDG probability	0.8
Road length	12 km
Mobility	Uniform Platoon
Nodes speed	100 km/h

4.6.2 Simulation Results

In this section, we present the performance evaluation of SocialDrive strategy in terms of efficiency (update load) and match rate (the ratio of the updates matched against current created subscriptions in a short time interval) in high node density. We compare SocialDrive's performance with the same strategy but based on Counter-Based Scheme (CBS). We chose CBS scheme because it provides prime performance in terms of efficiency and reliability (Ni et al., 1999). We first present a strategy comparison for different platoon sizes where the average distance between vehicles is 10 meters. Then, we evaluate the strategy performance while fixing the platoon size to 80 nodes and varying the distance between vehicles from 2 to 20 meters.

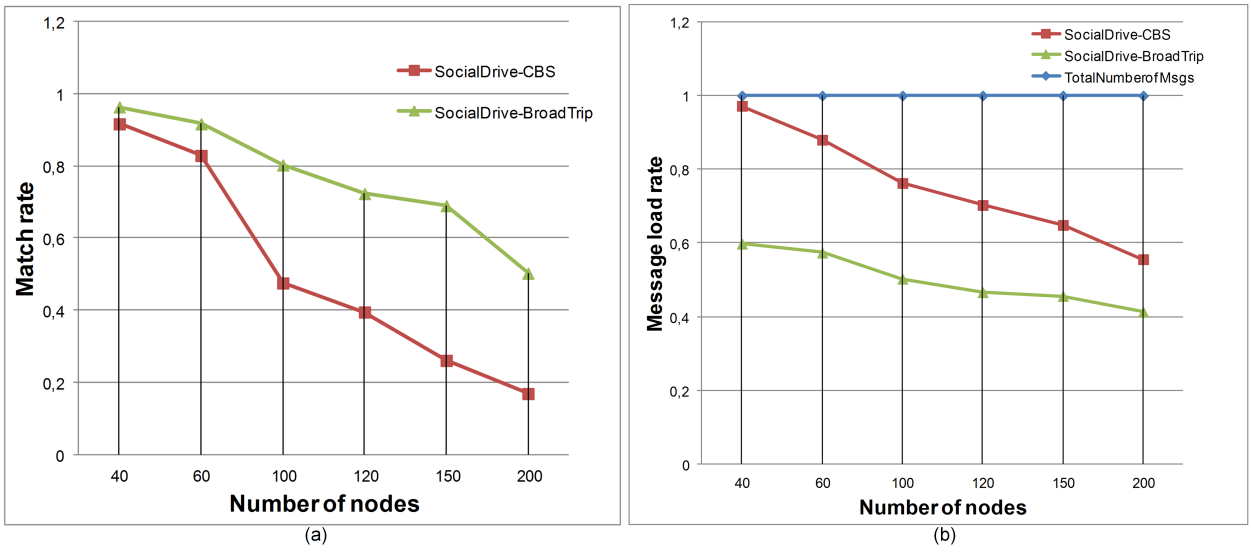

Figure 4.8 (a) Match rate. (b) Message load rate. Distance between nodes = 10 m

Figures 4.8 illustrates the variation of the match rate and the message load rate with the

number of vehicles for each strategy. The average distance between vehicles is equal to 10 meters.

In terms of the match rate (see figure 4.8(a)), SocialDrive-BroadTrip outperforms SocialDrive-CBS strategy: In dense platoons where the number of nodes varies from 60 - 150 nodes, 68% to 96% of the latest issued updates match the current nodes' subscriptions in a short time interval that is one second. However, 26% to 82% is the total match rate of the SocialDrive-CBS using the same scenario.

Figure 4.8(a) also shows that the performance of SocialDrive-CBS decreases drastically as long as the size of the platoon increases. For a platoon of 200 nodes, 50% of the published updates match the current nodes' subscriptions using SocialDrive-BroadTrip strategy. However, SocialDrive-CBS fails to reach the same ratio and only produces 16% of positive matches in the same short time interval.

On the other hand, figure 4.8(b) illustrates the variation of the message load rate with the number of nodes for each strategy. The results show that the message load rate of SocialDrive-BroadTrip strategy represents 61% to 74% of the SocialDrive-CBS message load rate and 42% to 59% of the total expected number of messages exchanged throughout the network.

The reason that stays behind these results is that when we increase the size of a platoon, we increase the number of forwarders since the average distance between vehicles is 10 meters, hence the message load rate decreases.

However, since both strategies rely on different messages retransmission mechanisms, the match rate greatly depends on the speed by which messages are retransmitted throughout the network. ScialDrive-BroadTrip relies on a location based wait-and-count mechanism and on network coding that combines and forwards two messages at the expense of one message throughout the network, however, CBS establishes random waiting delays in order to select the messages' forwarders. The latter delays the messages' retransmission process which lowers the chances for commuters to receive latest issued updates.

Figures 4.9 portrays the variation of the match rate and the message load rate with the distance between nodes for both strategies.

The results show that SocialDrive-CBS fails to reach the same match rate of SocialDrive-BroadTrip strategy even if the vehicles are located very close to each other (2 - 5 meters).

In fact, SocialDrive-BroadTrip strategy succeeds to meet the commuters' needs in terms of producing positive matches with the latest issued updates, for an inter-node distance equal to 2 meters. In addition, its match rate curve decreases slightly for inter-node distances

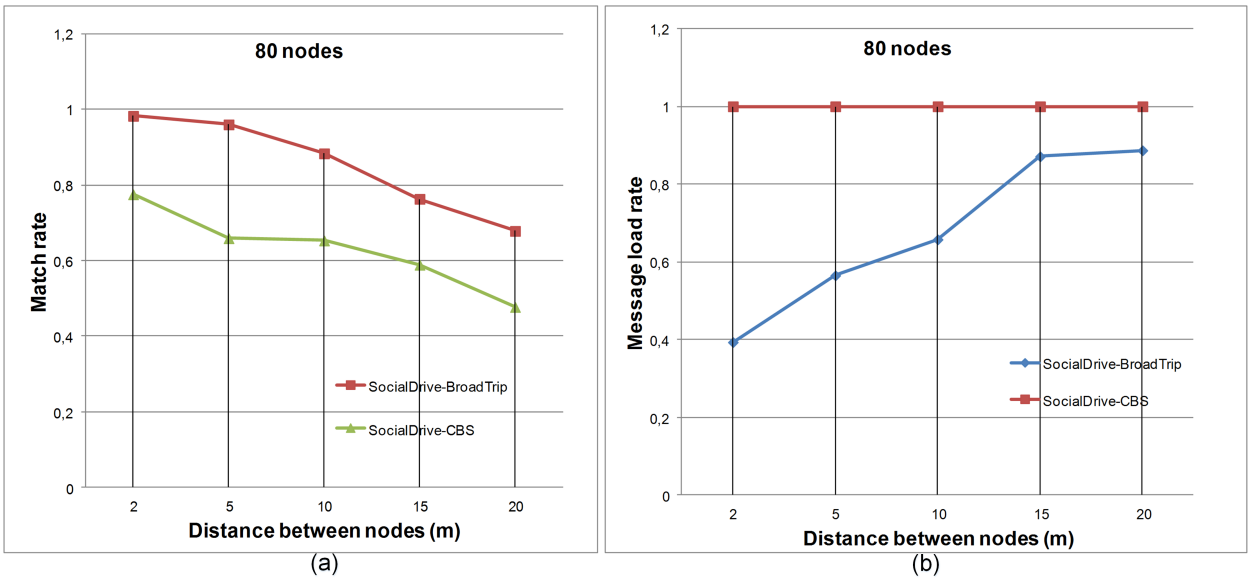

Figure 4.9 (a) Match rate. (b) Message load rate. Number of nodes = 80 nodes

between 2 and 20 meters, and reaches 68% of positive matches with a very low message load rate for inter-node distance equal to 20 meters compared to only 46% of positive matches for SocialDrive-CBS for the same inter-node distance (20 meters).

The reason that stays behind these results is that when the distance between nodes is too small, SocialDrive-BroadTrip strategy reduces the message overhead by generating several coded updates combinations that travel throughout the network using a few number of forwarders thanks to its location-awareness and its use of the network coding technique. This strategy greatly helps nodes avoid messages collisions and deliver a high number of positive matches. It also keeps them up to date about current issued information in dense communities.

However, SocialDrive-CBS strategy selects the updates' forwarders based on random waiting times. The latter increases the number of forwarders, causes more message collisions and lowers the number of positive matches. In fact, a node may not produce a positive match during the selected time interval in the subscription filter, but it can do it later (out of the selected time interval) and also it can decide to forward an update without reaching any new nodes, which means the end of the dissemination of this update.

4.7 Conclusion

The synergy between social networking and vehicular communication systems brings a novel way of social interaction among commuters on the road. It allows to enhance the quality of communication between them through a new kind of user-centric services.

In this chapter, we presented SocialDrive, an online publish/subscribe strategy specifically designed for communication between vehicles transiting in platoons. It allows commuters to publish their dynamic updates throughout the network and to subscribe using an online context-aware updates filter. Upon the delivery of updates, each subscriber creates a subscription and uses the time at which he created its subscription in the matching filter in order to compare it with the time at which the received updates were issued. A match is positive, if and only if, the time at which the update was issued is very close to the time at which the subscription is created.

SocialDrive uses BroadTrip in order to disseminate updates throughout the platoon. BroadTrip combines a proximity-based mechanism to reduce the message load generated by the broadcast of a single message and a network coding technique to mix messages coming from different directions. Our performance evaluation showed that BroadTrip significantly outperforms other existing alternatives in terms of efficiency for a comparable reliability and that is fairly resilient to location measurement errors. Moreover, simulation results also proved that SocialDrive-BroadTrip improves the timely and cooperative communication among commuters compared to other strategies that fail to deliver the same number of positive matches using the same matching filter. The encouraging results we obtained prove that SocialDrive-BroadTrip strategy can be used as a building block for the development of any kind of distributed vehicular social applications.

CHAPTER 5 DISCOVERING THE ARCHITECTURE OF A DYNAMIC MESSAGING SYSTEM FOR VEHICULAR SOCIAL NETWORKS

As mentioned in chapter 2, there are several missing features in the proposed vehicular communication systems. More precisely, existing works only address and focus on single aspects of vehicular social networking area that only include the design of services and software platforms for the development of a specific kind of vehicular social applications (Benslimane et al., 2011a; Eugster et al., 2009; Smaldone et al., 2008; Luan et al., 2015b,a). These works do not support the full design and the development of comprehensive architectures that rely on both vehicular social services and scalable communication routing protocols that perform under harsh vehicular conditions such as high mobility, limited bandwidth, and congested traffic. Moreover, the commuters' anonymity issue should not be neglected in vehicular social networks since users share common interests, but lack specialized vehicular communication abstractions that facilitate the creation of vehicular communities based on the different commuters' interests.

To overcome these challenges, we present a dynamic messaging system for vehicular social networks called DYMES. The goal of our proposed architecture is to promote real-time social interactions among commuters during their highway travels through efficient and reliable distributed and centralized vehicular communication abstractions.

DYMES copes with the commuters' anonymity in highly dense networks through a Dynamic Publish/Subscribe Clustering Strategy (DPSCS). It allows each commuter to create its own stable and self-updated vehicular community through the multicast of a single and persistent message, while ensuring high efficiency and reliability throughout dense distributed networks.

DYMES deals with the scarcity of the radio resources in dense vehicular networks, the commuters' intermittent connectivity and the extensive bandwidth consumption in infrastructure-based vehicular networks, through the introduction of a cooperative and efficient Mobile Gateway Discovery/Selection Strategy (MGDSS). The latter provides each commuter an opportunity to be informed about real-time traffic conditions and to select a minimum number of mobile gateways in order to ensure an efficient and reliable connectivity with the infrastructure.

The main difference between our cooperative mobile gateway discovery/selection strategy and the existing ones (Benslimane et al., 2011a), is that all vehicles are able to control the current state of their congested commute at the same time. In addition, the majority of vehicles can access the infrastructure, simultaneously, without requiring the sending of any

control messages, and without waiting to receive replies from an already defined cluster head (Benslimane et al., 2011a).

Establishing a stable route to connect vehicles with the infrastructure has been widely studied in the literature (Benslimane et al., 2011c). The goal was to allow users to download files from the Internet, which is similar to traditional vehicular applications, such as the GPS which connects to the server in order to provide information to individual users. However, since commuters have common interests and preferences on highway trips, they are eager to share/receive dynamic contextual information from other users, which requires particular communication abstractions. DYMES meets the requirements of commuters on the road, and proposes a novel online matching strategy that aims at sending the commuters' publications and subscriptions to the infrastructure through the selected mobile gateway. Our online matching strategy aims at updating the commuters' subscriptions and at sending the matches to the corresponding users through stable wireless links.

The next challenge that DYMES deals with is when commuters start losing their wireless connectivity with their current selected mobile gateway. At this moment, commuters launch an efficient and reliable handover mechanism that consists in rediscovering their new neighborhood in order to define a new mobile gateway.

The remainder of this chapter is organized as follows. Section 5.1 presents DYMES architecture, explains its operating and details the algorithms associated with the proposed strategies. Section 5.2 presents the simulation results in terms of reliability and efficiency. Section 5.3 concludes and points out some future research directions.

5.1 Proposed Dynamic Messaging Architecture

The topology of our proposed dynamic messaging architecture is hybrid and is illustrated in Figure 5.1. The network scenario considers three types of node: ordinary vehicles, selected mobile gateways that are moving in two directions, and Base Stations (BS). Vehicles are equipped with GPS devices allowing them to obtain their locations, speeds and directions. Vehicles communicate with each other through V2V communications, they also communicate with the infrastructure (BS) through selected mobile gateways (V2I). By a link, we mean a direct link between two adjacent vehicles, whereas, a route is composed of multiple links between n vehicles.

Figure 5.2 portrays the DYMES architecture that is mainly composed of two modules: the Dynamic Publish/Subscribe Clustering Strategy (DPSCS) and the Mobile Gateway Discovery/Selection Strategy (MGDSS).

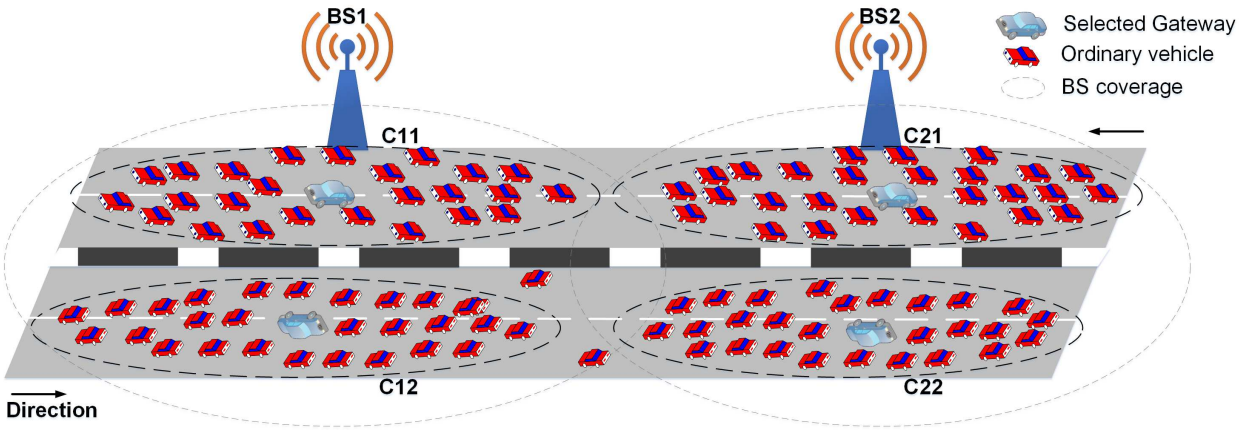

Figure 5.1 Proposed dynamic messaging architecture

5.1.1 Dynamic Publish/Subscribe Clustering Strategy

Hereafter, we present the operating of DPSCS. We first discuss and describe its concept, then we present its architecture and we detail its algorithm.

- **1 Concept**

DPSCS aims at facilitating the commuters' social interactions despite their anonymity on highway trips, and without flooding the network. It provides the opportunity for each commuter to build a stable and self-updated community based on its own interests. DPSCS is a distributed context-aware social clustering strategy that relies on two mechanisms. First, it introduces a novel dynamic publish/subscribe strategy that aims at exploring the commuters' common interests through the propagation of a single and dynamic publication around the publisher, for a determined duration *TTL (Time-To-Live)*, leaving the online matching process to subscribers. Second, the propagation of this persistent publication throughout the network relies on the *two Shot-Stable Routing Service* (2S-SRS). The latter improves the efficiency and the reliability of the commuter's single publication disseminated over the network. In fact, it allows to forward the commuter's single message by nodes that have a stable connectivity with the sender, and that are farther from the sender's location, and closer to two boundary points. It is worth noting that the building of the commuter's community and the propagation of its single publication stops once the TTL of its publication expires.

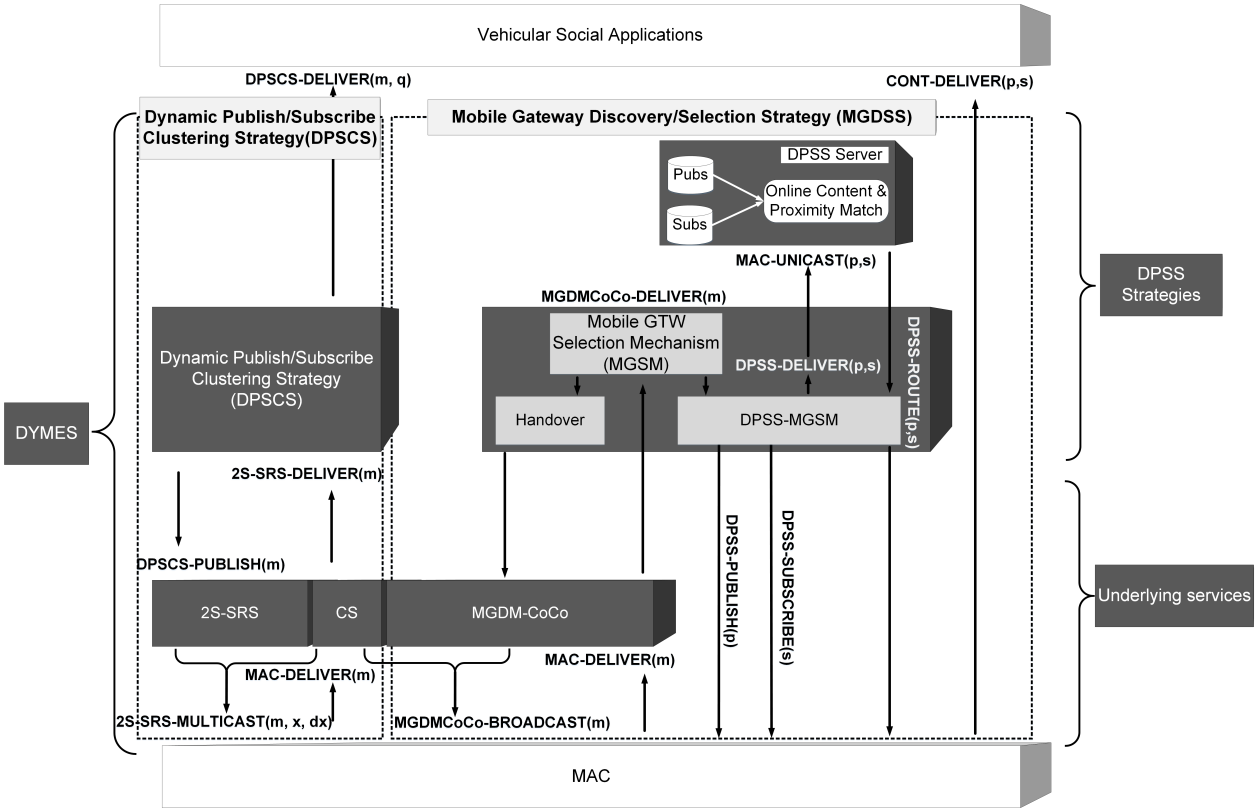

Figure 5.2 DYMES Architecture

- **2 Architecture**

DPSCS-2S-SRS's architecture is illustrated in Figure 5.2. When a commuter wants to build a stable, efficient and self-updated community, it publishes its single message through the DPSCS-PUBLISH primitive, that is multicasted throughout the network via the 2S-SRS-MULTICAST(m,x,Δx) primitive. Upon the delivery of the published message using the 2S-SRS-DELIVER(m) primitive, the commuter's single publication is self-updated at each intermediate node, matched against the intermediate nodes' current subscriptions, and then is forwarded by stable and farther nodes using the *2Shot-Stable Routing Service* (2S-SRS). The resulting match is delivered through the DPSCS-DELIVER primitive.

We first present the 2S-SRS that is underlying our dynamic publish/subscribe clustering strategy (see Figure 5.2).

Two Shot Stable Routing Service (2S-SRS). The two shot stable routing service is a stable wait and count mechanism that aims at efficiently and reliably disseminating the commuter's single persistent message in a predefined geographical area. It is based on stability metrics and on node proximity, and has two primitives:

- 2S-SRS-MULTICAST($m, x, \Delta x$) - disseminates the commuter's current contextual information encompassed in one message m to all its neighbors located within a range Δx of its position x. The commuter's message $m = \langle loc,\ vel,\ dir,\ t,\ \Delta t,\ RET_m \rangle$ includes its current location, velocity, direction, the TTL of the message that represents the lifetime of the message, and a stability parameter represented by the Route Expiration Time (RET) (to be described later), that is set to a large value ($RET{=}\infty$) the first time the message is sent by the commuter.

- 2S-SRS-DELIVER(m) - works as a callback and signals the reception of a message m to all the commuter's neighboring vehicles.

The message retransmission procedure occurs upon the delivery of the message m by the commuter's neighbors through the 2S-SRS-DELIVER(m) primitive defined above. When a message is received for the first time, it is scheduled before being forwarded via stable and farther nodes. Each receiver sets a global waiting time that depends on 1) stability metrics and on 2) node proximity. The more stable and farther is the receiver from the sender, i.e., closer to two boundary points, the shorter is its waiting time. The receivers cancel their scheduled message retransmission and increment the counter of the message m once they receive a similar copy of m. The message m is removed from the list of received messages once its counter exceeds a predefined threshold.

- **Stability metrics**

Stability metrics are used by each receiver to predict its link expiration time with the sender. The stability of links between the sender and the receivers is defined by the *Link Expiration Time* (LET) and the *Route expiration Time* (RET) metrics. The LET defines the expiration time of the link between the sender and its next hop receiver. Whereas, the RET defines the expiration time of the link that expires first along a route composed of multiple (n-1) links. According to (Su et al., 2001), if we consider two adjacent vehicles i and j with coordinates (x_i,y_i) and (x_j,y_j), a transmission range R, moving with speeds v_i and v_j in directions θ_i and θ_j, respectively, the estimated LET is

$$LET = \frac{-(ab + cd) + \sqrt{(a^2 + c^2)R^2 - (ad - bc)^2}}{a^2 + c^2} \tag{5.1}$$

where, $a = v_i \cos\theta_i - v_j \cos\theta_j$, $b = x_i - x_j$, $c = v_i \sin\theta_i - v_j \sin\theta_j$, $d = y_i - y_j$. The stability of a link is high $LET_{ij}{=}\infty$, if and only if, both vehicles i and j move with the same speed $v_i{=}v_j$ and in the same directions $\theta_i{=}\theta_j$. The *Route Expiration Time (RET)* consists of the

link that expires first in a route and is computed as follows:

$$RET_{n-1} = \min\{LET_{j,j+1}\}, j = 1....n-1 \tag{5.2}$$

Since each commuter is interested in building a stable interest-based community, the published message should be forwarded by the most stable receivers. In fact, upon the reception of the message m, each receiver computes the expiration time (LET) of its link with the sender using the contextual information inserted into the sent message m. Then, each receiver compares its computed LET value with the RET value inserted into the message. If the receiver's computed LET value is less than the value of the RET inserted into the message, then, the receiver's computed LET value will replace the message's RET value and a stability schedule will be established to forward the message. The more stable is the link between the sender and the receiver, the shorter is the stability schedule. We use the exponential stability function defined in (Barghi et al., 2009) in our global waiting time equation.

$$Stab(s,r) = 1 - \exp(\frac{-2LET_{sr}}{RET_m}) \tag{5.3}$$

This function responds to the above requirements, i.e., the larger is the LET value, the closer *Stab* is to 1 and the shorter will be our stability waiting time function.

$$WT(Stab(s,r)) = (1 - Stab(s,r)).mwt \tag{5.4}$$

where *mwt* is the maximum waiting time, i.e., maximum retransmission delay. It is worth noting that there could be many stable receivers with equal LET values. This implies that the message will be retransmitted by many forwarders which increases the message overhead and causes broadcast storms (Ni et al., 1999) over the distributed considered network. In this case, it is crucial to combine the stability waiting time with a function that optimizes and reduces the number of the forwarding candidates. The latter depends on the proximity of nodes to two boundary points.

- **Node proximity**

Unlike DPSCS-2S-SRS, the authors in (Barghi et al., 2009) have used the stability function mentioned above in Equation 5.3 and a progress function (Garbinato et al., 2010b) in order to spread beacon messages during their static gateway discovery process. The progress function they have used allows nodes that are farther from the sender to forward beacon messages in their considered network scenario. However, and contrary to their progress function mechanism, we modified a wait and count mechanism (Garbinato et al., 2010b) and adapted it to

the objective of our work, that is to disseminate the commuter's single message in the two opposite directions of the highway, in order to build efficient and reliable clusters, without the use of any control or beacon messages. Our mechanism allows each receiver to schedule a waiting time depending on its proximity to two boundary points located on the sender's transmission range. These two boundary points are computed once the message is sent via the publisher or the forwarder. They are mainly used to expand the spreading area of the single sent message in the two opposite directions of the highway, and also to reach more interested subscribers. The closer is the receiver to one of these two boundary points, the shorter is its waiting time. Moreover, the forwarding process is restricted to receivers located outside of the *Silent Zone* (SZ), as illustrated in Figure 5.3. The second proximity function of our global waiting time function is

$$Prox(r,b) = \frac{dist(r,b)}{R} \tag{5.5}$$

where *dist(r,b)* is the distance between receivers and one of the two boundary points. R is the sender's or the forwarder's transmission range. The proximity waiting time is defined as follows

$$WT(Prox(r,b)) = Prox(r,b).mwt \tag{5.6}$$

The global waiting time equation is as follows

$$WT_{glob} = [\frac{1}{\gamma} + \frac{1 - Stab(s,r)}{Prox(r,b)} - 1]\gamma.Prox(r,b).mwt \tag{5.7}$$

$$WT_{glob} = [\frac{1}{\gamma} + \frac{R.\exp(\frac{-2LET_{sr}}{RET_m})}{dist(r,b)} - 1]\frac{\gamma.dist(r,b)}{R}.mwt \tag{5.8}$$

where $\gamma \in]0,1[$ and $WT_{glob} \in [0, mwt]$. WT_{glob} represents a tradeoff between the link stability of senders and receivers, and the distance between receivers and two predefined boundary points. γ helps to define the best message's forwarder candidates whose stability functions are higher and whose proximity functions are lower. The proper value of γ is defined through simulations.

Figure 5.3 illustrates the functioning of our DPSCS-2S-SRS strategy. When Bob publishes a single and persistent message m in a predefined range, two boundary points are computed based on its location and its transmission range. Once Bob's neighbors deliver this message for the first time, they check if they are located in the silent zone, if not, they schedule each a waiting time. The latter depends on the stability of their links with Bob, and also on their proximity to the two precomputed Bob's boundary points. Alice, Carol, Mike and

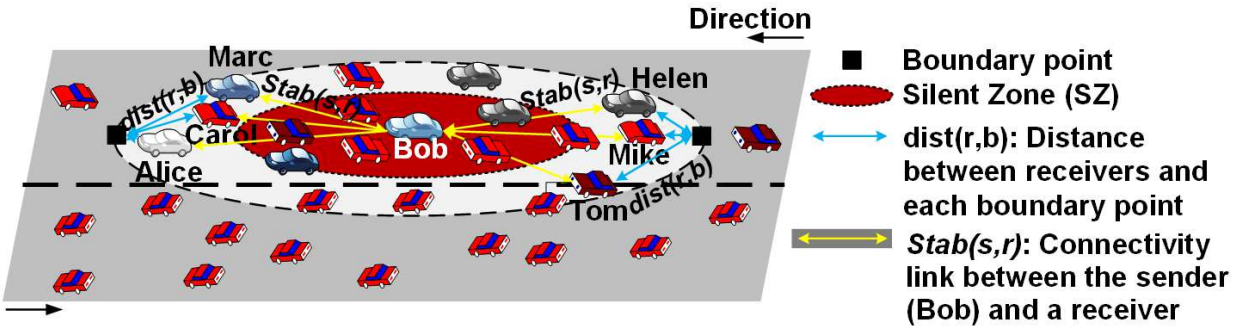

Figure 5.3 DPSCS-2S-SRS illustration

Helen compute their LET values with Bob and compare them with the RET inserted in the Bob's message. Their smallest LET values will replace the message's RET value before being forwarded. In the meantime, they compute their distance with one of the two boundary points located on Bob's transmission range, they schedule their proximity waiting time and add it to their stability waiting time. Since Mike and Alice are closer to one of the two boundary points, they can be considered as the best forwarder candidates, if and only if, their link's stability with Bob is high. Meanwhile, the neighbors of Bob compute his current location and direction based on the contextual information inserted in his message. They perform an online matching operation that aims at comparing their current computed Bob's contextual information with their current created queries. Once the match is positive, subscribers deliver Bob's message and join Bob's community.

- **3 Algorithm**

Algorithm 4 presents the DPSCS-2S-SRS strategy. It can be decomposed in four elements: Publishing, Subscribing, Refreshing and Matching.

Algorithm 4: DPSCS-2S-SRS

1: **uses 2 Shot-Stable Routing Service** 2S-SRS
2: **init**
3: $myMsgs, myQueries, matches \leftarrow \emptyset$

4: *To execute* DPSCS-PUBLISH(m) :
5: $myMsgs \leftarrow myMsgs \cup m$ {*adds m to own msg list*}
6: 2S-SRS-MULTICAST($m, x, \Delta x$)

7: *To execute* DPSCS-SUBSCRIBE(q) :
8: $myQueries \leftarrow myQueries \cup q$ {*adds q to own query list*}

9: DPSCS-DELIVER(m,q) *occurs as follows:*
10: **upon** 2S-SRS-DELIVER(m) **do**
11: **for all** $q \in myQueries$ **do** {*for all the host's queries*}
12: $t \leftarrow$ GETCURRENTMILLIS() {*gets the time*}
13: $l_m \leftarrow m.loc$ {*gets message m's origin location*}
14: $V_m \leftarrow m.vel$ {*gets message m's origin velocity*}
15: $l_m(t) \leftarrow V_m * t$ {*gets message m's current location since the publisher keep the same speed*}
16: $\theta_m \leftarrow \arctan(\frac{y_{m_i} - y_{m_{i-1}}}{x_{m_i} - x_{m_{i-1}}})$ {*gets message m's current movement direction*}
17: **if** DIRMATCH(m, q) **and** PROXMATCH(m, q) **and**LETMATCH(m, q) **then**
18: **if** $\langle m, q \rangle \notin matches$ **then** {*if the match is new*}
19: DPSCS-DELIVER(m, q) {*delivers the matching message*}
20: $matches \leftarrow matches \cup \{\langle m, q \rangle\}$ {*adds the match and join the group*}

21: **task** Refresh
22: **repeat every** $\Delta\tau$
23: **for all** $m \in myMsgs$ **do** {*for all created messages*}
24: **if** $m.\Delta t \neq \infty$ **then**
25: $m.\Delta t \leftarrow m.\Delta t - \Delta\tau$ {*decreases the lifetime*}
26: **if** $m.\Delta t \leq 0$ **then** {*if m is obsolete*}
27: $myMsgs \leftarrow myMsgs \setminus m$ {*removes m*}

1) Publishing (lines 4-6): To publish a message m, the DPSCS-PUBLISH(m) is called with m as parameter. Then, m is added to the $myMsgs$ set and is multicasted via the 2S-SRS-MULTICAST($m, x, \Delta x$) primitive around the publisher's position x in a predefined geographical range Δx. The message m is removed by the refresh process once its lifetime expires (line 27).

2) Subscribing (lines 7-8): To join a vehicular social community, each vehicle creates a subscription q, adds it to the $myQueries$ set, and then passes it as argument to the DPSCS-SUBSCRIBE(q) primitive.

3) Refreshing (lines 21-27): The refresh task is executed every $\Delta\tau$ to check the message's persistency that decreases with time. Once it is elapsed, the clustering process stops (line 27).

4) Matching (lines 9-20): The matching process is initiated at the DPSCS layer, upon the delivery of a message m, for the first time, by a subscriber through the 2S-SRS-DELIVER(m) primitive. The latter is responsible of scheduling a global waiting time that depends on both the subscribers' stability metrics and their proximity to two boundary points, as it is described above. During the matching process, each subscriber computes the current position of the publisher since the latter does not change its velocity (line 15). Moreover, each subscriber computes the current direction of the publisher (line 16) and executes three match operations:

- The *direction match*, DirMatch(m, q): consists of checking if the current subscriber's direction is the same as the current publisher's direction, i.e., $\theta_q = \theta_m$.

- The *proximity match*, ProxMatch(m, q): consists of checking if the current distance between the publisher and the subscriber is less than $k.R$, $|l_m(t)$ - $l_q| \leq k.R \leq |\Delta x|$, where $k \in \mathbb{N}$. $k.R$ is delimited by Δx, i.e., the space in which the message m has been propagated through the 2S-SRS-MULTICAST($m, x, \Delta x$) primitive.

- The *stability match*, LetMatch(m, q): aims at checking if the current link connectivity between the publisher and the subscriber is in its maximal stability, i.e., $v_q = v_m$

The match is positive, if and only if, it satisfies the three match operations. It is then delivered through the DPSCS-DELIVER(m, q) primitive and the subscriber joins the publisher's community (line 20).

5.1.2 Mobile Gateway Discovery/Selection Strategy in Hybrid VSNs

Once commuters are clustered in a dynamic interest-based community, they are eager to socialize with each other and to connect to the internet. To do so, commuters discover their

neighborhood in order to cooperatively and effectively control their current highway travel conditions. Then, they select a minimum number of appropriate gateway candidates that will link them to the internet. Upon the end of this process, they start sending their publications and subscriptions to the infrastructure via the most stable selected gateway. An online matching strategy is executed at the infrastructure and is in charge of updating the commuters' subscriptions and sending positive matches to the subscribers through stable links. Commuters perform a handover mechanism once they start losing their optimal connection with the selected gateway candidate.
In the following, we present the operating of the Mobile Gateway Discovery Mechanism (MGDM), then the operating of the Mobile Gateway Selection Mechanism (MGSM).

- **1 Mobile Gateway Discovery Mechanism**

We first describe the concept of MGDM, then we present its architecture and we detail its algorithm.

- **Concept**

MGDM is based on a novel *Context-aware Coding* service (CoCo) that allows each commuter to efficiently and reliably explore its neighborhood through broadcasting its contextual information using a single message, and without flooding the network with beacons or control messages. Two mechanisms are combined in MGDM-CoCo in order to provide high efficiency and reliability in highly dense environments. It is based on 1) network coding (Fragouli et al., 2008) that aims at reducing the overhead of multiple messages, and on 2) *two shot stable routing service* (2S-SRS), described above (5.1.1), and that aims at reducing the overhead of a single message and at increasing its reliability over the network. It is worth noting that our proposed mechanism is devised for any kind of vehicles' distribution and is not only restricted to a specific kind of networks, i.e., vehicular platoons (Holzer et al., 2011).

- **Architecture**

The architecture of MGDM-CoCo is illustrated in Figure 5.2. Each commuter retrieves its current contextual information using the *contextual sensor* (CS) component, then it broadcasts its message through the MGDMCoCo-BROADCAST primitive to the nodes located in its transmission range. The commuters' messages are received through the MGDMCoCo-DELIVER primitive.

Figure 5.4 describes the operating of MGDM-CoCo with four senders. First, Alice and Tom set the size of their silent zones and associate two boundary points to their messages a and

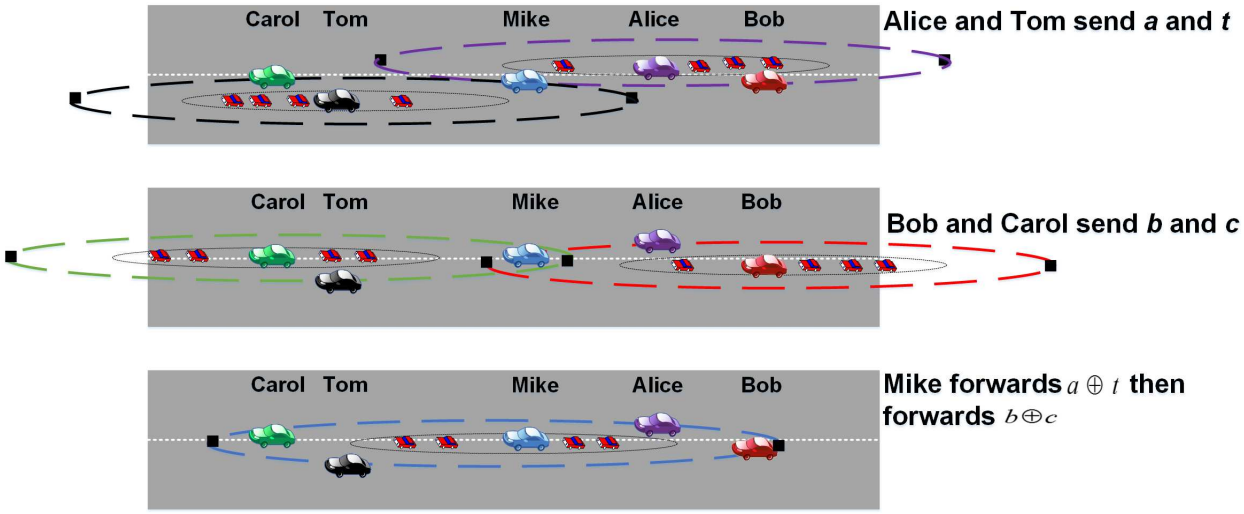

Figure 5.4 MGDM-CoCo illustration

t before they broadcast them in their transmission ranges. Their neighbors Bob, Mike and Carol deliver their messages and check if they are located inside the SZ. If not, they schedule each a waiting time that depends on their proximity to the sender's two boundary points. Their waiting times depend also on their stability metrics with the senders. The closer they are to one of the two predefined boundary points and the more stable their links are with the senders, the shorter are their waiting times. Mike's waiting time is shorter since he is much closer to the boundary points of Alice and Tom and he is located between the two senders, i.e., his stability with Alice and Tom is much higher than Bob and Carol. Then, Carol and Bob broadcast their messages c and b. Again, Mike's waiting time is shorter since his links with the senders are stable and also because he is closer to the two predefined boundary points of Carol and Bob. The senders' neighbors located inside the SZ only deliver the sent messages and do not forward them.

Finally, when Mike's waiting time expires, he mixes a and t into one message $a \oplus t$ and broadcasts it to its neighbors. Alice and Bob will retrieve t by decoding $a \oplus a \oplus t$ and Bob will cancel his scheduled retransmission of a. Meanwhile, Carol and Tom will retrieve message a by performing $t \oplus a \oplus t$ and Carol will cancel her scheduled retransmission of t. During this time and again upon the expiration of Mike's waiting time, he combines b and c into one message $b \oplus c$ and broadcasts it in its transmission range. Alice and Bob will retrieve c by decoding the coded Mike's message $b \oplus b \oplus c$ and Alice will cancel her scheduled retransmission of b. Similarly, Tom and Carol retrieve b by performing $c \oplus b \oplus c$ and Tom will cancel his programmed forward of c.

The main difference between our proposed MGDM-CoCo mechanism and BroadTrip (Holzer et al., 2011) is that MGDM-CoCo is designed to operate in highly dense environments and is not only restricted to a specific simple class of vehicular platoons (Holzer et al., 2011). Moreover and contrary to BroadTrip, forwarders are selected based on their links' connectivity with the senders, and also on their proximity to two predefined boundary points located on the transmission ranges of the senders. In addition, the forward process is only allocated to receivers located outside of the silent zone. Furthermore, our proposed mechanism provides the opportunity for each commuter to control the current state of the congested traffic during rush hours. In fact, commuters do not have to wait for a reply from a source vehicle, as it is mentioned in (Benslimane et al., 2011a), they instead discover their neighborhood and select an adequate gateway in order to connect to the infrastructure, simultaneously, and without flooding the network.

- **Algorithm**

Algorithm 5 presents the MGDM-CoCo mechanism. It can be decomposed in five elements: the MGDSCoCo-broadcast and the MGDSCoCo-deliver primitives, the refresh task and two important functions, namely GetSchedule and GetBoundaryPoints.

Algorithm 5: MGDM-CoCo

```
1: uses MAC, CS
2: init
3:    schedule, counter ← ⟨0, 0, ...0⟩ {sets of schedules and counters indexed by message id}
4:    M, RT ← ∅   {set of messages to handle and a routing table in which routes to nodes are managed}
5:    RET ← RET_max   {The Route Expiration Time is initialized to a large value by the sender}
6:    SZ ← ...   {Size of the Silent Zone estimated by the sender}

7: To execute MGDSCoCo-broadcast(m) :
8:    m.loc ← GetLocation()   {sets location of m's sender}
9:    m.vel ← GetVelocity()   {sets velocity of m's sender}
10:   m.dir ← GetDirection()   {sets direction of m's sender}
11:   m.ret ← RET   {sets RET value of the sender}
12:   MAC-broadcast(m)

13: MGDSCoCo-deliver(m) occurs as follows:
14:   upon MAC-deliver(msg) do
15:      msgs ← decode(msg, counter)   {decodes msg}
16:      for all m ∈ msgs do   {for all decoded messages}
17:         if counter_m == 0 then   {if m is received for the first time}
18:            RT ← RT ∪ m   {Stores node information in the routing table}
19:            M ← M ∪ m   {adds m}
20:            counter_m ← counter_m + 1   {adds m}
21:            MGDSCoCo-deliver(m)   {delivers m}
22:            l_m ← m.loc   {gets message m's sender's location}
23:            l_r ← GetLocation()   {gets the current position of the receiver}
24:            if l_m l_r ≥ SZ then   {if m is outside the silent zone}
25:               schedule_m ← GetSchedule(m)   {computes the global schedule}
26:         else
27:            replace(m_RT, m_RT_received)   {updates the routing table}
28:            M ← M \ m   {removes m from the handled messages}

29: function GetSchedule(m)
30:   distance, LET ←⊢
31:   stab ← 0
32:   for all b ∈ GetBoundaryPoints(m.loc) do   {gets the closest boundary point}
33:      if (l_r b < distance) ∨ (distance =⊢) then
34:         distance ← l_r b
35:   if (LET_m,r > LET) ∨ (LET =⊢) then   {gets the stable node}
36:      LET ← LET_m,r
37:   return schedule(LET, distance)   {returns the global schedule in function of the distance and the stability}

38: function GetBoundaryPoints(b)
39:   boundaries ← ∅
40:   for all 0 ≤ i ≤ 1 do   {for all two boundary points}
41:      b_i ← ⟨b_x + cos(π ∗ i) ∗ R, b_y + sin(π ∗ i) ∗ R⟩   {creates two boundary points}
42:      boundaryPoints ← boundaryPoints ∪ b_i   {adds b_i to the list of boundary points}
43:   return boundaryPoints   {returns the boundary points}

44: task Refresh
45:   repeat every Δτ
46:   t ← GetCurrentMillis()   {gets the time}
47:   for all (m_i, m_j) ∈ M|i ≠ j ∧ schedule_m_i < t ∧ schedule_m_j < t do
48:      if (GetLocation() ∈ l_m_i l_m_j) ∧ l_m_i l_m_j ≤ 2R then
49:         msg ← m_i ⊕ m_j   {mix the messages that satisfy the condition above}
50:         m_i.loc ← GetLocation()   {sets the location of m_i's sender}
51:         m_i.vel ← GetVelocity()   {sets the velocity of m_i's sender}
52:         m_j.loc ← GetLocation()   {sets the location of m_j's sender}
53:         m_j.vel ← GetVelocity()   {sets the velocity of m_j's sender}
54:         RET_msg ← min(RET_m_i, RET_m_j, LET_rm_i, LET_rm_j)   {sets the RET of msg}
55:         MAC-broadcast(msg)   {forward the coded message}
56:         M ← M \ m_i, m_j   {removes the messages m_i, m_j}
57:   for all m ∈ M/schedule_m < t do
58:      m.loc ← GetLocation()   {sets the location of m's sender}
59:      m.vel ← GetVelocity()   {sets the velocity of m's sender}
60:      RET_m ← min(RET_m, LET_rm)   {sets the RET of m}
61:      MAC-broadcast(m)   {forwards the message}
62:      M ← M \ m   {removes m}
```

1) Broadcasting (lines 7-12): users initiate the broadcasting through the MGDMCoCo-BROADCAST primitive with a single message m as parameter. The location, the velocity, the direction and the RET values are added to m before it is broadcasted over the MAC layer. When users create their single message they set the RET to a large value.

2) Delivering (lines 13-28): When the message *msg* is delivered through the MAC-DELIVER primitive, it is decoded via the DECODE function, which produces the set *msgs* of decoded messages contained in *msg*. The set *msgs* is the result of the decode operation that consists of decoding the location, the velocity and the direction while keeping the RET value of the message *msg*. Then, for each of the messages m in *msgs*, if m is received for the first time, it is delivered, added to the set of messages to retransmit M, its routing information (source node, RET) is stored in the node's routing table RT and a schedule for its retransmission is set using the GETSCHEDULE function, if and only if, the receiver is outside of the silent zone (lines 24-25). The GETSCHEDULE function computes a schedule using the contextual information contained in m. If other similar copies of m are received during the m's scheduled waiting time, the m's counter is incremented and its routing information is updated. The message m is forwarded once its waiting time expires and is removed from the set M if its counter exceeds a predefined limit (line 28).

3) Schedule computing function (lines 29-37): The GETSCHEDULE function is called if the receiver is located outside of the silent zone. This function computes a schedule that depends on the distance between the receiver's location and two boundary points located on the sender's transmission range. In addition, this function allows the receiver to compute its link expiration time with the sender using the contextual information contained in m. The closer is the receiver to one of the sender's boundary points and the stable it is with the sender, the shorter is its global waiting time.

4) Boundary points computing function (lines 38-43): The GETBOUNDARYPOINTS function computes the sender's two boundary points based on its location and its transmission range.

5) Refreshing (lines 44-62): The refreshing task is executed every $\Delta\tau$ and is in charge of forwarding messages. It checks the schedule of each pair of the received messages in the set M. If the checked pair is ready for retransmission, the task checks if the messages are originally sent from opposite sides of the forwarder. If this is the case, it mixes the location, the velocity and the direction contained in the received messages, computes the forwarder's LET values with the two senders, compares the computed LET values with the RET values inserted into the received messages, and finally, adds the minimum resulting RET value, the forwarder's location and velocity to the coded message. It then, retransmits

the coded message through the MAC layer and removes the messages from the set M. Next, the refreshing task retransmits ripe messages that cannot be mixed with others. It compares the computed LET value, between the forwarder and the senders of these messages, adds the minimum RET value, the location and the velocity of the forwarder to the ripe messages and removes them from the set M after the forward.

- **2) Mobile Gateway Selection Mechanism**

We first discuss and describe the concept of our proposed Mobile Gateway Selection Mechanism (MGSM), then we present and detail its architecture and its algorithm.

- **Concept**

MGSM allows commuters to select a minimum number of gateway candidates able to link them with the infrastructure, based on the network information they gathered in the MGDM-CoCo mechanism. The adequate gateway candidate is selected based on its location in the considered cluster, and also based on its stability metrics with commuters. In other words, the node located at the center of the cluster and the ones located at one hop of this central node and with which commuters are connected through stable links, i.e., large RET values, are the best gateway candidates to link commuters with the infrastructure.

The stability of the commuters' links with the selected gateway is an important metric that controls the execution of two mechanisms, namely the Dynamic Publish/Subscribe Strategy based on MGSM (DPSS-MGSM) and the handover mechanism.

DPSS-MGSM aims at sending the commuters' dynamic publications/subscriptions to the infrastructure via the selected gateway, if and only if, the commuters are strongly connected to the selected gateway. An online matching strategy is executed at the server side (infrastructure), and aims at updating the received subscriptions and at sending positive matches to the corresponding subscribers.

The handover mechanism occurs upon the expiration of the route expiration time of commuters and the selected gateway(s). It aims at rediscovering the commuters' new neighborhood using the MGDM-CoCo mechanism and at selecting a new gateway candidate.

- **Architecture and Algorithm**

The architecture of MGSM is illustrated in figure 5.2, and its algorithm is described in Algorithm 6. This task occurs once commuters stop receiving messages through the MGDMCoCo-DELIVER primitive. They first extract the location information from the received messages,

and define the central node and its one hop neighbors in their considered cluster through the GetCentralNode function. Then, they extract the RET values associated with the received central nodes' messages stored in their routing tables (line 5). Finally, they select the central gateway candidate whose message arrived from a stable route with a larger route expiration time.

Algorithm 6: MGSM

```
1:  task Gateway Selection
2:    upon MGDMCoCo-deliver(m) do
3:      for all m ∈ M, RT do
4:        GetCentralNode(l_m)        {ranks the messages received depending on their
          locations}
5:        GetRET_{m_center}
6:        if RET_{m_center} ≥ RET_th then
7:          myPubs ← myPubs ∪ p                          {creates a publication}
8:          mySubs ← mySubs ∪ q                          {creates a subscription}
9:          DPSS-publish(p)                  {unicasts p to the selected gateway}
10:         DPSS-subscribe(s)                {unicasts s to the selected gateway}
11:       else
12:         MGDMCoCo-broadcast(m)              {executes the handover process}

13: upon DPSS-deliver(p, s) do
14:   MAC-unicast(p, s)                      {communicates with the infrastructure}

15: proc OnlineMatch(p,q)
16:   upon MAC-deliver(p, q) do
17:     for all q ∈ myQueries ∧ p ∈ myMsgs do   {for all the hosts' publications and
        subscriptions}
18:       t ← GetCurrentMillis()                                   {gets the time}
19:       l_{p,q} ← (p, q).loc          {gets message p's and query q's origin location}
20:       V_{p,q} ← (p, q).vel          {gets message p's and query q's origin velocity}
21:       l_(p, q)(t) ← V_{p,q} ∗ t   {gets message p's and query q's current location}
22:       θ_{p,q} ← arctan(y_{p,q} / x_{p,q})   {gets message m's current movement direction}
23:       if LetMatch(p, q) and ProxMatch(p, q) then
24:         matches ← matches ∪ {⟨p, q⟩}                                {adds the match}
25:         Cont-deliver(p) on q.h          {the delivery on the subscribing node}
```

DPSS-MGSM occurs, if and only if, the commuters' RETs with the selected central node are above a predefined threshold (line 6). Commuters are then allowed to create publications/subscriptions and send them to the selected gateway via the DPSS-publish and the DPSS-subscribe primitives, while reversing the routes stored in their routing tables (lines 6-10). Once the selected gateway receives the commuters' publications and subscriptions via the DPSS-deliver primitive, it forwards the received messages and queries to the infras-

tructure using the MAC-UNICAST primitive (lines 13-14).
An online matching strategy is executed at the server side (lines 15-25). It aims at updating the current locations and directions of the publishers and subscribers, and at checking if they are both located at the intersection of their publications' and subscriptions' spaces, and also if they are heading toward the same direction. Once a positive match is found, it is sent to the selected gateway that reverses the path from which the subscription arrived, and forwards the received positive matches to the corresponding subscribers.

The handover mechanism occurs as soon as the commuters' link stability with the selected gateway decreases and reaches a minimum value. At this moment, commuters re-execute the MGDM-CoCo mechanism using the MGDMCoCo-BROADCAST primitive in order to define a new stable gateway (line 12).

5.2 Performance Evaluation

In this section, we present the performance evaluation of DYMES. It is implemented in the Network Simulator NS3 (Riley and Henderson, 2010). SUMO is used for simulation of vehicular movements and environments (Behrisch et al., 2011). We conducted extensive simulation experiments to evaluate the performance of the two main strategies on which is based our system, namely the Dynamic Publish/Subscribe Clustering Strategy based on the 2S-SRS service (DPSCS-2S-SRS), and the Mobile Gateway Discovery/Selection Strategy based on the CoCo service (MGDSS-CoCo). SUMO is used to simulate a highway scenario with three lanes. All the functionalities have been implemented based on IEEE 802.11 MAC, with TwoRayGround propagation model for traffic propagation in the considered highway scenario. The performance of our system is evaluated in terms of efficiency and reliability metrics defined as follows:

- Efficiency is defined by the forward ratio, i.e., the number of nodes that forward a broadcast message m divided by the number of nodes in the network;

- Reliability is defined by the delivery ratio, i.e., the number of nodes that deliver the message divided by the number of nodes in the network.

An algorithm is said to be efficient and reliable, if it ensures that a minimum number of nodes retransmit the sent messages, and that all nodes in the network deliver the sent/forwarded messages.

5.2.1 Performance Evaluation of DPSCS-2S-SRS

Hereafter, we evaluate the performance of our proposed DPSCS-2S-SRS strategy. We have compared DPSCS-2S-SRS with DPSCS based on four different routing services, namely, Lifetime-Based Routing protocol (LBR) (Barghi et al., 2009; Benslimane et al., 2011c), Counter-Based Scheme (CBS) (Haas et al., 2006), flooding (FLOOD) and Power-Aware Message Propagation Algorithm (PAMPA) (Garbinato et al., 2010b), that have been reviewed in background and related work. Note that LBR is a combination of stability metrics and on PAMPA and is introduced in (Barghi et al., 2009; Benslimane et al., 2011c). We first find the exact value of γ through simulations, as it is stated in equations 5.7 and 5.8, then we analyze and evaluate the performance of our strategy by examining the impact of varying the nodes' density on the efficiency and the reliability of the considered vehicular scenario.

- **1) Simulation Parameters**

Table 5.1 shows the simulation parameters. The considered highway scenario is 2 km length with three curved lanes. The simulation time is 500 s. The vehicles and the network became steady after 150 seconds of the start of the simulation.

To simulate our strategy, a vehicle is selected randomly to publish a UDP packet of size 1000 bytes to its neighbors in a zone Δx of 500 m. We choose the maximum waiting time mwt to be 1 second, which proved to be a practical setting in (Garbinato et al., 2010b).

- **2) Simulation Results**

- **Defining the value of γ**

We simulated different scenarios in which we set the values of γ to 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, and 0.9, and we changed the density of the nodes.

Table 5.1 Configuration parameters for simulation of highway scenario

Parameters	Value
Type and length of road	Highway, 2 km
Number of lanes	Three lanes in the same direction
Maximum speed	10-35 m/s
Number of vehicles	100, 200, 300 and 400 vehicles
Transmission range	100 m
Message lifetime	8 s
Simulation time	500 s
warm-up period	150 s
Simulation runs	20

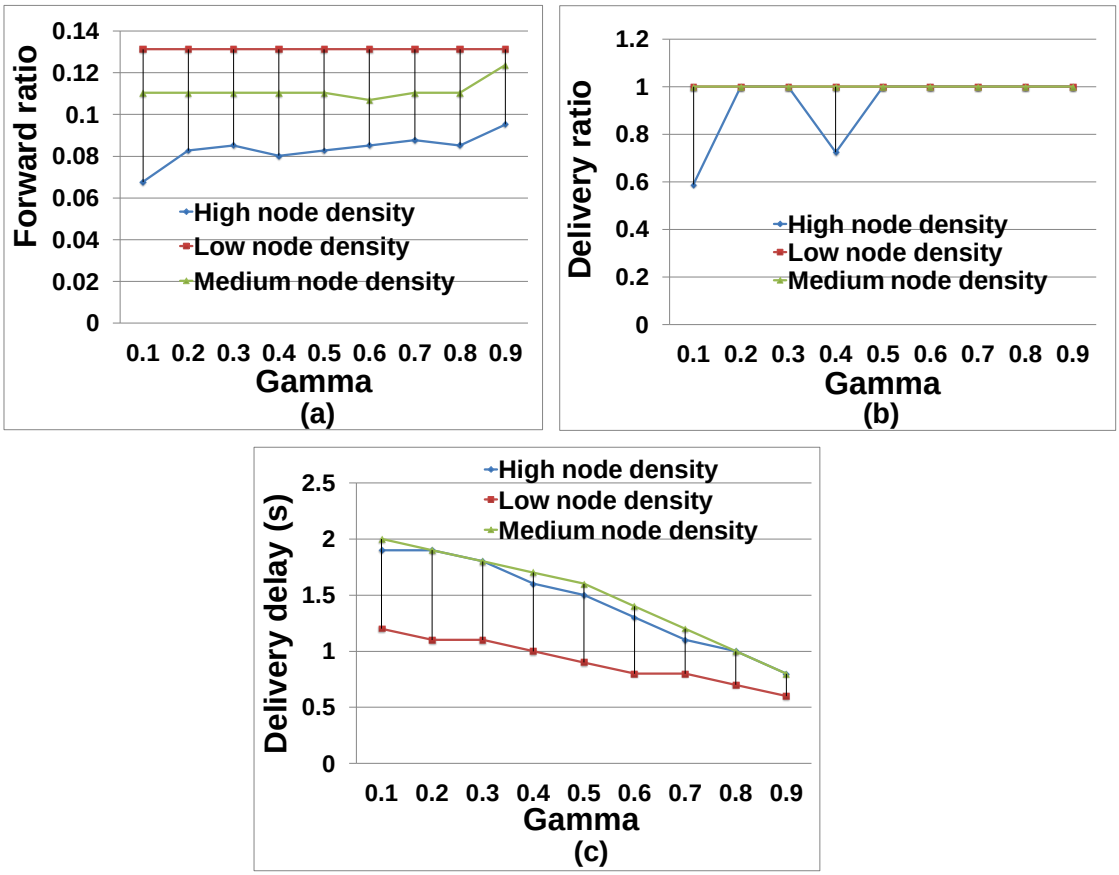

Figure 5.5 Performance of DPSCS-2S-SRS under different network settings to define γ - (a) Forward ratio, (b) Delivery ratio, (c) Delivery delay

Figure 5.5 shows the forward ratio, the delivery ratio and the delivery delay (the time elapsed between the broadcasting of a message by a source and the reception of this message by all vehicles in the network) for different values of γ with different node densities. We have varied the silent zone in each scenario in order to minimize, as much as possible, the number of forwarders. As it can be seen from the simulation results, the forward ratio increases in high and medium densities as long as we increase the value of γ, however, it keeps a steady value in the low node density scenario. Furthermore, all nodes in the medium and low density scenarios have received the sent message, however, the message was not received by all nodes in the high node density scenario for values of γ equal to 0.4 and 0.1. In addition, the delay decreases, as long as, we increase the value of γ.

The reason that stays behind this behaviour is that when γ is low, a big weight is given to the proximity function mentioned in equations 5.7 and 5.8. In fact, the receivers check if they are located inside the silent zone. If not, they establish a schedule that depends on their proximity to two boundary points located on the sender's/forwarder's transmission range. The latter increases the message's delivery delay in the three considered scenarios. More in line with the variation of the forward ratio. We see that it keeps a steady value in the low node density scenario since we have fixed the size of the silent zone, which is one of the main particularities of our proposed strategy. In fact, since there are only few nodes in the network (low density), the size of the silent zone ensures that all nodes deliver the message with the minimum overhead, and does not give the opportunity for any additional receiver to forward the sent message for the different values of γ.

On the other hand, the forward ratio in high and medium node densities increases starting of $\gamma = 0.8$. This is because this value of γ increases the weight of the stability function mentioned in equations 5.7 and 5.8, and decreases the weight of the proximity function. This pushes many receivers to deliver the message and to forward it using the stability schedule they established once they received the message. We can conclude that the stability function speeds up the delivery delay of the message, i.e., it decreases the message's delivery delay for $\gamma = 0.8$ and 0.9 in the three considered node densities.

Figure 5.5(b) shows that the message was not received by all nodes in the high node density scenario, for $\gamma = 0.1$ and 0.4. This is mainly due to the large number of receivers that contend for accessing the medium which causes collisions and decreases the delivery ratio.

Since we are interested in selecting forwarders that maximize both efficiency and reliability, while decreasing the delivery delay of the sent message, and on the basis of the simulation results and the conducted analysis, we can state that the best value for γ is equal to 0.8.

- **Evaluation**

In what follows, and after having defined the value of γ, we present the performance of DPSCS-2S-SRS compared with the performance of DPSCS-LBR (Barghi et al., 2009; Benslimane et al., 2011c), DPSCS-PAMPA, DPSCS-CBS and DPSCS-FlOOD, in terms of efficiency (forward ratio), reliability (delivery ratio), delivery delay and match ratio (the ratio of nodes whose current subscriptions positively match the published message).

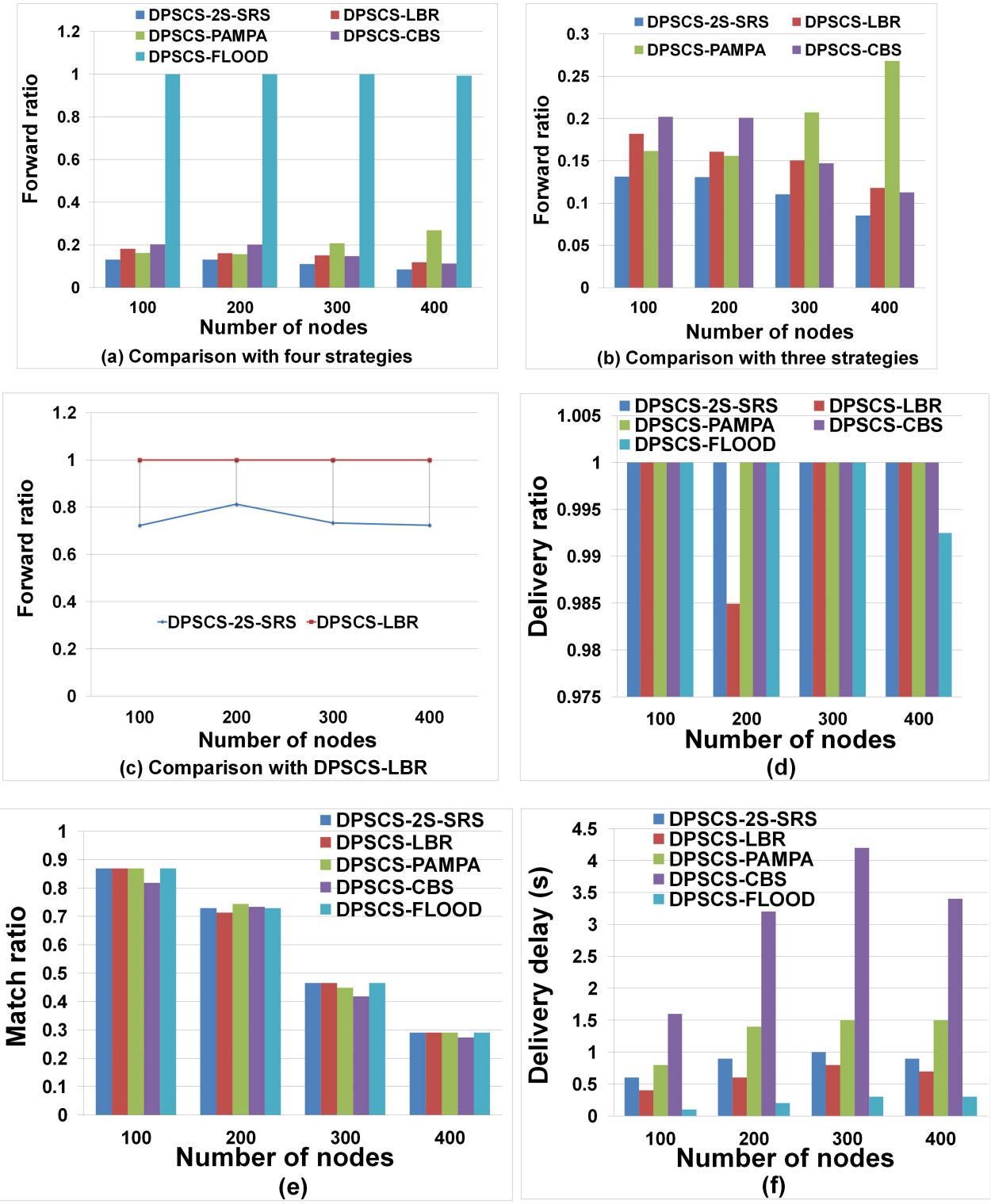

Figure 5.6 Simulation results of DPSCS-2S-SRS compared with four strategies - (a)-(c) Forward ratio, (d) delivery ratio, (e) match ratio and (f) delivery delay.

We see, in Figure 5.6, that the trends are similar for low, medium and high densities. In terms of reliability (see Figure 5.6(d)), all strategies reach a 100% delivery ratio for the different node densities, except DPSCS-LBR that only reaches 98% in the scenario composed of 200 nodes, and DPSCS-FLOOD that also fails to deliver the published message to all nodes in the high density scenario (400 nodes).

Figures 5.6(a)-(c) show the variation of the forward ratio with the number of nodes. In terms of efficiency, and as it can be seen from the simulation results, the forward ratio of DPSCS-2S-SRS decreases as long as the nodes' density increases. In fact, DPSCS-2S-SRS outperforms all other strategies for any network size and any density. In low node density (100 nodes), DPSCS-2S-SRS beats DPSCS-LBR by 27.8%, it beats DPSCS-PAMPA by 18.7%, it beats DPSCS-CBS by 35% and it beats DPSCS-FLOOD by 86.8%. In medium node density (200-300 nodes), DPSCS-2S-SRS outperforms DPSCS-LBR by 18.75% to 26.6%, it outperforms, DSPSCS-PAMPA by 16% to 46%, it outperforms DPSCS-CBS by 35% to 25% and it surpasses DPSCS-FLOOD by 86.9% to 88.9%. Finally, in high node density (400 nodes), DPSCS-2S-SRS beats DPSCS-LBR by 27%, it beats DPSCS-PAMPA by 68%, it beats DPSCS-CBS by 24.4% and it outperforms DPSCS-FLOOD by a factor of 9.

In terms of the delivery delay (see Figure 5.6(f)), DPSCS-2S-SRS succeeds to deliver the published message to all nodes in all network sizes and in all densities, in less than 1 second, with a reasonable forward ratio. We see that DPSCS-FLOOD delivers the published message in a very small delay in low and medium node densities (100-300 nodes), but, with an exponential forward ratio (100%), compared with other best strategies. This is due to the fact that all nodes in the different densities forward the message without establishing any schedule. However, DPSCS-FLOOD fails to reach 100% delivery ratio in the high node density (400 nodes), due to the large number of nodes that compete to access the medium, which causes collisions. The difference in the delivery delay between DPSCS-2S-SRS and the DPSCS-LBR is 0.2 second in low, medium and high densities (100, 300 and 400 nodes). In fact, DPSCS-LBR fails to deliver the published message to all nodes in the medium node density (200 nodes); only 98% of the nodes deliver the message in this scenario during 0.6 s, compared to 100% delivery rate for DPSCS-2S-SRS in the same scenario during a delay of 0.9 s.

The reason that stays behind the DPSCS-LBR's low delivery ratio, in this scenario (200 nodes), is due to the fact that the nodes' schedule depends on the links stability as well as the distance between sender/forwarders and receivers. Only stable farther nodes from the sender/forwarders retransmit the message. It may happen that depending on the nodes' schedules, several forwarder candidates contend to access the medium in order to retransmit

the message at the same time which causes collisions and decreases the message's delivery ratio and the delivery delay. On the other hand, DPSCS-2S-SRS outperforms DPSCS-LBR in terms of the number of retransmissions thanks to the presence of the silent zone that hides a large number of forwarders and only allows a minimum number of receivers to forward the message. This important characteristic combined with the stability function greatly impacts the performance of DPSCS-2S-SRS by increasing efficiency and reliability and decreasing the delivery delay.

Regarding the match ratio (see Figure 5.6(e)), and following the operating of DPSCS, each node delivers a positive match that is the result of three match operations described in 5.1.1. In Figure 5.6(e), the variation of the match ratio with the density of nodes are depicted for each DPSCS strategy. Figure 5.6(e) shows that the match ratio decreases as long as the density of nodes increases. This is due to the vehicles' movement distribution generated by SUMO. In fact, 81% to 86% of nodes deliver a positive match in the low node density scenario (100 nodes) compared with 41% to 71% in the medium node density scenario and 27% to 29% in the high node density scenario (400 nodes). The reason for these results is that, when subscribers deliver the message, they compute the current location and direction of the publisher (based on the received message). They use their current location, velocity and direction in the message filter and execute the three match operations, that do not produce a high number of positive matches, in the high and the medium node densities compared with the low node density. In fact, the current subscribers velocities, locations and directions follow a SUMO distribution, that react differently based on the considered number of nodes in each evaluated network density scenario, i.e., the large number of nodes is associated with quite different mobility values (location, velocity and direction) that do not always satisfy the three match conditions for all densities, all the time, and at the same time.

5.2.2 Performance Evaluation of MGDSS

Hereafter, we evaluate the performance of our proposed Mobile Gateway Discovery/Selection Strategy (MGDSS), that is based on MGDM-CoCo and MGSM. We have compared MGDSS-CoCo with MGDSS based on four different routing services, namely, BroadTrip (Holzer et al., 2011), Counter-Based Scheme (CBS) (Haas et al., 2006), flooding (FLOOD) and Power-Aware Message Propagation Algorithm (PAMPA) (Garbinato et al., 2010b), that have been reviewed in background and related work.

Since the MGDM-CoCo's waiting delay also depends on γ, we have defined the exact value of γ through simulations, following the same procedure described in Section 5.2.1. We found that $\gamma = 0.8$ is the best value that ensures a good level of efficiency and reliability throughout the considered scenarios.

The next step is to assess the performance of MGDSS-CoCo by examining the impact of varying the nodes' density on the efficiency and the reliability of the considered vehicular scenarios.

- **1) Simulation Parameters**

Table 5.2 is an extension of the simulation parameters presented in Table 5.1. The considered highway scenario is 2 km length with three lanes. There are two infrastructure units beside the highway which are located 1 km apart from each other. The transmission range of vehicles is 150 m, whereas, the transmission range of the base station is 1 km. The velocity and the density of nodes are set based on the different scenarios.

To simulate our strategy, all vehicles send a single UDP packet of size 1000 bytes. We choose the maximum waiting time mwt to be 1 second and we varied the size of the silent zone for each considered scenario. The average distance between vehicles varies between 8 and 10 meters.

The simulation lasts for 500 s. The first 150 s are used to initialize the simulations and to make sure that the vehicles and the network have become stable.

- **2) Simulation Results**

Hereafter, we present the performance of MGDSS-CoCo in terms of efficiency (forward ratio), reliability (delivery ratio) and the ratio of received matches (the ratio of nodes that received a positive match sent from the base station through the selected gateway).

First, we present the simulation results of MGDM-CoCo. Figure 5.7 contrasts the forward ratio and the delivery ratio while varying the number of nodes.

Then, we evaluate the performance of MGDSS-CoCo in terms of the number of forwarded messages and the ratio of the received matches. Figure 5.8 shows how increased density impacts the strategies' forward and delivery ratios.

Table 5.2 Configuration parameters for simulation of highway scenario

Parameters	Value
Maximum speed	10-20 m/s
Number of vehicles in clusters	40, 50, 60 and 70 vehicles
Number of RSUs	2
Distance between the RSUs	1 km
Simulation runs	20

- **Evaluation of MGDM-CoCo**

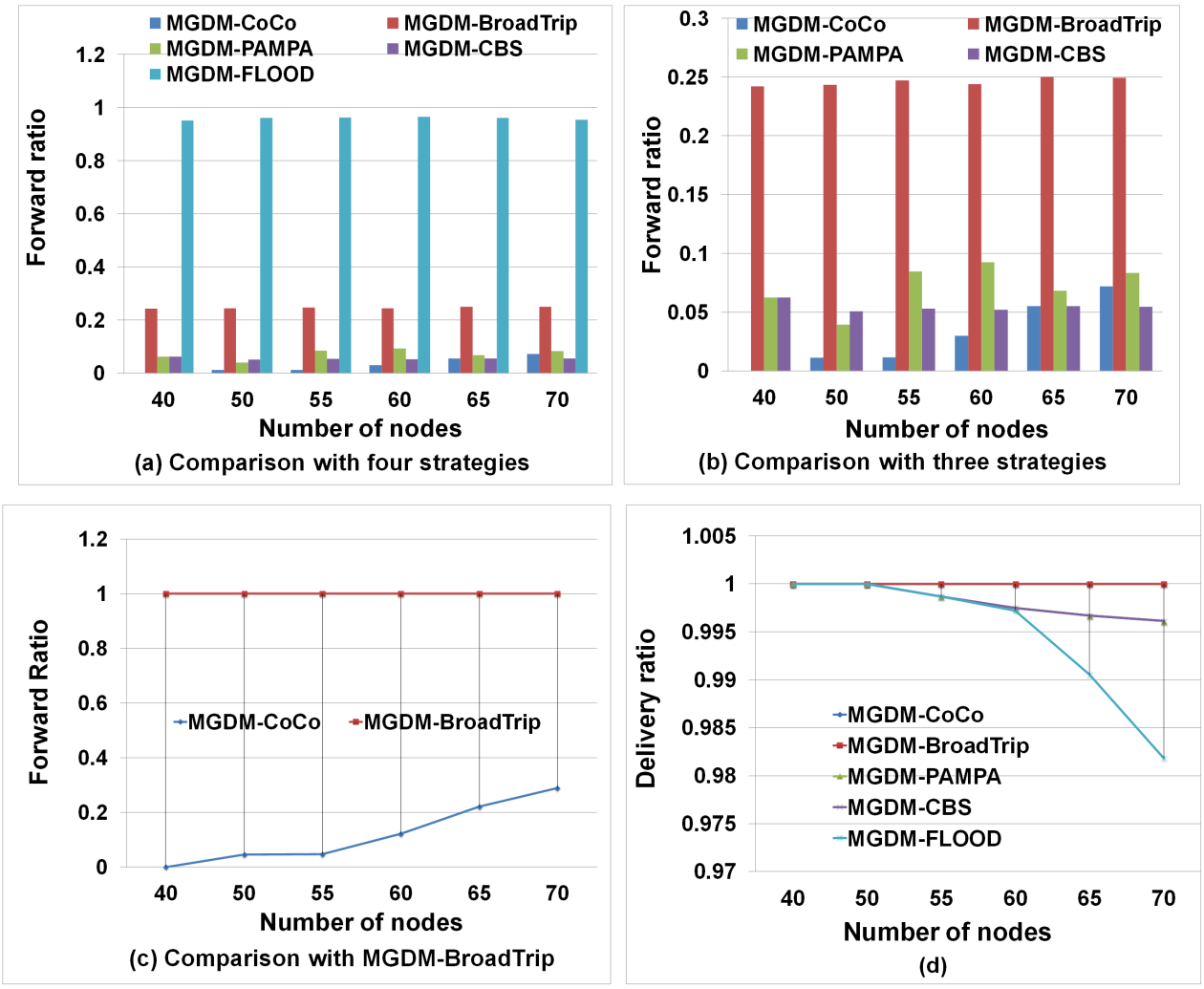

Figure 5.7 Simulation results of MGDM-CoCo compared with four mechanisms - (a), (b) and (c) Forward ratio, (d) delivery ratio.

Figures 5.7(a)-(c) show the variations of the forward ratio with the nodes' density where the distance between nodes varies between 8 and 10 meters. As it can be seen from the simulation results, the number of forwarders increases as long as we increase the density of nodes. On the other hand, and in terms of reliability, figure 5.7(d) shows that MGDM-CoCo and MGDM-Broadtrip reach 100% delivery rate for any network size and any density. Nevertheless, the performance of other mechanisms namely, MGDM-PAMPA, MGDM-CBS and MGDM-FlOOD decreases, as long as the network size and the nodes' density increases.

In low node density (40-50 nodes), all MGDM mechanisms reach 100% delivery rate. In fact, MGDM-CoCo succeeded to deliver all messages with 0% retransmissions (see Figure 5.7(a)). It beats MGDM-BroadTrip by 95% to 100%, it beats MGDM-PAMPA by 71% to 100%, it beats MGDM-CBS by 77% to 100% and it outperforms MGDM-FLOOD by 98% to 100%. In medium node density (55-60 nodes) and in high node density (65-70 nodes), we see that MGDM-CoCo surpasses MGDM-BroadTrip by 71% to 95%, while other mechanisms fail to deliver all broadcast messages.

The reason that stays behind 0% forward ratio in our proposed mechanism lies in the fact that all nodes in the considered node density (40 nodes) decode the coded messages, deliver them and they are all located inside the silent zone, so they do not establish a schedule to retransmit coded and single messages. They only decode, deliver messages and stay silent.

However, despite the fact that MGDM-BroadTrip also relies on network coding, it selects forwarders based on their distance from the senders. Farther they are from the senders, shorter are their waiting times. It is evident that increasing the nodes' density will drastically impact the forward ratio, in this mechanism, by increasing the number of forwarders.

Regarding MGDM based on PAMPA, CBS and FLOOD, we see that they fail to deliver all messages when the nodes' density increases. This is due to the large number of users that compete to access the medium at the same time, in order to forward the received messages upon the expiry of their schedules. The latter leads to a decrease in the number of delivered messages and causes collisions.

We can conclude that the network coding mechanism combined with the stability metrics, used in our proposed strategy, ensure a high delivery ratio. Moreover, the node proximity and the use of the silent zone restrict the forward process to only few nodes, which greatly optimizes the forward ratio as long as the nodes' density increases.

- **Evaluation of MGDSS-CoCo**

In the simulation settings of MGDSS-CoCo, half of the nodes, in each considered node density, create publications and subscriptions and send them to the selected gateway. In the following,

we evaluate the performance of MGDSS-CoCo, using these settings, and we compare it with MGDSS based on BroadTrip, PAMPA, CBS and FLOOD.

Figures 5.8(a) and (b) illustrate the simulation results of MGDSS-CoCo in terms of the forward ratio that includes the number of retransmitted messages in the MGDM mechanism, and the number of forwarded publications, subscriptions and matches in the MGSM mechanism. Figure 5.8(c) portrays the ratio of the received matches sent by the base station to the corresponding subscribers through the selected gateway.

In terms of efficiency, we see that the cost of our proposed strategy MGDSS-CoCo represents only 30% to 32% of the cost of MGDSS-FLOOD in low and medium node densities, and about 35% to 37% in high node densities (65-70 nodes).

Furthermore, MGDSS-CoCo beats its next comparable approach MGDSS-BroadTrip, in low and medium node densities, by 37% to 45% and surpasses the same strategy MGDSS-BroadTrip, in high node densities (65-70 nodes), by 35% to 36%.

In terms of the ratio of the received matches coming from the base station, we see that it varies depending on the variation of the delivery ratio in the MGDM mechanism (see figure 5.7(d)). That is, MGDSS-CoCo always reaches a high level of the delivered matches that varies from 40% to 50% in all network sizes and densities. This is because MGDM-CoCo ensures 100% delivery ratio in all considered scenarios and also because the routes to the selected gateway are known and available in the routing tables at each node.

MGDSS-BroadTrip follows the trend of MGDSS-CoCo and delivers 38% to 45% matches to the corresponding subscribers in all considered densities, whereas MGDSS-PAMPA, MGDSS-CBS and MGDSS-FLOOD succeed to deliver 45% to 50% matches to the corresponding subscribers in the low node density, since they reach 100% delivery rate in this case (see figure 5.7(d)). However, they fail to deliver a comparable ratio to the subscribers in medium and high node densities due to their low delivery ratio in the MGDM mechanism.

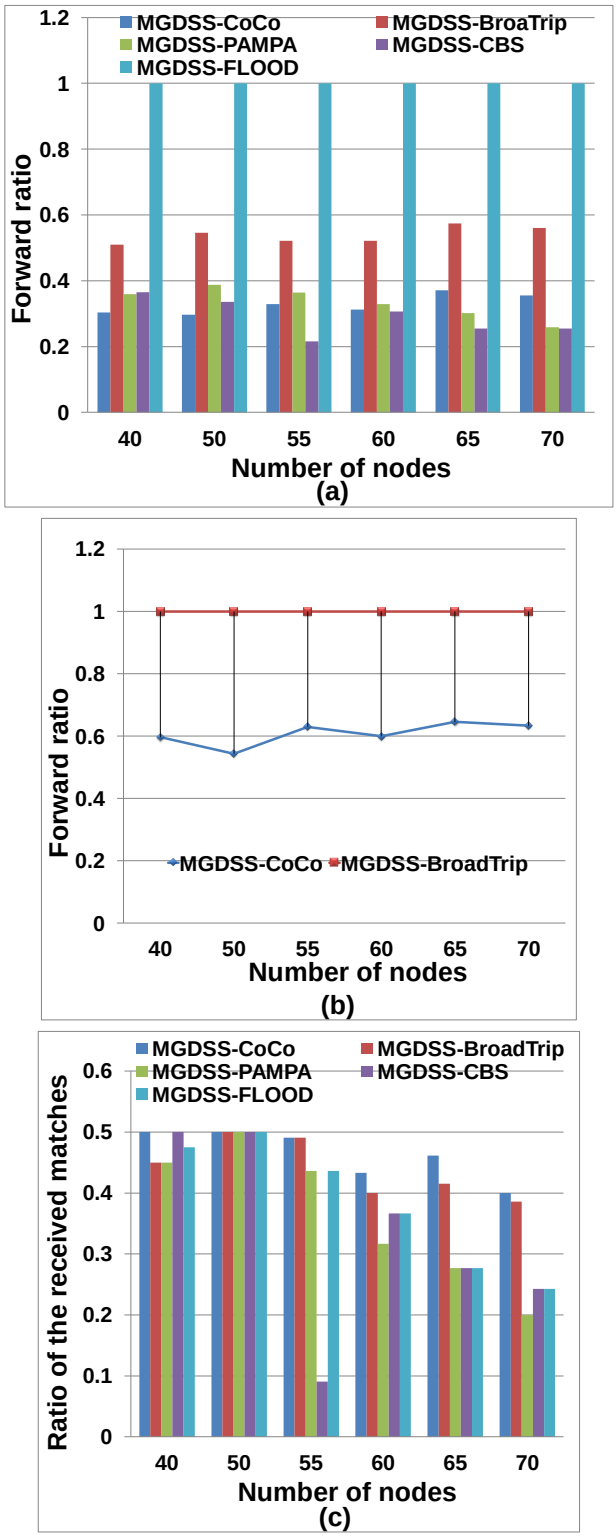

Figure 5.8 Simulation results of MGDSS-CoCo compared with four strategies-(a), (b) Forward ratio and (c) ratio of the received matches.

5.3 Conclusions

In this chapter, a novel Dynamic Messaging System (DYMES) was presented to facilitate and promote real-time social interactions among commuters during their highway trips. DYMES provides a set of original, dynamic, efficient and reliable strategies to face the existing challenges in vehicular social networks. DYMES faces the commuters' anonymity issue and their urgent need to share their driving experiences through a self-updated, efficient and reliable publish/subscribe strategy (DPSCS-2S-SRS), that allows commuters to discover the interest of each other and to be organized in stable interest-based communities in less than one second, depending on settings. Simulation results have shown that DPSCS-2S-SRS outperforms its next comparable approach by 27% to 28%, for any network size and any density, and ensures 100% reliability for all analyzed node densities.

The next challenges that DYMES copes with are the extensive bandwidth consumption in infrastructure-based vehicular networks and the urgent need of commuters to control the current state of their congested commute. DYMES allows commuters clustered in communities to be informed about the real-time traffic conditions using an efficient and reliable Mobile Gateway Discovery Mechanism (MGDM) that outperforms its next best comparable approach by 71% to 100% for any network size and any density. DYMES allows nodes clustered in congested commutes to communicate with the infrastructure, simultaneously, using a Mobile Gateway Selection Mechanism (MGSM). A handover mechanism is also proposed in DYMES and allows commuters to rediscover their neighborhood once they lose their optimal connection with the selected gateway(s). Extensive simulation results have shown that our Mobile Gateway Discovery/Selection Strategy based on the Context-aware Coding service, MGDSS-CoCo, outperforms its next comparable approach by 35% to 45% for any network size and any density. The performance evaluation of the overall architecture and the encouraging results we obtained, prove that the DYMES's dynamic, efficient and reliable strategies can be used as a building block for the development of any kind of vehicular social applications. The simulation results also proved that DYMES is the suitable messaging system to reduce the burden of congestion during the commuters' highway trips, and also to improve the quality of their real-time social interactions in dense environments.

CHAPTER 6 CONCLUSION

6.1 Summary of the Thesis

Vehicular social networks have recently drawn the attention among researchers. They are paving the way for an appealing kind of vehicular applications and services that are user-oriented and that aim at facilitating social interactions among commuters on the road during their daily dreaded commutes.

Many researchers have been focusing upon devising either distributed or centralized vehicular social systems that tackle the specific needs of commuters on the road. However, and to the best of our knowledge, none of them have proposed a hybrid vehicular social system that supports the development of centralized and distributed vehicular social applications and services while paying the required attention to the underlying hybrid vehicular communication layers.

In this thesis, we tackled the problem of the lack of hybrid vehicular social communication abstractions in the literature. We introduced a novel, efficient, reliable, hybrid and Dynamic Messaging System (DYMES) that enables real-time social interactions among commuters during their lengthy commutes using a set of efficient, dynamic, distributed and centralized communication abstractions.

The first contribution of this thesis was to build DYMES. In order to do so, we first identified a meta-strategy to guide the design of the distributed and centralized communication abstractions on which it relies. We found that the publish/subscribe model is an attractive solution for mobile computing and that it matches with the commuters' communication requirements, in a way that it allows them to publish and subscribe to messages without even knowing each other.

We then designed a set of hybrid dynamic publish/subscribe abstractions that allow commuters to efficiently and reliably communicate with each other based on their contextual information and without revealing their identities, while taking into account the dynamic nature of the shared information.

Our proposed dynamic publish/subscribe abstractions were then subdivided into four original contributions, all falling under the umbrella of the DYMES architecture. They were evaluated and compared to the previously proposed mechanisms and are summarized as follows.

The second contribution of this thesis was a Dynamic Publish/Subscribe Clustering Strategy

(DPSCS) that breaks the commuters' social isolation on the road and allows them to be organized in stable interest-based communities, without flooding the network. DPSCS introduced a novel dynamic publish/subscribe strategy that provides the opportunity for each commuter to build a stable and self-updated community based on its own interest through the propagation of a single and dynamic publication around him, for a determined duration, leaving the online matching process to subscribers. Second, the propagation of this persistent publication throughout the network was achieved using a novel two Shot-Stable Routing Service (2S-SRS) that improves the efficiency and the reliability of the commuter's single publication disseminated over the network through forwarding it using nodes that have a stable connectivity with the sender, and that are farther from the sender's location, and closer to two boundary points. DPSCS-2S-SRS stops the building of each commuter's community one the TTL of the commuter's publication expires. Our results showed that DPSCS-2S-SRS succeeds to build self-updated, efficient and reliable communities in less than once second and that it outperforms its next comparable approach by 27% to 28%, for any network size and any density, and also ensures 100% reliability for all analyzed node densities.

The third contribution of this thesis was a Mobile Gateway Discovery and Selection Strategy (MGDSS) that tackled the problem of the huge bandwidth consumption in infrastructure-based vehicular networks and the urgent need of commuters to control, simultaneously, the current state of their congested commute.

MGDSS was based on the outcome of the second contribution (DPSCS) and allows commuters clustered in communities to cooperatively and efficiently control the current state of the traffic conditions and to connect to the Internet.

To do so, MGDSS uses an efficient and reliable Mobile Gateway Discovery Mechanism (MGDM) that is based on a novel Context-aware Coding Service (CoCo). The latter allows each commuter to efficiently and reliably explore its neighborhood by broadcasting its contextual information using a single message, and without flooding the network with control messages. MGDM-CoCo provides high efficiency and reliability in highly dense environments using two mechanisms. It is based on 1) network coding that aims at reducing the overhead of multiple messages and on 2) 2S-SRS that aims at reducing the overhead of a single message and at increasing its reliability throughout the network. Our results showed that MGDM outperforms its next best comparable approach by 71% to 100% for any network size and any density.

MGDSS allows nodes clustered in congested commutes to communicate with the infrastructure, simultaneously, upon the end of the discovery process. To do so, it uses a Mobile Gateway Selection Mechanism (MGSM) that allows nodes to select a minimum number of

central stable gateways based on the nodes' information they received during the discovery strategy (MGDM). They then establish a connection with the base station through the selected node(s) to which they send their publications and subscriptions simultaneously, using stable links. The gateway forwards the received messages/queries to the base station, where an online matching strategy is executed and that aims at updating the current locations and speeds of subscribers and at sending positive matches to the corresponding subscribers by reversing the paths received in their subscriptions. A handover mechanism was also proposed in MGDSS strategy and allows commuters to rediscover their neighborhood once they lose their optimal connection with the selected gateway(s). Our extensive simulation results showed that our MGDSS strategy outperforms its next comparable approach by 35% to 45% for any network size and any density.

The fourth contribution of this thesis was BroadTrip, a novel efficient and reliable broadcasting algorithm for vehicles transiting in platoons. It tackled the problem of disseminating messages in distributed vehicular environments and was specifically intended for automated driving technology where cars follow each other in platoons and exchange messages in order to coordinate their movements, inform each other of emergencies and share relevant contextual social information. Two mechanism were combined in BroadTrip. The first one is a proximity-based mechanism to increase the efficiency of a single message broadcast, the second one is network coding that further increases the efficiency by performing wise coding decisions without requiring knowledge about the nodes' neighborhood. Our results proved that BroadTrip outperforms the next best comparable approach by 12% to 38% depending on settings.

The fifth contribution of this thesis was SocialDrive, an online publish/subscribe strategy for vehicles transiting in platoons. It tackled the problem of disseminating dynamic contextual updates in highly dense and distributed platoons, through extending BroadTrip and using it as an underlying protocol in the DYMES's architecture. SocialDrive allows commuters to publish their dynamic persistent updates throughout the network and to subscribe using an online context-aware filter. Upon the delivery of persistent updates, each subscriber creates a subscription and uses the time at which he created its subscription in the matching filter in order to compare it with the time at which the received updates were issued. A match is positive, if and only if, the time at which the persistent update was issued is very close to the time at which the subscription is created. SocialDrive-BroadTrip improves the quality of social communication among commuters clustered in dense platoons and increases their driving comfort by allowing them to receive the latest issued dynamic contextual updates that match their interests. Our results showed that SocialDrive outperformed the next best comparable approaches by 26% to 58% depending on settings.

Overall, the encouraging results we obtained proved that the DYMES' dynamic, efficient, reliable, stable and hybrid communication abstractions can be used as a building block for the development of any kind of vehicular social applications. Our results also proved that DYMES is the best hybrid messaging system that efficiently and reliably meets the diverse commuters' requirements in hybrid and dense vehicular environments.

6.2 Limitations

Despite the above promising results, our findings are also exposed to some limitations such as:

- The work in chapter 4 has a limitation that can lead to an interesting future project. It must be recalled that the proposed broadcasting protocol, BroadTrip allows to efficiently and reliably disseminate messages among vehicles transiting in platoons. In fact, in certain sparse settings where the distance between vehicles is above 50 meters, the reliability of BroadTrip could drastically decrease due to the heterogeneous transmission capabilities of some nodes that might decide to retransmit a message and cancel the schedule of their neighbors without reaching any new nodes.

- Chapter 5 proposed a Dynamic Publish/Subscribe Clustering Strategy based on the two Shot Stable Routing Service, (DPSCS-2S-SRS), that allows each commuter to create a stable and self-updated interest-based community through the propagation of a single dynamic and persistent publication which flies throughout the network using stable and farther nodes. Matching is performed by subscribers based on the publisher's current location, direction and velocity. The performance evaluation of DPSCS strategy showed that the match rate, which represents the number of commuters that match the source's publication, decreases as long as the density of nodes increases. This is due to the vehicles' movements distribution generated by SUMO. We believe that this result would change if we use a real movement trace.

- Security remains a big concern of our work. The advantages of building proximity and interest-based social communities in order to enable real-time social interactions among commuters and to promote user-oriented services in VSNs, may lead to undesirable consequences such as the spread of misinformation through information brokers to several communities. Moreover, although releasing information about location enables many useful location-based services, it may raise many privacy issues about the malicious use of location information. In addition, and despite the fact that anonymization (Hara

et al., 2014) and blurring (Shin et al., 2015) mechanisms ensure and enhance the user's privacy by restricting the access to the identity and to the location, there is still a need of strategic investigation on the privacy and the security issues in VSNs.

6.3 Future Work

Our future research directions mostly build on the above mentioned limitations. We outline some of them as follows.

- We plan to implement DYMES in real testbed that will integrate different wireless technologies such as IEEE 802.11, 4G, LTE, among others, in order to assess DYMES's performance in real scenarios. Such testbed experiments will pave the way for further improvements and will provide commuters with seamless wireless connectivity. It will also promote the proliferation of a plethora of real-time vehicular social applications such as online gaming and video conferencing that will certainly shape the future of the internet of cars.
- We intend to improve our proposed handover mechanism in order to allow nodes to establish a connection to another BTS through a minimum number of new stable mobile gateways.
- We also plan to investigate ideas to support social trust mechanisms in order to reduce the presence of malicious commuters over the network. The latter may annoy the message dissemination process during social communications. It may also lead to undesirable consequences, such as the spread of misinformation to several social communities.
- Enabling social interactions among commuters during their highway travels makes them generate a huge amount of real-time geosocial data, which transforms VSNs to a Big Data problem. We plan to design adequate strategies and architectures using Big data distributed technologies, in order to learn about the dynamic sociological patterns of commuters on the road and to formulate decision policies able to differentiate commuter-centric services according to their dynamic priorities.

We believe that investigating these issues represents a significant research avenue and we hope that the findings in this thesis may help stimulate further work in this area.

REFERENCES

Location-based Publish/Subscribe, 2005. Online: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1565971

N. Abbani, M. Jomaa, T. Tarhini, H. Artail, et W. Hajj, "Managing social networks in vehicular networks using trust rules", In *IEEE Symposium on Wireless Technology and Applications (ISWTA)*, 2011.

R. Ahlswede, N. Cai, S. Y. Li, et R. W. Yeung, "Network information flow", *IEEE Trans. Inf. Theor.*, vol. 46, no. 4, pp. 1204–1216, Sep. 2006. DOI: 10.1109/18.850663. Online: http://dx.doi.org/10.1109/18.850663

S. Ahmed et S. S. Kanhere, "Vanetcode: Network coding to enhance cooperative downloading in vehicular ad-hoc networks", In *Proceedings of the 2006 International Conference on Wireless Communications and Mobile Computing*, série IWCMC '06. New York, NY, USA: ACM, 2006, pp. 527–532. DOI: 10.1145/1143549.1143654. Online: http://doi.acm.org/10.1145/1143549.1143654

A. Asterjadhi, E. Fasolo, M. Rossi, J. Widmer, et M. Zorzi, "Toward network coding-based protocols for data broadcasting in wireless ad hoc networks", *IEEE Transactions on Wireless Communications*, vol. 9, no. 2, pp. 662–673, 2010. DOI: 10.1109/TWC.2010.02.081057. Online: http://dx.doi.org/10.1109/TWC.2010.02.081057

S. Barghi, A. Benslimane, et C. Assi, "A lifetime-based routing protocol for connecting vanets to the internet", In *10th IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks, WOWMOM 2009, Kos Island, Greece, 15-19 June, 2009*, 2009, pp. 1–9. DOI: 10.1109/WOWMOM.2009.5282470. Online: http://dx.doi.org/10.1109/WOWMOM.2009.5282470

P. Basu, N. Khan, et T. D. Little, "A mobility based metric for clustering in mobile ad hoc networks", In *In International Workshop on Wireless Networks and Mobile Computing (WNMC2001*, 2001, pp. 413–418.

M. Behrisch, L. Bieker, J. Erdmann, et D. Krajzewicz, "Sumo - simulation of urban mobility: An overview", In *in SIMUL 2011, The Third International Conference on Advances in System Simulation*, 2011, pp. 63–68.

A. Benslimane, S. Barghi, et C. Assi, “An efficient routing protocol for connecting vehicular networks to the internet”, *Pervasive and Mobile Computing*, vol. 7, no. 1, pp. 98–113, 2011. DOI: 10.1016/j.pmcj.2010.09.002. Online: http://dx.doi.org/10.1016/j.pmcj.2010.09.002

——, “An efficient routing protocol for connecting vehicular networks to the internet.” *Pervasive and Mobile Computing*, vol. 7, no. 1, pp. 98–113, 2011. Online: http://dblp.uni-trier.de/db/journals/percom/percom7.html#BenslimaneBA11

A. Benslimane, T. Taleb, et R. Sivaraj, “Dynamic clustering-based adaptive mobile gateway management in integrated VANET - 3g heterogeneous wireless networks”, *IEEE Journal on Selected Areas in Communications*, vol. 29, no. 3, pp. 559–570, 2011. DOI: 10.1109/JSAC.2011.110306. Online: http://dx.doi.org/10.1109/JSAC.2011.110306

P. Costa, C. Mascolo, M. Musolesi, et G. P. Picco, “Socially-aware routing for publish-subscribe in delay-tolerant mobile ad hoc networks”, *IEEE J.Sel. A. Commun.*, vol. 26, no. 5, pp. 748–760, Juin 2008. DOI: 10.1109/JSAC.2008.080602. Online: http://dx.doi.org/10.1109/JSAC.2008.080602

G. Cugola et M. Migliavacca, “On context-aware publish-subscribe”, 2008.

G. Cugola, A. Margara, et M. Migliavacca, “Context-aware publish-subscribe: Model, implementation, and evaluation.” In *ISCC*. IEEE, 2009, pp. 875–881. Online: http://dblp.uni-trier.de/db/conf/iscc/iscc2009.html#CugolaMM09

C. Ellis, H. Miranda, et F. Taïani, “Count on me: Lightweight ad-hoc broadcasting in heterogeneous topologies”, In *Proceedings of the International Workshop on Middleware for Pervasive Mobile and Embedded Computing*, série M-PAC ’09. New York, NY, USA: ACM, 2009, pp. 1:1–1:6. DOI: 10.1145/1657127.1657129. Online: http://doi.acm.org/10.1145/1657127.1657129

P. T. Eugster, P. A. Felber, R. Guerraoui, et A.-M. Kermarrec, “The many faces of publish/subscribe”, *ACM Comput. Surv.*, vol. 35, no. 2, pp. 114–131, Juin 2003. DOI: 10.1145/857076.857078. Online: http://doi.acm.org/10.1145/857076.857078

P. T. Eugster, B. Garbinato, et A. Holzer, “Location-based publish/subscribe”, In *Fourth IEEE International Symposium on Network Computing and Applications (NCA 2005),*

27-29 July 2005, Cambridge, MA, USA, 2005, pp. 279–282. DOI: 10.1109/NCA.2005.29. Online: http://dx.doi.org/10.1109/NCA.2005.29

——, "Effective location-based publish/subscribe in manets", In *Technical Report DOP20081215*, 2008.

——, "Pervaho: A specialized middleware for mobile context-aware applications", *Electronic Commerce Research*, vol. 9, no. 4, pp. 245–268, 2009. DOI: 10.1007/s10660-009-9042-4. Online: http://dx.doi.org/10.1007/s10660-009-9042-4

R. Fei, K. Yang, et X. Cheng, "A cooperative social and vehicular network and its dynamic bandwidth allocation algorithms", In *IEEE INFOCOM Conf.*, 2011.

U. T. T. Force, "The high cost of congestion in canadian cities", *Council of Ministers Responsible for Transportation and Highway Safety*, 2012.

C. Fragouli, J. Widmer, et J. L. Boudec, "Efficient broadcasting using network coding", *IEEE/ACM Trans. Netw.*, vol. 16, no. 2, pp. 450–463, 2008. DOI: 10.1145/1373990.1374006. Online: http://doi.acm.org/10.1145/1373990.1374006

N. Frangiadakis, D. Câmara, F. Filali, A. A. F. Loureiro, et N. Roussopoulos, "Virtual access points for vehicular networks", In *Proceedings of the 1st International Conference on MOBILe Wireless MiddleWARE, Operating Systems, and Applications*, série MOBILWARE '08. ICST, Brussels, Belgium, Belgium: ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2007, pp. 14:1–14:5. Online: http://dl.acm.org/citation.cfm?id=1361492.1361509

L. C. Freeman, "Centrality in social networks conceptual clarification", *Social Networks*, p. 215, 1978.

D. Frey et G.-C. Roman, "Context-aware publish subscribe in mobile ad hoc networks", In *Proceedings of the 9th International Conference on Coordination Models and Languages*, série COORDINATION'07. Berlin, Heidelberg: Springer-Verlag, 2007, pp. 37–55. Online: http://dl.acm.org/citation.cfm?id=1764606.1764610

B. Garbinato, A. Holzer, et F. Vessaz, "Six-shot broadcast: A context-aware algorithm for efficient message diffusion in manets", In *On the Move to Meaningful Internet Systems: OTM 2008, OTM 2008 Confederated International Conferences, CoopIS, DOA, GADA, IS,*

and ODBASE 2008, Monterrey, Mexico, November 9-14, 2008, Proceedings, Part I, 2008, pp. 625–638.

——, "Context-aware broadcasting approaches in mobile ad hoc networks", *Comput. Netw.*, vol. 54, no. 7, pp. 1210–1228, Mai 2010. DOI: 10.1016/j.comnet.2009.11.008. Online: http://dx.doi.org/10.1016/j.comnet.2009.11.008

——, "Six-shot multicast: A location-aware strategy for efficient message routing in manets", In *Proceedings of The Ninth IEEE International Symposium on Networking Computing and Applications, NCA 2010, July 15-17, 2010, Cambridge, Massachusetts, USA*, 2010, pp. 1–9. DOI: 10.1109/NCA.2010.8. Online: http://dx.doi.org/10.1109/NCA.2010.8

Z. J. Haas, J. Y. Halpern, et L. Li, "Gossip-based ad hoc routing", *IEEE/ACM Trans. Netw.*, vol. 14, no. 3, pp. 479–491, Juin 2006. DOI: 10.1109/TNET.2006.876186. Online: http://dx.doi.org/10.1109/TNET.2006.876186

L. Han, S. Smaldone, P. Shankar, J. Boyce, et L. Iftode, "Ad-hoc voice-based group communication", In *Eigth Annual IEEE International Conference on Pervasive Computing and Communications, PerCom 2010, March 29 - April 2, 2010, Mannheim, Germany*, 2010, pp. 190–198. DOI: 10.1109/PERCOM.2010.5466977. Online: http://dx.doi.org/10.1109/PERCOM.2010.5466977

T. Hara, Y. Arase, A. Yamamoto, X. Xie, M. Iwata, et S. Nishio, "Location anonymization using real car trace data for location based services", In *Proceedings of the 8th International Conference on Ubiquitous Information Management and Communication*, série ICUIMC '14. New York, NY, USA: ACM, 2014, pp. 34:1–34:8. DOI: 10.1145/2557977.2558044. Online: http://doi.acm.org/10.1145/2557977.2558044

A. Holzer, S. Maaroufi, et S. Pierre, "DYMES: A dynamic messaging service for vanets", In *IEEE 6th International Conference on Wireless and Mobile Computing, Networking and Communications, WiMob 2010, Niagara Falls, Ontario, Canada, 11-13 October, 2010*, 2010, pp. 513–520. DOI: 10.1109/WIMOB.2010.5645017. Online: http://dx.doi.org/10.1109/WIMOB.2010.5645017

——, "BROADTRIP: broadcast for transit in platoons", In *IEEE 7th International Conference on Wireless and Mobile Computing, Networking and Communications, WiMob 2011, Shanghai, China, October 10-12, 2011*, 2011, pp. 301–306.

DOI: 10.1109/WiMOB.2011.6085384. Online: http://dx.doi.org/10.1109/WiMOB.2011.6085384

A. Holzer, P. Eugster, et B. Garbinato, “Evaluating implementation strategies for location-based multicast addressing”, *IEEE Trans. Mob. Comput.*, vol. 12, no. 5, pp. 855–867, 2013. DOI: 10.1109/TMC.2012.54. Online: http://doi.ieeecomputersociety.org/10.1109/TMC.2012.54

X. Hu, J. Zhao, D. Zhou, et V. C. Leung, “A semantics-based multi-agent framework for vehicular social network development”, In *Proceedings of the First ACM International Symposium on Design and Analysis of Intelligent Vehicular Networks and Applications*, série DIVANet ’11. New York, NY, USA: ACM, 2011, pp. 87–96. DOI: 10.1145/2069000.2069015. Online: http://doi.acm.org/10.1145/2069000.2069015

X. Hu, W. wang, et V. C. Leung, “Vssa: A service-oriented vehicular social-networking platform for transportation efficiency”, In *Proceedings of the Second ACM International Symposium on Design and Analysis of Intelligent Vehicular Networks and Applications*, série DIVANet ’12. New York, NY, USA: ACM, 2012, pp. 31–38. DOI: 10.1145/2386958.2386964. Online: http://doi.acm.org/10.1145/2386958.2386964

B. Hughes, R. Meier, R. Cunningham, et V. Cahill, “Towards real-time middleware for vehicular ad hoc networks”, In *Proceedings of the 1st ACM International Workshop on Vehicular Ad Hoc Networks*, série VANET ’04. New York, NY, USA: ACM, 2004, pp. 95–96. DOI: 10.1145/1023875.1023894. Online: http://doi.acm.org/10.1145/1023875.1023894

U. Javaid, T. M. Rasheed, D. Meddour, et T. Ahmed, “Adaptive distributed gateway discovery in hybrid wireless networks”, In *WCNC 2008, IEEE Wireless Communications & Networking Conference, March 31 2008 - April 3 2008, Las Vegas, Nevada, USA, Conference Proceedings*, 2008, pp. 2735–2740. DOI: 10.1109/WCNC.2008.479. Online: http://dx.doi.org/10.1109/WCNC.2008.479

K. R. Jayaram, P. Eugster, et C. Jayalath, “Parametric content-based publish/subscribe”, *ACM Trans. Comput. Syst.*, vol. 31, no. 2, pp. 4:1–4:52, Mai 2013. DOI: 10.1145/2465346.2465347. Online: http://doi.acm.org/10.1145/2465346.2465347

S. Kaveh, “Application-based packet routing in vehicular networks”, Thèse de doctorat, 2012. DOI: 10.14288/1.0072906. Online: http://hdl.handle.net/2429/42787

I. Leontiadis, “Publish/subscribe notification middleware for vehicular networks”, In *Proceedings of the 4th on Middleware Doctoral Symposium*, série MDS ’07. New York, NY, USA: ACM, 2007, pp. 12:1–12:6. DOI: 10.1145/1377934.1377936. Online: http://doi.acm.org/10.1145/1377934.1377936

I. Leontiadis et C. Mascolo, “Geopps: Geographical opportunistic routing for vehicular networks”, *Proceeding of IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks 2014*, vol. 0, pp. 1–6, 2007. DOI: http://doi.ieeecomputersociety.org/10.1109/WOWMOM.2007.4351688

——, “Opportunistic spatio-temporal dissemination system for vehicular networks”, In *Proceedings of the 1st International MobiSys Workshop on Mobile Opportunistic Networking*, série MobiOpp ’07. New York, NY, USA: ACM, 2007, pp. 39–46. DOI: 10.1145/1247694.1247702. Online: http://doi.acm.org/10.1145/1247694.1247702

I. Leontiadis, P. Costa, et C. Mascolo, “A hybrid approach for content-based publish/subscribe in vehicular networks”, *Pervasive and Mobile Computing*, vol. 5, no. 6, pp. 697 – 713, 2009, perCom 2009. DOI: http://dx.doi.org/10.1016/j.pmcj.2009.07.016. Online: http://www.sciencedirect.com/science/article/pii/S1574119209000662

I. Lequerica, M. Longaron, et P. Ruiz, “Drive and share: efficient provisioning of social networks in vehicular scenarios”, *IEEE Communications Magazine*, vol. 48, no. 11, pp. 90–97, Nov. 2010. DOI: 10.1109/mcom.2010.5621973. Online: http://dx.doi.org/10.1109/mcom.2010.5621973

T. Litman, “Smart congestion relief comprehensive evaluation of traffic congestion costs and congestion reduction strategies”, *Victoria Transport Policy Instit.*, 2014.

T. H. Luan, R. Lu, X. Shen, et F. Bai, “Social on the road: enabling secure and efficient social networking on highways”, *IEEE Wireless Commun.*, vol. 22, no. 1, pp. 44–51, 2015. DOI: 10.1109/MWC.2015.7054718. Online: http://dx.doi.org/10.1109/MWC.2015.7054718

T. H. Luan, X. Shen, F. Bai, et L. Sun, “Feel bored? join verse! engineering vehicular proximity social networks”, *IEEE T. Vehicular Technology*, vol. 64, no. 3, pp. 1120–1131, 2015. DOI: 10.1109/TVT.2014.2329481. Online: http://dx.doi.org/10.1109/TVT.2014.2329481

S. Maaroufi et S. Pierre, “Vehicular social systems: an overview and a performance case study”, In *Proceedings of the fourth ACM international symposium on Development and analysis of intelligent vehicular networks and applications, ACM DIVANet@MSWiM 2014, Montreal, QC, Canada, September 21-26, 2014*, 2014, pp. 17–24. DOI: 10.1145/2656346.2656352. Online: http://doi.acm.org/10.1145/2656346.2656352

——, “Onlinecruise: An online social grouping strategy for vehicular social networks”, In *Proceedings of the 5th ACM Symposium on Development and Analysis of Intelligent Vehicular Networks and Applications, ACM DIVANet@MSWiM 2015*. New York, NY, USA: ACM, 2015, pp. 67–70. DOI: 10.1145/2815347.2815360. Online: http://doi.acm.org/10.1145/2815347.2815360

——, “Discovering the architecture of a dynamic messaging system for vehicular social networks”, *IEEE Transactions On Mobile Computing (Second Round)*, 2015.

Mediative, “Location-based marketing made easy: a marketer’s guide”, *Mediative.com*, 2013.

R. Meier et V. Cahill, “Steam: event-based middleware for wireless ad hoc networks”, In *Distributed Computing Systems Workshops, 2002. Proceedings. 22nd International Conference on*, 2002, pp. 639– 644. DOI: 10.1109/ICDCSW.2002.1030841. Online: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1030841

——, “STEAM: event-based middleware for wireless ad hoc network”, In *22nd International Conference on Distributed Computing Systems, Workshops (ICDCSW ’02) July 2-5, 2002, Vienna, Austria, Proceedings*, 2002, pp. 639–644. DOI: 10.1109/ICDCSW.2002.1030841. Online: http://dx.doi.org/10.1109/ICDCSW.2002.1030841

——, “On event-based middleware for location-aware mobile applications”, *IEEE Trans. Softw. Eng.*, vol. 36, no. 3, pp. 409–430, Mai 2010. DOI: 10.1109/TSE.2009.90. Online: http://dx.doi.org/10.1109/TSE.2009.90

H. Miranda, S. Leggio, L. Rodrigues, et K. Raatikainen, “A power-aware broadcasting algorithm”, In *Proceedings of the 17th annual IEEE international symposium on personal, indoor and mobile radio communications (PIMRC 06)*, 2006.

S.-Y. Ni, Y.-C. Tseng, Y.-S. Chen, et J.-P. Sheu, “The broadcast storm problem in a mobile ad hoc network”, In *Proceedings of the 5th Annual ACM/IEEE*

International Conference on Mobile Computing and Networking, série MobiCom '99. New York, NY, USA: ACM, 1999, pp. 151–162. DOI: 10.1145/313451.313525. Online: http://doi.acm.org/10.1145/313451.313525

F. J. Ovalle;Martínez, A. Nayak, I. Stojmenovic, J. Carle, et D. Simplot-Ryl, "Area-based beaconless reliable broadcasting in sensor networks", *Int. J. Sen. Netw.*, vol. 1, no. 1/2, pp. 20–33, Sep. 2006. DOI: 10.1504/IJSNET.2006.010831. Online: http://dx.doi.org/10.1504/IJSNET.2006.010831

C. E. Palazzi, M. Roccetti, S. Ferretti, G. Pau, et M. Gerla, "Online games on wheels: Fast game event delivery in vehicular ad-hoc networks", In *IEEE Intelligent Vehicles Symposium 2007*. Istanbul: IEEE Computer Society, Juin 2007.

C. E. Palazzi, M. Roccetti, et S. Ferretti, "An intervehicular communication architecture for safety and entertainment", *Trans. Intell. Transport. Sys.*, vol. 11, no. 1, pp. 90–99, Mars 2010. DOI: 10.1109/TITS.2009.2029078. Online: http://dx.doi.org/10.1109/TITS.2009.2029078

V. Paruchuri, A. Durresi, et R. Jain, "Optimized flooding protocol for ad hoc networks", *CoRR*, vol. cs.NI/0311013, 2003. Online: http://arxiv.org/abs/cs.NI/0311013

M. R., R. S., S. S., et S. K., "A novel multi-hop b3g architecture for adaptive gateway management in heterogeneous wireless networks", *2013 IEEE 9th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, vol. 0, pp. 447–452, 2009. DOI: http://doi.ieeecomputersociety.org/10.1109/WiMob.2009.82

R. Ramdhany, P. Grace, G. Coulson, et D. Hutchison, "MANETKit: supporting the dynamic deployment and reconfiguration of ad-hoc routing protocols", In *Middleware '09: Proceedings of the 10th ACM/IFIP/USENIX International Conference on Middleware*. New York, NY, USA: Springer-Verlag New York, Inc., 2009. Online: http://portal.acm.org/toc.cfm?id=1656980&idx=SERIES877&type=proceeding&coll=ACM&dl=ACM&part=series&WantType=Proceedings&title=Middleware&CFID=64297195&CFTOKEN=23882669

A. Ranganathan, J. Al-Muhtadi, S. Chetan, R. Campbell, et M. D. Mickunas, "Middlewhere: A middleware for location awareness in ubiquitous computing applications", In *Proceedings of the 5th ACM/IFIP/USENIX International Conference on Middleware*, série Middleware '04. New York, NY, USA: Springer-Verlag New York, Inc., 2004, pp. 397–416. Online: http://dl.acm.org/citation.cfm?id=1045658.1045686

G. F. Riley et T. R. Henderson, “The ns-3 Network Simulator Modeling and Tools for Network Simulation”, In *Modeling and Tools for Network Simulation*, K. Wehrle, M. Güneş, et J. Gross, éds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, ch. 2, pp. 15–34. DOI: 10.1007/978-3-642-12331-3_2. Online: http://dx.doi.org/10.1007/978-3-642-12331-3_2

W. Sha, D. Kwak, B. Nath, et L. Iftode, “Social vehicle navigation: integrating shared driving experience into vehicle navigation”, In *14th Workshop on Mobile Computing Systems and Applications, HotMobile '13, Jekyll Island, GA, USA, February 26-27, 2013*, 2013, p. 16. DOI: 10.1145/2444776.2444798. Online: http://doi.acm.org/10.1145/2444776.2444798

M. Shin, C. Cornelius, A. Kapadia, N. Triandopoulos, et D. Kotz, “Location Privacy for Mobile Crowd Sensing through Population Mapping”, *Sensors*, Juin 2015. DOI: 10.3390/s150715285. Online: http://dx.doi.org/10.3390/s150715285

S. Smaldone, L. Han, P. Shankar, et L. Iftode, “Roadspeak: Enabling voice chat on roadways using vehicular social networks”, In *Proceedings of the 1st Workshop on Social Network Systems*, série SocialNets '08. New York, NY, USA: ACM, 2008, pp. 43–48. DOI: 10.1145/1435497.1435505. Online: http://doi.acm.org/10.1145/1435497.1435505

W. Su, S.-J. Lee, et M. Gerla, “Mobility prediction and routing in ad hoc wireless networks”, *Int. J. Netw. Manag.*, vol. 11, no. 1, pp. 3–30, Jan. 2001. DOI: 10.1002/nem.386. Online: http://dx.doi.org/10.1002/nem.386

T. Taleb, E. Sakhaee, A. Jamalipour, K. Hashimoto, N. Kato, et Y. Nemoto, “A stable routing protocol to support its services in vanet networks”, *IEEE Transactions on Vehicular Technology*, vol. 56, no. 6, pp. 3337–3347, 2007.

S. Ucar, S. C. Ergen, et z. Özkasap, “Vmasc: Vehicular multi-hop algorithm for stable clustering in vehicular ad hoc networks.” In *WCNC*. IEEE, 2013, pp. 2381–2386. Online: http://dblp.uni-trier.de/db/conf/wcnc/wcnc2013.html#UcarEO13

A. M. Vegni et V. Loscri, “A Survey on Vehicular Social Networks”, *Communications Surveys and Tutorials, IEEE Communications Society*, p. 28, Juil. 2015. Online: https://hal.archives-ouvertes.fr/hal-01174026

B. Vo et S. Bellovin, “Anonymous publish-subscribe systems”.

N. Wisitpongphan, O. K. Tonguz, J. S. Parikh, P. Mudalige, F. Bai, et V. Sadekar, “Broadcast storm mitigation techniques in vehicular ad hoc networks”, *Wireless Commun.*, vol. 14, no. 6, pp. 84–94, Déc. 2007. DOI: 10.1109/MWC.2007.4407231. Online: http://dx.doi.org/10.1109/MWC.2007.4407231

Z. Zhang, A. Boukerche, et R. Pazzi, “A novel multi-hop clustering scheme for vehicular ad-hoc networks”, In *Proceedings of the 9th ACM International Symposium on Mobility Management and Wireless Access*, série MobiWac ’11. New York, NY, USA: ACM, 2011, pp. 19–26. DOI: 10.1145/2069131.2069135. Online: http://doi.acm.org/10.1145/2069131.2069135

Y. Zhao et J. Wu, “Socially-aware publish/subscribe system for human networks”, In *2010 IEEE Wireless Communications and Networking Conference, WCNC 2010, Proceedings, Sydney, Australia, 18-21 April 2010*, 2010, pp. 1–6. DOI: 10.1109/WCNC.2010.5506366. Online: http://dx.doi.org/10.1109/WCNC.2010.5506366