

GERENCIAMIENTO HIERÁRQUICO DE REDES VEICULARES

EWERTON MONTEIRO SALVADOR

GERENCIAMENTO HIERÁRQUICO DE REDES VEICULARES

Tese apresentada ao Programa de Pós-Graduação em Ciência da Computação do Instituto de Ciências Exatas da Universidade Federal de Minas Gerais como requisito parcial para a obtenção do grau de Doutor em Ciência da Computação.

ORIENTADOR: JOSÉ MARCOS SILVA NOGUEIRA
COORIENTADOR: DANIEL FERNANDES MACEDO

Belo Horizonte
Outubro de 2016

EWERTON MONTEIRO SALVADOR

HIERARCHICAL MANAGEMENT OF VEHICULAR NETWORKS

Thesis presented to the Graduate Program
in Computer Science of the Universidade
Federal de Minas Gerais - Departamento
de Ciência da Computação in partial ful-
fillment of the requirements for the degree
of Doctor in Computer Science.

ADVISOR: JOSÉ MARCOS SILVA NOGUEIRA
CO-ADVISOR: DANIEL FERNANDES MACEDO

Belo Horizonte

October 2016

© 2016, Ewerton Monteiro Salvador.
Todos os direitos reservados.

Salvador, Ewerton Monteiro

S182h Hierarchical Management of Vehicular Networks /
Ewerton Monteiro Salvador. — Belo Horizonte, 2016
xxv, 109 f. : il. ; 29cm

Tese (doutorado) — Universidade Federal de Minas
Gerais - Departamento de Ciência da Computação

Orientador: José Marcos Silva Nogueira

Coorientador: Daniel Fernandes Macedo

1. Computação - Teses. 2. Vehicular ad hoc network.
3. Redes de computadores - Administração.
I. Orientador. II. Coorientador. III. Título.

CDU 519.6*22(043)



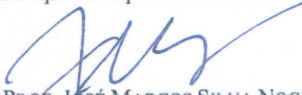
UNIVERSIDADE FEDERAL DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

FOLHA DE APROVAÇÃO

Hierarchical management of vehicular networks

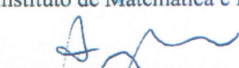
EWERTON MONTEIRO SALVADOR

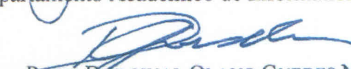
Tese defendida e aprovada pela banca examinadora constituída pelos Senhores:

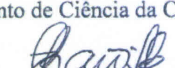

PROF. JOSÉ MARCOS SILVA NOGUEIRA - Orientador
Departamento de Ciência da Computação - UFMG


PROF. DANIEL FERNANDES MACEDO - Coorientador
Departamento de Ciência da Computação - UFMG


PROF. ALFREDO GOLDMAN VEL LEJMAN
Instituto de Matemática e Estatística - USP


PROFA. ANELISE MUNARETTO FONSECA
Departamento Acadêmico de Informática - UTFPR


PROF. DORGIVAL OLAVO GUEDES NETO
Departamento de Ciência da Computação - UFMG


PROF. LISANDRO ZAMBENEDETTI GRANVILLE
Instituto de Informática - UFRGS

Belo Horizonte, 07 de outubro de 2016.

I dedicate this work to my fiancée, Geórgia Cordeiro Dantas, and to my family.

Acknowledgments

Wow... what a long and incredible journey this doctoral course was! And along this journey I had the pleasure to meet many amazing people, who taught me and helped me in many more ways than they would ever imagine.

Firstly I would like to thank God for watching over me during both good and difficult times along my years of doctoral studies. I would be nothing without You, so thank You for making me who I am.

To my advisor, Professor José Marcos, thanks for the insane amount of lessons I will carry with me for my entire life. Thanks for teaching me how to be a better student, a better researcher, a better professor, and a better human being. Thank you for teaching me the importance of taking care of myself as well as taking care of everyone around me.

I also thank my co-advisor, Professor Daniel Macedo, for his sincere dedication in helping me to become a better researcher, for all the passionate conversations about the directions to be followed along the doctoral research, and for being a great friend that I can count on to be there for me. Whenever you need, you can count on me as well.

To Geórgia, my fiancée and “academic journey partner”, thank you for all the support, all the inspiration and all the love. You bring light and meaning to my life, and I just can’t be grateful enough for that.

I also would like to thank my family, for always being rooting for my success, and for their unconditional support during my entire life. Mom, dad, Alisson, Eweline, Maiara, and Nick, thank you for being the best family any person would ever wish for.

To Professor Stephen Farrell, from Trinity College Dublin, thank you for sharing so much valuable knowledge with me, and thank you for taking care of me during the amazing time I had in this beautiful country called Ireland.

I would like to thank Professor Lisandro Granville for being a great friend and inspiration, helping me to overcome some of the most difficult obstacles I encountered during my academic life.

To all the examiners of the final presentation of my thesis, thank you for accepting our invitation and for being willing to enlighten our research with your expertise.

To my friends Eduardo and Emanuelly, thank you for the good times together, for the unvaluable support you guys gave to Georgia and me during our doctoral studies, and for being awesome. My thanks also goes to Moshe Moo and Guru Pancita for being such great pals!

To Virgil Almeida and his family, thank you for your kindness and friendship. Thank you for all the support you gave me whenever I needed, and know that I will be there for you too whenever necessary.

To my friends from UFMG Alisson Rodrigues, Aloizio Silva, André Poersch, Cristiano Silva, Diego Da Hora, Erik de Britto, Fernando Teixeira, Jesse Leoni, Junio Salomé, Lucas Silva, Leandro Maia, Livia Almeida, Pablo Goulart, Rone Ilidio, Thiago Oliveira, Vinícius Fonseca, Vinícius Mota and Wendley Silva, thanks for the great conversations, the valuable advices, and for the good moments together.

I would like to thank my friends Rodrigo and Ayla Rebouças for your friendship. It's not long ago that we met for the first time, but ever since, you guys gave me nothing but tons of love and support, and I will always be grateful for that.

To all the many colleagues I met along my stay at UFMG, thank you very much for all the shared moments: all the laughs, all the study sessions, and even all the collective panic attacks! I also learned a lot from every one of you, and I hope we can meet again as many times as possible.

To the Programa de Pós-Graduação em Ciência da Computação (PPGCC) of UFMG and all its staff, thank you for the great support and for your patience. I'm very lucky to count with the support of such a professional and dedicated graduate programme.

And last but not least, I would like to thank CAPES, CNPq, FAPEMIG, and CEMIG for their crucial financial support to my doctoral research. Thanks for making it possible for me to pursue the dream of becoming a doctor in doing what I love to do.

Thank you. Thank you. Thank you.

*“We can only see a short distance ahead, but we can see plenty there that needs to be
done.”*

(Alan Turing)

Resumo

Redes Ad Hoc Veiculares (VANETs) são redes móveis que podem se estender por grandes áreas e cujos nós possuem mobilidade intensa. Essas características levam a frequentes atrasos de comunicação e desconexões de enlaces. Uma solução para os problemas de conectividade de VANETs é o emprego do paradigma de Rede Tolerante a Atrasos e Desconexões (Delay-Tolerant Network, ou simplesmente DTN), resultando na criação de VDTNs (Vehicular Delay-Tolerant Networks). Contudo, as desconexões frequentes e as restrições relacionadas a atraso e confiabilidade de certas aplicações para VDTN impedem o uso tanto de arquiteturas de gerenciamento convencionais quanto das arquiteturas baseadas em DTN. Por conta disso, se faz necessário o desenvolvimento de uma solução de gerenciamento que seja capaz de reagir de forma mais rápida e eficaz aos problemas que podem ocorrer em uma VDTN. Esta tese apresenta uma arquitetura de gerenciamento hierárquico (HiErarchical MANagement - HE-MAN) para VDTNs. Essa arquitetura implementa uma topologia hierárquica de gerenciamento, onde um gerente global da rede (Top-Level Manager, ou TLM) delega a gerentes intermediários (Mid-Level Managers, ou MLMs) o gerenciamento de grupos de nós interconectados. A arquitetura também é composta por nós agentes e protocolos que coordenam trocas de mensagens e a organização lógica da topologia de gerenciamento. Para construir a topologia hierárquica, foram propostos algoritmos para agrupamento de nós da rede, monitoramento local e remoto, e configuração local e remota. Resultados de simulações mostram que a arquitetura proposta organiza com sucesso a VDTN em grupos relativamente estáveis, resultando em um gerenciamento dos nós da VDTN de forma mais eficiente. Além disso, os algoritmos de monitoramento e configuração melhoram a confiabilidade e o tempo de resposta das tarefas de gerenciamento.

Palavras-chave: Gerenciamento de Redes, Redes Tolerantes a Atrasos e Desconexões, Redes Veiculares Tolerantes a Atrasos e Desconexões.

Abstract

Vehicular Ad Hoc Networks (VANETs) are mobile networks that extend over vast areas and have intense node mobility. These characteristics lead to frequent communication delays and link disruptions. A solution to the connectivity problems of VANETs is to employ the Delay Tolerant Network (DTN) paradigm, resulting in the creation of Vehicular Delay-Tolerant Networks (VDTNs). However, the frequent disruptions as well as the delay and reliability constraints of certain VDTN applications hinder the employment of both conventional and DTN-based management architectures. Because of this, the development of a management solution able to react quicker and more reliably to problems that can occur in a VDTN becomes necessary. This thesis presents the HiErarchical MANagement (HE-MAN) architecture for VDTNs. This architecture implements a hierarchical management topology, where a Top-Level Manager (TLM) delegates to Mid-Level Managers (MLMs) the management of clusters of interconnected nodes. The architecture is also composed of agents and protocols that coordinate message exchanges and the logical organization of the management topology. In order to build the hierarchical topology, algorithms were designed for network clustering, local and remote monitoring, and local and remote configuration. Simulations results show that the proposed architecture successfully organizes the VDTN in relatively stable clusters, leading to a more efficient management of VDTN nodes. Moreover, the proposed monitoring and configuration algorithms successfully improve the reliability and response time of the management tasks.

Palavras-chave: Network Management, Delay-Tolerant Network, Vehicular Delay-Tolerant Network.

List of Figures

2.1	Network management approaches	9
2.2	Example of the SNNP network management architecture [Stallings, 1998] .	11
2.3	Example of the topology of a DTN through time	13
2.4	Bundle Layer in an example of DTN communication	15
2.5	Example of a vehicular network	19
4.1	Hierarchical monitoring in a VDTN	37
4.2	Main components of the HE-MAN architecture	39
4.3	Behavior of isolated nodes according to the received type of beacon message	43
4.4	Behavior of MLMs according to the received type of beacon message . . .	45
4.5	Behavior of agent nodes according to the received type of beacon message .	46
4.6	Messages exchange in the publisher-subscriber monitoring	47
4.7	Transmission Status Propagation System supported by HE-MAN	53
5.1	Snapshot of the buses (red dots) and bus stops (black dots) in the traffic trace [Doering et al., 2010].	61
5.2	Cumulative distribution function (CDF) of contact duration within the trace files	62
6.1	Average time that nodes spend connected to clusters	72
6.2	Average lifetime of a cluster	73
6.3	Average number of clusters formed during the experiments	74
6.4	Average size of clusters formed during the experiments	74
6.5	Number of cluster-head transferances per cluster	75
6.6	Notification delay in local communication	76
6.7	Notification delivery ratio in local communication	76
6.8	Average time that nodes spend connected to clusters	78
6.9	Average lifetime of a cluster	79
6.10	Average number of clusters formed during the experiments	79

6.11	Average size of clusters formed during the experiments	80
6.12	Number of CH transferences per cluster	81
6.13	Notification delay in local communication	81
6.14	Notification delivery ratio in local communication	82
6.15	Average local notification delay with different buffer sizes	84
6.16	Remote reports average delay with different buffer sizes	84
6.17	Fault notification delivery ratio with different buffer sizes	85
6.18	Remote reports delivery ratio with different buffer sizes	85
6.19	Fault notification average delay	86
6.20	Remote reports average delay	87
6.21	Fault notification delivery	88
6.22	Remote reports delivery ratio	88
6.23	Fault notification average delay	89
6.24	Remote reports average delay	89
6.25	Fault notification delivery	90
6.26	Remote reports delivery ratio	91
6.27	Delivery delay for configuration messages	92
6.28	Delivery probability for configuration messages	92

List of Tables

2.1	Vehicular networks applications' classes	18
3.1	Main DTN management solutions	27
3.2	Main clustering solutions for vehicular networks	32
5.1	General parameters of the simulations	63
5.2	Simulation parameters for the VDTN clustering techniques	66
5.3	Simulation parameters used for the monitoring evaluation	69
5.4	Simulation parameters used for evaluating VDTN configuration techniques	69

Contents

Acknowledgments	xi
Resumo	xv
Abstract	xvii
List of Figures	xix
List of Tables	xxi
1 Introduction	1
1.1 Problem Definition and Objectives	4
1.2 Research Methodology	4
1.3 Thesis Contributions	5
1.4 Thesis Organization	5
2 Basic Concepts and the VDTN Management Problem	7
2.1 Network Management Overview	7
2.1.1 Management Approaches	8
2.1.2 SNMP: A Traditional Management Architecture	10
2.2 Delay-Tolerant Network	11
2.2.1 Contacts in DTN	12
2.2.2 DTN Architecture and Bundle Protocol	14
2.2.3 DTN Management	15
2.3 Vehicular Networks	16
2.3.1 Application Classes	17
2.3.2 Architecture and Protocols	18
2.3.3 Vehicular Delay-Tolerant Networks	20
2.4 The VDTN Management Problem	20

2.4.1	Network Model	21
2.5	Chapter Conclusions	22
3	Related Work	23
3.1	Methodology	23
3.2	Delay-Tolerant Network Management	24
3.3	Vehicular Network Management	27
3.4	Vehicular Network Clustering Algorithms	28
3.5	Chapter Conclusions	33
4	The HE-MAN Architecture	35
4.1	HiErarchical MANagement (HE-MAN) Architecture for Vehicular Delay-Tolerant Networks	36
4.2	Clustering Algorithm	39
4.2.1	General description of the algorithm	40
4.2.2	Algorithms' details	41
4.3	Monitoring Support	46
4.3.1	Publisher/Subscriber Model	46
4.3.2	Local and Remote Monitoring	47
4.4	Configuration Support	48
4.4.1	Local Configuration	49
4.4.2	Remote Configuration	49
4.5	Case Studies	51
4.5.1	Case Study One: Platoon of Autonomous Vehicles	51
4.5.2	Case Study Two: Bus Fleet Monitoring	52
4.5.3	Case Study Three: Transmission Status Propagation System . .	53
4.6	Chapter Conclusions	54
5	Evaluation: Methodology and Preparation	57
5.1	Methodology	57
5.2	Objectives	58
5.2.1	Objectives for the clustering algorithm	58
5.2.2	Objectives for HE-MAN monitoring	58
5.2.3	Objectives for HE-MAN configuration	59
5.3	Metrics	59
5.4	Mobility trace selection	60
5.5	Simulation environment and configuration	63
5.5.1	Clustering algorithm	65

5.5.2	HE-MAN monitoring	67
5.5.3	HE-MAN configuration	68
5.6	Chapter Conclusions	69
6	Evaluation: Results and Analyses	71
6.1	Clustering algorithm evaluation	71
6.1.1	Tunning of the configurable parameters in HE-MAN's clustering algorithm	71
6.1.2	HE-MAN's clustering algorithm performance analysis	77
6.2	Monitoring solution evaluation	82
6.2.1	Analysis of the impact of the size of buffer in HE-MAN Monitoring	83
6.2.2	Performance analysis of HE-MAN Monitoring	86
6.2.3	Analysis of the impact of DTN routing algorithms over VDTN monitoring	87
6.3	Configuration solution evaluation	90
6.4	Chapter Conclusions	92
7	Conclusions and Future Work	95
7.1	Limitations	97
7.2	On the Applicability of the Proposals on Different Scenarios	97
7.3	Future Work	98
7.4	Publications	99
	Bibliography	101

Chapter 1

Introduction

Various cities around the world experience increase in the number of vehicles travelling on their roads, arising concerns on vehicular transportation safety and efficiency, among others. In order to overcome a number of limitations concerning vehicular travelling in general, such as the vulnerability caused by the human response time or driver error, computational vehicular systems are being developed [Barba et al., 2012]. These systems demand inter-vehicular communication in order to allow vehicles to exchange data for supporting different types of applications, such as safety, transport efficiency or information/entertainment applications [Hartenstein and Laberteaux, 2008]. Vehicular Ad Hoc Networks (VANETs) are designed to provide communication among vehicles connected to other vehicles and to fixed equipment along roads, constituting an ad hoc network [Karagiannis et al., 2011].

VANET are becoming increasingly popular. Some countries are already conducting efforts to develop their own VANET systems, aiming to approach their specific needs. For instance, the National Highway Traffic Safety Administration, in the United States, released an advance notice of proposed rulemaking (ANPRM) proposing a Federal Motor Vehicle Safety Standard (FMVSS). The proposed standard requires vehicle-to-vehicle (V2V) communication capability for light vehicles and the creation of minimum performance requirements for V2V devices and messages, in order to primarily support advanced safety applications [National Highway Traffic Safety Administration, 2014]. Another example can be found in Brazil, where the ANTT (the Brazilian National Agency of Land Transports) intends to monitor all the land transports under the responsibility of this agency, automatically keeping record of commercial data, like passengers and travels, in order to improve the quality and reliability of information within the passenger transportation sector [Agência Nacional de Transportes Terrestres, 2014]. Monitoring parameters on ANTT's system include alarms of vehicle

breakdowns, the number of passengers on-board and the duration of mandatory driver breaks during long trips.

Traditional wireless networking approaches should not be effective for implementing VANETs, since vehicular networks suffer from frequent topology changes and constant node disconnection. This occurs because of the high speed and mobility of vehicles, together with the fact that VANETs commonly extend themselves over vast areas, which may lead to partitioning of the network. While the usage of mobile internet services (e.g., 3G/4G, GPRS, etc.) is increasingly growing and is an alternative for dealing with the mobility and the size of VANETs, service outages will still happen from time to time, and it is unrealistic to assume universally consistent coverage of such services in large and/or underdeveloped countries as well as in challenging areas (e.g., mountains, deserts, forests, etc.). Thus, a solution that is able to cope with partitioning in vehicular networks will always be superior to a solution that is not [Crowcroft et al., 2008]. Delay Tolerant Network (DTN) is a more suitable approach for the connectivity problem in VANETs, since it enables communications in environments with frequent disconnections and long transmission delays through store-carry-forward message switching. A VANET employing DTN protocols is usually referred as Vehicular Delay-Tolerant Network (VDTN) [Pereira et al., 2012].

Just like any production network, VDTN needs to be managed. Companies and institutions that own fleets of vehicles would benefit from a solution for monitoring and configuring not only the devices and services responsible for V2V and vehicle-to-infrastructure (V2I) communication, but also elements of vehicles that should be monitored - such as engines, tires, fuel consumption, etc. Moreover, traffic administration agencies can receive alarms from traffic management systems installed in the cars, in order to detect traffic jams and other types of road incidents in real-time. Finally, vehicle manufacturers can employ vehicles monitoring using the VDTN communication capabilities, aiming to ensure the proper effectiveness and performance of the vehicles' parts.

Network management techniques are needed in VDTNs for keeping the network operational and performing as intended. The task of managing a VDTN is highly necessary due to the high dynamicity of this type of network, together with the existence of critical systems that aim to prevent road accidents and improve traffic efficiency, among other features. These characteristics lead to the existence of two groups of systems (software and/or hardware) in a VDTN: delay-sensitive systems and non-delay-sensitive systems. A management solution for VDTNs must provide the means for both types of systems to be managed. Unfortunately it is unlikely that traditional management solutions will work properly in VDTNs. These solutions are able

to manage delay-sensitive systems, but they assume permanent existence of end-to-end paths between any pair of nodes and communication delays that are relatively low. However, these assumptions are not always met in VDTNs. As an example of this problem, most of current production networks use the Simple Network Management Protocol (SNMP) for constantly polling devices and services [Hussain and Gurkan, 2011]. This way of monitoring a network is almost impossible to be accomplished in a VDTN, since the long delays and constant disconnections would make it unlikely for a manager to send/receive polling messages in a regular basis. Issuing configuration commands to a vehicle node in a VDTN is also problematic using current network management technologies, because there is a good chance of the message arriving too late in its destination, which might compromise the validity of the issued configuration, or the command not arriving at all. Currently one can find a relatively small number of works in the literature introducing architecture proposals for the management of DTNs. Moreover, the vast majority of these proposals are either work in progress or focused on the management of other types of DTN other than VDTNs.

Architectures conceived for other types of DTN (e.g., deep-space DTN) are also not suitable for many monitoring needs of VDTNs, since a number of applications, such as the safety-related applications, require management operations with particular characteristics such as very strict delay constraints in order to ensure timely response for detected problems. For instance, the intersection collision warning application deal with messages being issued by neighboring vehicules every 100ms [Papadimitratos et al., 2009]. An eventual fault in such application must be detected and dealt with in a matter of miliseconds, or a few seconds at most. Not coping with this delay constraint may lead to road accidents caused, among other things, by an untreated fault in the intersection collision warning application. Therefore, there are considerable differences between a management architecture for DTN and a management architecture for VDTN.

One way to accomplish the management of delay-sensitive systems in VDTNs is to establish managers within every “island” composed of connected nodes. These established managers in connected groups would be subordinate to a superior manager in the network, forming a hierarchy of managers. Thanks to the low latency of the communication within connected groups, the established managers are able to successfully monitor and configure delay-sensitive systems, as it will be further explained along this thesis. It is important to notice that while other types of network may share characteristics similar to the ones found in VDTNs, such as intermitent connectivity and the existence of delay-sensitive systems, this thesis will focus on the problem of managing vehicular delay-tolerant networks.

1.1 Problem Definition and Objectives

This thesis presents an investigation on the management of Vehicular Delay-Tolerant Networks. The research question we aim to answer with this thesis is the following: *How can one effectively manage Vehicular Delay-Tolerant Networks?* The main objective of this PhD research can be stated as “*to design a set of models and algorithms to manage Vehicular Delay-Tolerant Ad Hoc Networks (VDTNs)*”. Moreover, we formulate the following specific objectives for our research:

1. To review the literature regarding the management of DTNs and vehicular networks;
2. To formalize the problem of VDTN management;
3. To design an architecture that manages VDTNs;
4. To develop techniques for monitoring devices and services;
5. To develop techniques for configuring devices and services;
6. To evaluate the effectiveness and performance of the presented proposals.

1.2 Research Methodology

The research presented in this thesis was divided into four major phases, described below:

1. **Study of the literature:** the first phase comprehended the study of the literature regarding Network Management, Delay-Tolerant Networks (DTNs) and Vehicular Ad Hoc Networks (VANETs), in order to acquire the necessary knowledge for conducting the other steps of the research;
2. **Problem definition and specification:** after that, an investigation on the management of Vehicular Delay-Tolerant Networks (VDTNs) was carried out, in order to devise a more formal specification of the problem to be approached;
3. **Design:** the next phase was the design of an architecture and protocols for the management of VDTNs;
4. **Evaluation:** finally, there was an implementation and evaluation phase, where the designed proposals were materialized in the form of simulations in order to validate our proposals.

1.3 Thesis Contributions

In this section the main contributions of this thesis are presented.

1. **Literature review and analysis of the VDTN management problem:** the thesis presents an overview of the VDTN management problem, highlighting the current open research questions. In the light of this review, a VDTN hierarchical management model is defined, which describes the most important challenges pertaining the VDTN management problem;
2. **A new management architecture for VDTNs:** a new management architecture that provides monitoring and configuration solutions for VDTNs;
3. **A set of new algorithms for hierarchically organizing the management topology in a VDTN:** the organization of the VDTN management topology in a hierarchical structure is a key requirement for the remainder of the proposals described in this thesis. Because of this, a set of algorithms was designed for organizing a VDTN in connected groups;
4. **New monitoring and configuration techniques:** techniques are proposed to approach the main needs of VDTNs concerning monitoring and configuration, in order to allow near real-time management as well as to improve the reliability of remote management tasks in VDTNs.

1.4 Thesis Organization

This thesis is organized as the following. Chapter 1 provides an introduction to the theme of this thesis, stating the main motivations behind this doctorate work and defining the objectives, methodology and contributions of the research. Chapter 2 presents a set of basic concepts and definitions regarding Network Management in general, Delay-Tolerant Networking, and Vehicular Delay-Tolerant Networks. At the end of the Chapter a more formal definition of the VDTN management problem is presented. Chapter 3 presents a review of the literature concerning the main topics related to this thesis, namely Delay-Tolerant Network management, Vehicular Network management and Clustering algorithms. The proposal of a hierarchical architecture for VDTN management is presented in Chapter 4. The chapter also describes algorithms for clustering, monitoring and configuration in VDTNs that are part of the proposed architecture. Chapter 5 provides a description of the simulations used in the evaluation of the algorithms for clustering, monitoring and configuration in VDTNs. The results

obtained through the simulations are presented and analyzed in Chapter 6. Finally, Chapter 7 concludes this thesis presenting the main conclusions of this research, a discussion of its limitations, and possible applications outside the VDTN scenario. It also presents comments on the publications resulted from this work, and future work.

Chapter 2

Basic Concepts and the VDTN Management Problem

This chapter reviews basic concepts of this thesis. At the end of the chapter, a model formalizing the main aspects of the Vehicular Delay-Tolerant Network considered in this doctorate research is proposed and discussed.

2.1 Network Management Overview

A typical computer network is composed of a series of complex hardware and software components interacting with each other through messages. Devices and software may malfunction, users might cause errors (intentionally or not), links might suffer interference from the external environment, among a number of other things that can go wrong and may interrupt the normal functioning of a network. As a network grows larger, the probability of these problems happening also increases, because of the growth of the number of elements involved in the communication process.

Network management refers to the activities, methods, procedures, and tools related to the operation, administration, maintenance, and provisioning of networked systems [Clemm, 2006]. In order to do so, management solutions employ a set of pieces of hardware and software to help network administrators to control the network. Network management tools help the management by relieving the network administrator of dealing with complex tasks, such as accessing network nodes one by one for updating specific configuration parameters, dealing with vendor specific hardware/software functionalities or manually aggregating large sets of data in order to determine the cause of a problem. Nowadays network management plays a crucial economic role, since to-

day's business applications depend on reliable, secure, and well-performing networked computer infrastructures [Ding, 2009].

2.1.1 Management Approaches

The placement of management software and hardware can be done in different ways, according to different network management approaches. Such approaches influence the flow of communication in the management system. Because of this, management approaches have to be carefully chosen, considering the particularities of each type of network.

Traditional network management solutions, such as the SNMP architecture, use a **centralized** approach. In other words, the management station is the sole responsible for managing all the other nodes in the network. The centralized management approach is illustrated in Figure 2.1a. One advantage of this approach is the convenience provided by the centralization of the data collected from the agents in a single network entity, which perform functions based on a view of the whole network. On the other hand, a central, unique point acting as the sole network manager allows the existence of a single point of failure in the management architecture, which means that if the network manager fails for some reason, the entire network's management system would be inoperative. Moreover, the centralized approach cannot scale easily, since a high number of agents communicating with a single management station can lead to the management station not being able to handle all the necessary communication with the agents.

The **distributed** approach, illustrated in Figure 2.1b, is another way to organize the network management elements, where multiple managers communicate and cooperate with each other in a peer-system [Kahani and Beadle, 1997]. In this approach it is possible to have multiple managers, or even make every node act as both agent and manager, forming a fully-distributed management. The main advantage of this approach is the potential increase of the reliability, robustness and performance of the management system due to the redundancy provided by the multiple managers spread over the network. However, this type of management requires complex algorithms to coordinate the managers. This might increase the overhead of the management solution and make the design of new management applications more complex.

A middle-term solution is the **hierarchical** approach, which involves the existence of multiple managers hierarchically organized, using the concepts of "Managers of Managers" (MOM) [Kahani and Beadle, 1997]. In this approach, a manager in a higher level manages the managers that are in the lower levels of the hierarchy, as illustrated

in Figure 2.1c. One important instance of the hierarchical management approach is the Management by Delegation (MbD), which uses the concept of elastic server. Elastic server is a concept in which a higher level manager is able to extend the features of lower level managers by sending scripts that add new delegated functionalities to the receiving manager [Goldszmidt, 1993].

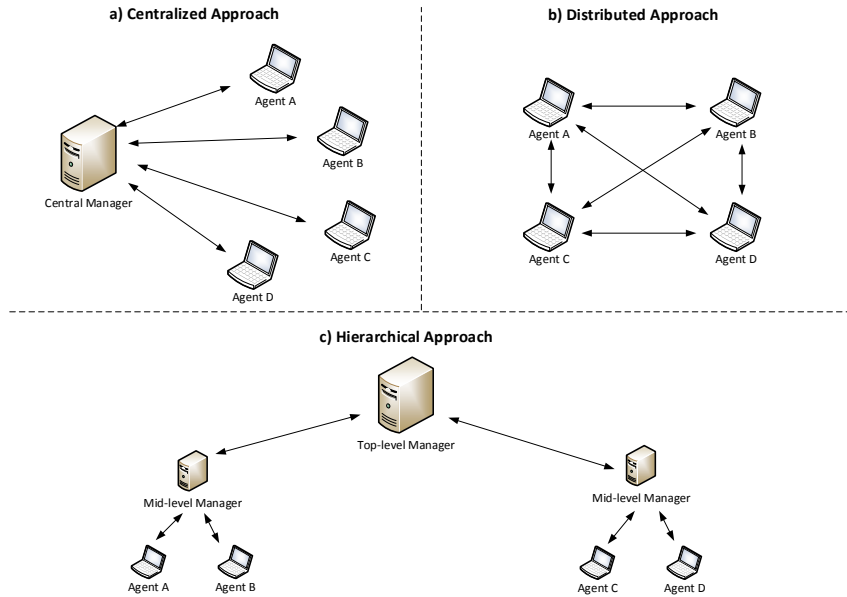


Figure 2.1. Network management approaches

This thesis employs the taxonomy defined above, but there are other taxonomies available in the literature. The main taxonomies were proposed by Martin-Flatin *et al.* [Martin-Flatin et al., 1999] and Schoenwaelder *et al.* [Schoenwaelder et al., 2000]. Martin-Flatin *et al.* [Martin-Flatin et al., 1999] proposed a simple and an enhanced taxonomies. The former is based on the organization criterion and divides the network management approaches in centralized, weakly distributed hierarchical, strongly distributed hierarchical, and strongly distributed cooperative. Schoenwaelder *et al.* [Schoenwaelder et al., 2000] introduced a taxonomy very similar to the simple version of Martin-Flatin's. However, the authors categorized the network management approaches considering solely the number of managers and agents. This taxonomy has four management categories: centralized, weakly distributed, strongly distributed, and cooperative. The main difference between the main network management taxonomies available in the literature and the taxonomy used in this thesis is the amount of categories in which the networks can be divided. Here, only three categories are considered (centralized, distributed and hierarchical). This reduced number of categories allowed

a simple yet effective exploration of how each category would suit VDTN management. Taxonomies with a higher number of categories will be more useful in future investigations, in order to fine tune the identification of management approaches that are viable for VDTNs.

2.1.2 SNMP: A Traditional Management Architecture

The current *de facto* standard for TCP/IP network management is the Simple Network Management Protocol (SNMP) [Cerroni et al., 2015]. SNMP is a set of network management specifications created more than 25 years ago in the RFC 1157 [Case et al., 1990], which includes a protocol, data structures, and associated concepts [Stallings, 1998]. The concepts and the network organization popularized by SNMP are still widely used nowadays.

The management model employed by SNMP is composed by the following fundamental elements [Stallings, 1998]:

- **Management Station:** also known as **manager**, it is the element that manages the network. These stations have, at least, a set of management applications (for data analysis, failures recovery, etc.), an interface that allows the operator to manage the network, the ability to realize the decisions of the operator, and a database storing all the collected management data;
- **Management Agent:** also known as **agent**, this component is installed in the network devices in order to make them manageable. Agents interact with the pieces of hardware and software that compose the network device in order to provide responses to the requests originated from a network manager;
- **Management Information Base (MIB):** it is the collection of the objects that represent the many aspects to be managed in an agent. These objects are variables in their essence, and these variables are read by network managers when an agent needs to be monitored, and in many cases the managers can also update the values of these variables in order to configure something in the managed node. These objects are divided into groups (e.g., System, Interfaces, IP, ICMP, UDP, etc.), and these groups provide a) means of assigning object identifiers, and b) a method for SNMP agents implementations to know which objects must be implemented [McCloghrie and Rose, 1988];
- **Network Management Protocol:** unites the management station and the management agents. It has three functionalities: **get method**, which allows the

management station to request the values of objects in the managed agents; **set method**, which can redefine values for the objects of the managed agents; and the **trap** functionality, which allows the agent to notify the management station about the occurrence of relevant events [Case et al., 1990].

Figure 2.2 presents the overall architecture of a typical TCP/IP network environment managed via SNMP.

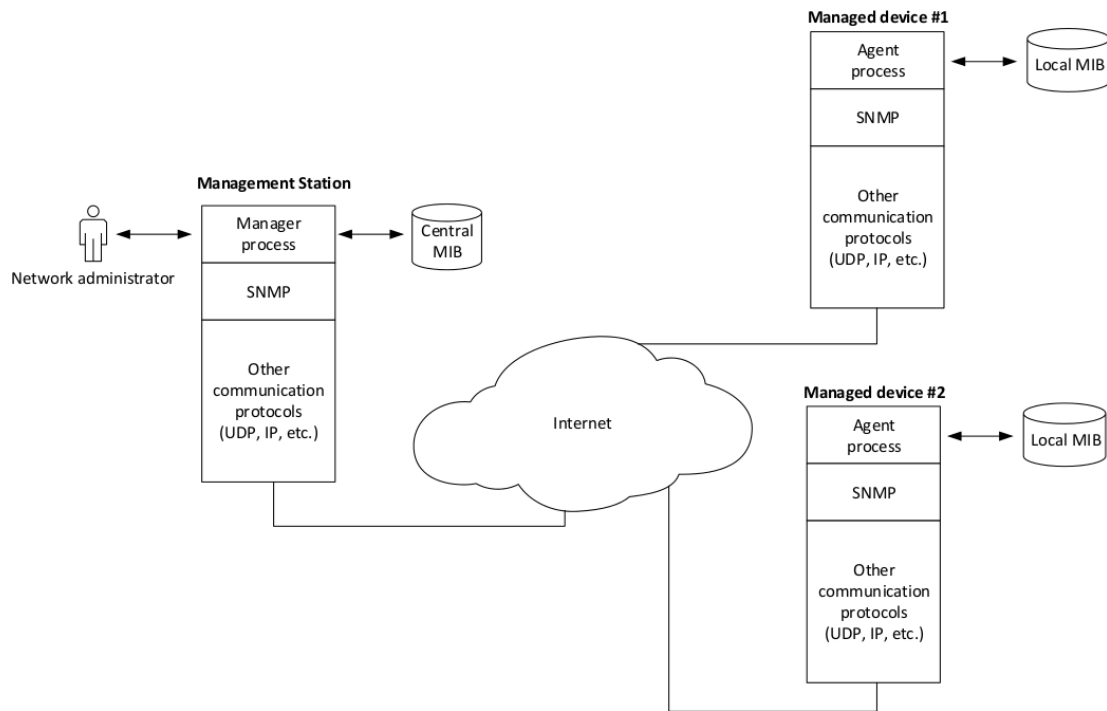


Figure 2.2. Example of the SNMP network management architecture [Stallings, 1998]

2.2 Delay-Tolerant Network

The class of challenged networks is characterized by latency, bandwidth limitations, error probability, node longevity, or path stability that are significantly worse in comparison with today's Internet. Despite the increasing importance of challenged networks, it is not well served by end-to-end TCP/IP model, since some of the assumptions on which such protocols were built on are not met. For example, Internet protocols assume that an end-to-end path between any pair of nodes will be available during a communication session, or that the delays involved in the transmissions are always relatively low [Fall, 2003].

Delay Tolerant Network (DTNs) is an architecture conceived for provisioning communication in environments with frequent disconnections and long data delivery delays [Cerf et al., 2007]. Although DTN technologies have been initially developed for interplanetary communications, many characteristics of these networks can also be found in terrestrial applications, especially in scenarios where communication is asynchronous and subject to long message delivery delays [Ivansic, 2009]. These conditions usually appear in sparse and/or mobile networks, or in networks where links have high error rates.

For example, networks based on *data mules* use vehicles (e.g., bus, bicycle, unmanned aerial vehicle, etc.) that regularly travel between two or more remote locations to carry messages from one place to another in order to provide connectivity among the visited locations [Grasic and Lindgren, 2014]. In such networks, the vehicles are likely to go through high periods of disruption during the trips between both locations, thus being able to deliver bundles only when inside the communication range of the villages. Another example is an emergency network, which is an ad hoc network deployed in disaster scenarios where the existent telecommunication infrastructure is lost or inoperative [George et al., 2010]. In such networks, the communication among the devices being used by the disaster mitigation units can be broken frequently for a number of different reasons, such as a sparse node deployment or the presence of debris disrupting the devices' communication links.

Figure 2.3 presents a typical DTN dynamic scenario, where the circles represent network nodes, the solid lines represent existing communication links between a pair of nodes, and the dashed lines represent disrupted communication links. For example, at Instant 1 the link between **a** and **b** is up, while the link between **a** and **c** is down. The term **contact** is commonly employed in the DTN literature to refer to the existence of connectivity between a pair of nodes. For instance, there is a contact between **a** and **b**, but not between **a** and **c** at Instant 1. The remainder of Figure 2.3 (Instants 2, 3 and 4) illustrates the volatile nature of contacts through time in a DTN environment, with contacts getting up and down as the time progresses.

2.2.1 Contacts in DTN

Contacts between DTN nodes can be classified into several categories, according to their predictability and to whether some action is required to bring them into existence. In this thesis, we classify the contacts as follows [Cerf et al., 2007]:

- **Persistent Contacts:** this type of contact is always available, such as a DSL (Digital Subscriber Line) or cable modem Internet connection;

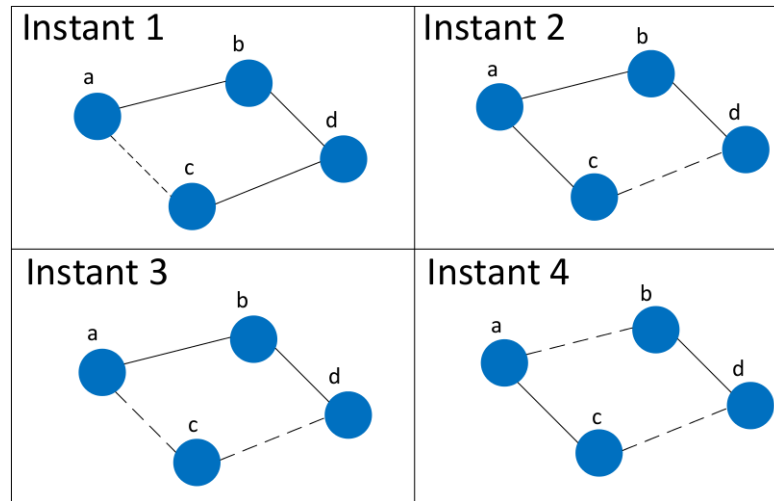


Figure 2.3. Example of the topology of a DTN through time

- **On-demand Contacts:** these contacts are similar to the persistent contacts, but they require some action to be instantiated, remaining available until the contact is explicitly terminated. A dial-up connection can be an example of this class of contact;
- **Intermittent - Scheduled Contacts:** this type of contact is predictable, that is, it is known beforehand when the contact should occur, and for how long. A typical example is the links with low-earth orbiting satellites;
- **Intermittent - Opportunistic Contacts:** this class is composed of the contacts that are not scheduled, but rather present themselves unexpectedly. This type of contact can be very common in vehicular networks, where two cars can unexpectedly get into communication range of one another to form a contact;
- **Intermittent - Predicted Contacts:** this type of contact is not based on a schedule, but is rather based on predictions regarding the likelihood of a contact occurrence and duration according to historic data or some other information. This kind of contact can happen in Pocket Switched Networks (PSN), where data is transferred between mobile users' devices using human mobility [Hui et al., 2005]. In this type of network, some contacts can be predicted by the routine followed by the users (e.g., people travelling to their offices in working days).

2.2.2 DTN Architecture and Bundle Protocol

The DTN architecture employs store-carry-and-forward message switching to counter frequent disconnections and long delays. In this type of switching, a node is able to receive a message, and if the message is not addressed to that specific node, it is stored in a persistent memory to be forwarded in the future. As the node moves and gets in range with other nodes, it may forward the message (or a copy of it) to another intermediate node, or forward it directly to the final destination of the message. In this sense, the type of communication provided by a DTN is similar to electronic mail [Cerf et al., 2007].

In order to provide the intrinsic capabilities of the DTN architecture, the Bundle Protocol (BP) was specified in RFC 5050 [Scott and Burleigh, 2007]. The key features of this protocol are the following:

- Custody-based retransmission, where the “custodian” or “responsible entity” is a node required to keep the bundle in persistent memory until another custodian has received it successfully [FALL and FARREL, 2008];
- Ability to cope with intermittent connectivity;
- Ability to take advantage of scheduled, predicted, and opportunistic connectivity (in addition to continuous connectivity);
- Late binding of overlay network endpoint identifiers to constituent internet addresses. The late binding is the opposite of the early binding that is typical of Domain Name System (DNS) in the Internet, where the symbolic name of a resource is bound to a specific network address before a message is forwarded to its destination. In late binding, a message is forwarded to another node by matching the name of the destination against name entries that are found in the DTN nodes’ routing directives [FALL and FARREL, 2008].

The Bundle Protocol is placed between the application layer and the transport layer, as illustrated in Figure 2.4. Therefore the DTN is implemented as an overlay network, with the Bundle Layer tying together different types of network technologies across the entire network. The Bundle Layer is connected to the underlying network-specific protocols through a convergence layer, which deals with the specifics of each network technology while providing a consistent interface to the Bundle Layer [Cerf et al., 2007].

The messages in a DTN are usually referred as bundles, and these bundles are stored and forwarded between nodes through the Bundle layer. The communication

between any pair of DTN nodes occurs in simple sessions with minimal or no round-trips, with whole bundles being transmitted at once in the Bundle Layer. Bundles are the result of the transformation that Application Data Units (ADUs) suffer in the Bundle Layer. ADUs have arbitrary length, although they are subject to implementation limitations. The bundle contains two or more “blocks” of data, with each block containing either application data or other information used to deliver the bundle to its destination. Bundles may be fragmented, and the resulting fragments are bundles themselves, with the possibility of being further fragmented. Finally, bundle sources and destinations are identified by Endpoint Identifiers (EIDs), which identify the original sender and final destination of bundles, respectively [Cerf et al., 2007].

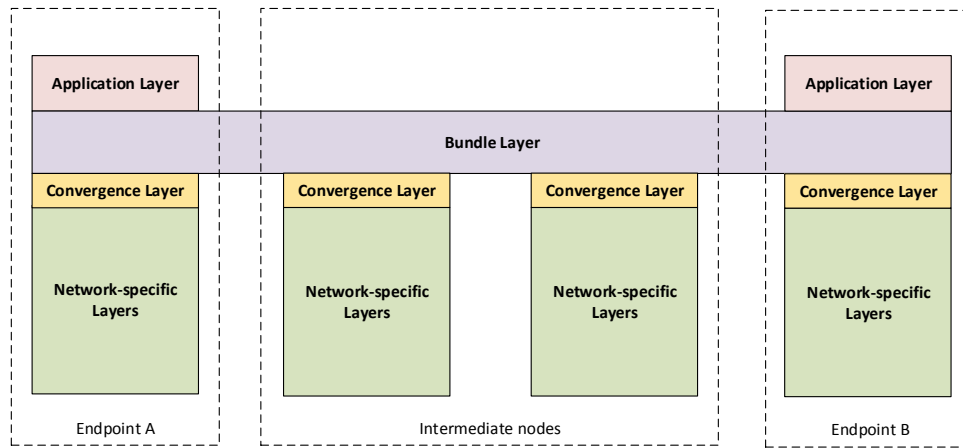


Figure 2.4. Bundle Layer in an example of DTN communication

Nowadays there are various new researches being published in the DTN area, covering areas such as mobility models, routing, data dissemination, security and applications. Moreover, there are also some interesting trends regarding intersections between DTN and other areas, such as DTN associated with social characteristics of people (social behavior) in order to estimate the probability of contacts. Another trending area is Opportunistic Computing, which opportunistically distributes and combines services to users, based on said users interests and needs [Conti and Giordano, 2014].

2.2.3 DTN Management

Traditional network management solutions cannot be employed in DTNs. One important characteristic in the management model employed by traditional networks is the “chatty” form of communication (e.g., SNMP). Traditional TCP/IP networks do not take into account loss of messages, long delays or constant disconnections, and as such managers expect a prompt response by the agents. Although this model works

fine in the networks it was designed for, it should not perform well in more challenged networks, such as DTNs. Some problems arise as a result of the usage of traditional management techniques on DTNs, such as:

- The polling process being compromised by managers' get-request messages not being delivered to the agents, or their respective responses not getting back to the managers on time or at all;
- Monitoring information, configuration commands and notification messages (alarms) taking long periods of time towards the managers, in such a way that it would be too late for a manager to correct the network behavior.

The requirements for managing DTNs are not fully known yet, and this is currently an open research question. W. Ivansic points to some directions for the development of DTN management architectures, without specifying how exactly these directions could be realized [Ivansic, 2009]. According to the author, one possible solution is to divide the DTN management tasks into local management and remote management. Regions where local management is possible will likely benefit from traditional management based solutions. On the other hand, remote management occurs when it is necessary to manage one or more nodes via a DTN connection. Since this communication will use the bundle protocol, frequent link disconnections and long propagation delays will probably make the message delivery much more time-consuming. As for the configuration functions, it is necessary that a DTN management solution has sufficient autonomic capabilities to validate an operation before its deployment, as well as perform a rollback operation to the last known safe state whenever necessary. This thesis agrees with the general directions pointed by W. Ivansic, and make efforts to instantiate the general recommendations into algorithms for vehicular networks.

2.3 Vehicular Networks

The wide adoption of wireless communication technologies and the increasing interest of both academia and industry in developing vehicular wireless communication made Vehicular Ad Hoc Networks (VANETs) a promising field of research. Improvements on vehicle and road safety, traffic efficiency, and convenience/comfort to both drivers and passengers are among the main incentives behind VANET research [Zeadally et al., 2012]. Several automobile manufacturers and research centers are conducting investigations on protocols, applications and the use of inter-vehicle communication for the establishment of VANETs [Liu et al., 2016].

2.3.1 Application Classes

Applications in vehicular networks can be divided into three major classes from the customer's perspective: safety-oriented, traffic management applications, and commercial applications [Bai et al., 2006]. They are briefly described:

1. **Safety applications:** applications in this category monitor the surroundings of the vehicle and exchange messages with other vehicles, to assist drivers to avoid potential dangers. Some examples of applications are “Stopped or Slow Vehicle Advisor”, “Emergency Electronic Brake Light” and “Road Hazard Condition Notification”. Typical safety applications require high packet delivery rates and low latency communication, because the safety messages need to arrive at their destination as fast as possible, in order to prevent accidents to happen;
2. **Traffic Management applications:** this class of applications provides a better distribution of vehicles on the roads, in order to reduce traffic jams and travel time. Such objective is usually achieved by roadway infrastructure and vehicles exchanging traffic information. “Congested Road Notification” and “Free Flow Tolling” are examples of applications in this class. Some applications in this class require low latency communication in order to support real-time monitoring of the traffic conditions, while others are resilient to delay, since they require only historical data of the traffic (e.g., for future traffic planning);
3. **Commercial applications:** this class of applications improves driver productivity, entertainment, and satisfaction. Some examples are “Service Announcements” and “Real-time Video Relay”. The network requirements for this class are highly dependent on the application itself: it can range from delay-tolerant commercial applications, such as mobile advertising, to time sensitive applications, like real-time voice communication.

The main characteristics of these classes are summarized in Table 2.1, including message packet format (from the perspective of user benefit), network-layer routing protocol, and network routing triggering (whether by detected events, by a periodic-based mechanism or by an user-initiated interaction) [Bai et al., 2006]. Message packet format is an abstract definition determined by the type of application. It can be categorized as light-weight short messages or traditional heavy-weighted IP messages. The network-layer routing protocol is clarified according to reachability and recipient patterns. Finally, routing protocol triggering defines how messages forwarding is initiated, whether by events, by periodic broadcasted beacons, or by the own users of

the applications.

Network Attribute	Safety	Convenience	Commercial
Message packet format	Light-weighted	Light-weighted	Heavy-weighted
Network-layer routing protocol	Multi-hop geocast and single-hop broadcast	Multi-hop geocast, single-hop broadcast and unicast	Multi-hop geocast, single-hop broadcast and unicast
Routing protocol triggering	Event-driven and periodic-driven	User-initiated	User-initiated

Table 2.1. Vehicular networks applications' classes

2.3.2 Architecture and Protocols

Communication in VANET is standardized by the IEEE 802.11p standard and the IEEE 1609 WAVE (Wireless in Vehicular Networks) standards, which operate in the Dedicated Short Range Communications (DSRC) in the United States [IEE, 2014]. The main purposes of each of the standards are the following [Uzcategui et al., 2009]:

- IEEE 802.11p specifies the PHY and MAC layers for vehicular networks;
- IEEE 1609.1 describes the interaction of Onboard Units (OBUs) with both more limited computing resources and more complex processes running outside the OBU;
- IEEE 1609.2 describes the format and processing of secure messages in vehicular networks;
- IEEE 1609.3 provides addressing and routing services within a WAVE system;
- IEEE 1609.4 describes enhancements to the 802.11p MAC, in order to support multichannel operation.

WAVE uses a multi-channel concept to separate the traffic of each application type. Messages are prioritized through different Access Classes (ACS), having different channel access settings. Thus, high priority safety messages are able to reach their destination in a timely and reliable fashion, even in a dense network [Eichler, 2007].

Vehicular networks combine communication with fixed infrastructure, known as Roadside Units (RSUs), and ad hoc communication among vehicles equipped with

OBUs, as illustrated in Figure 2.5. Communication between vehicle and fixed infrastructure is commonly referred as V2I (Vehicle-to-Infrastructure), while communication between vehicles is referred as V2V (Vehicle-to-Vehicle). V2V communication has a number of advantages over V2I communication [Su and Zhang, 2007]. For instance, V2V-based VANETs are more flexible and independent of roadside conditions and other forms of infrastructure, such as the mobile phone network, which can be particularly interesting for developing countries or remote rural areas. V2V is also less expensive than V2I-based networks, since there is no need for investments on roadside infrastructure. The term V2X (Vehicle-to-X) is a reference to a more general (hybrid) form of communication involving either vehicles or fixed infrastructure. WAVE supports two protocol stacks, both using the same physical and data-link layers: the Internet Protocol version six (IPv6) and a protocol known as WAVE Short-Message Protocol (WSMP). The reasoning behind two different protocol stacks is the need to accommodate high-priority, time-sensitive communication, as well as traditional and less demanding exchanges, such as Transmission Control Protocol (TCP) or User Datagram Protocol (UDP) transactions [Uzcategui et al., 2009].

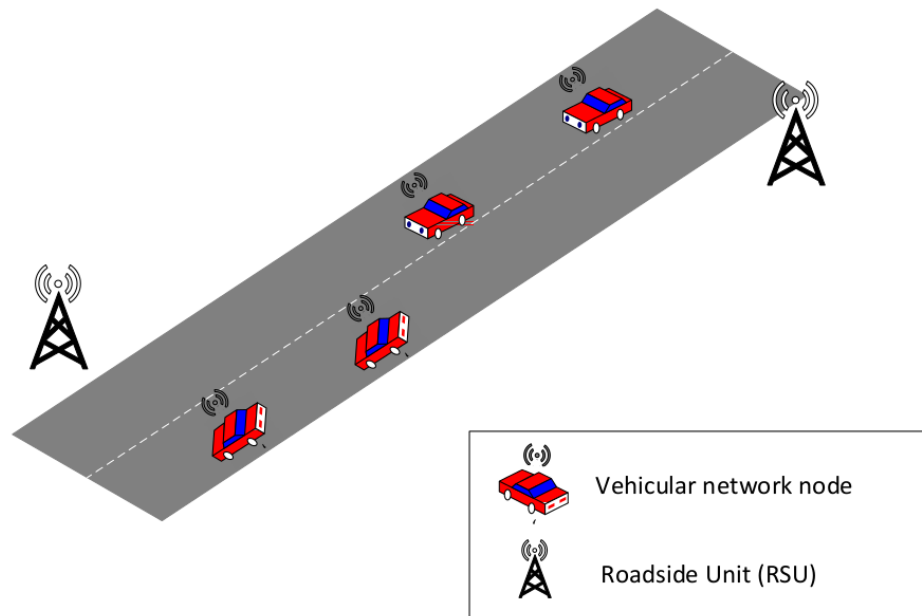


Figure 2.5. Example of a vehicular network

2.3.3 Vehicular Delay-Tolerant Networks

Because of the high mobility of the vehicular nodes and the vast area that a VANET can cover, partitions in the network are likely to occur, especially in areas where the nodes are sparsely distributed. Moreover, investments on some sort of communication infrastructure, such as a mobile phone network, can be inviable in most developing countries due to the cost of such infrastructure. In such cases, V2V communication is preferable because of its flexibility and low associated cost.

VANETs can benefit from the DTN architecture, due to its ability to cope with frequent link disconnections. Thus, a Vehicular Delay-Tolerant Network (VDTN) is a DTN where the nodes are composed by vehicles and RSUs [Pereira et al., 2012]. The main difference between VANETs and VDTNs is the ability to cope with sparse networks: while VANET protocols assume fully connected networks, VDTNs are more adapted to sparse networks and topology changes [Zhang and Wolff, 2010]. In VDTNs the contacts are mostly opportunistic (specially when no road side infrastructure is used), and the transmission delay between two nodes that are not in contact with each other can be considerably high, which has to be taken into account by VDTN application developers.

2.4 The VDTN Management Problem

DTNs and Vehicular DTNs have many common characteristics, as one would expect. However, VDTN management also carries some important aspects that are unique to its domain. The most important aspect is the fact that safety applications are very sensitive to communication delay, which imposes a serious challenge for the management of those applications. Critical messages arriving late may result in delayed detection of important events, which can result in road accidents. Vehicle proximity (used in collision avoidance systems), danger warnings and vehicle with faulty sensor warnings are examples of critical messages that are highly sensitive to long communication delay. Because of this, VDTN management systems need to support low-latency management in a DTN environment, which can be a challenging problem that is yet to be solved.

Since low-latency communication in VDTN can only be achieved when nodes are within local communication range from one another, a VDTN management solution must gather information on the current topology of the network in order to take advantage of local management opportunities. Because of this, VDTN management systems need to know data such as the immediate neighborhood of a node at a given moment, or the estimated duration of a given contact, in order to adapt the behavior of the

management solution to these changes in the network's topology. In more traditional networks, the need for a management system to know details on the topology of the network is less important, since in these networks one can always assume the existence of a route between any pair of nodes, eliminating the concern about the ability of the management system to communicate with other nodes.

In order to allow a better delimitation of the scope of this thesis, a network model will be provided. In the next chapters of this thesis some proposed concepts will be presented in terms of the definitions described in the model presented in this chapter, hence the importance of the next subsection.

2.4.1 Network Model

This subsection presents the formal model of VDTN considered in this thesis. The model below is based on the work of Ferreira *et al.* [Ferreira, 2002].

Definition 2.4.1. Evolving Graphs: Consider a graph $G = (V_G, E_G)$, where V_G is the set of vertices and E_G is the set of all possible edges, and $S_G = G_1, G_2, \dots, G_\tau$ ($\tau \in \mathbb{N}$) an ordered set representing a sequence of the subgraphs of G such that $\bigcup_{i=1}^\tau$. The system $\mathcal{G} = (G, S_G)$ is called an evolving graph [Xuan et al., 2003]. In such graph, the vertices represent the nodes in the VDTN (vehicles and RSUs), and an edge between two vertices represent a contact between the two nodes. Two vertices are considered adjacent in \mathcal{G} if and only if they are adjacent in some G_i . The degree of a vertex in \mathcal{G} is its degree in E_G by definition. Evolving graphs can also be weighted, and the term timed evolving graph is used when the weights on the edges represent their available traversal time (i.e., the amount of time necessary for an edge to be traversed). In such cases, let $\mathcal{I} = [t_1, t_{\mathcal{I}+1}[\in \mathbb{T}$ be a time interval, where G_i is the subgraph in place during $[t_i, t_{i+1}[$.

Definition 2.4.2. Journeys: A route in \mathcal{G} consists of a path $R = e_1, e_2, \dots, e_k$ with $e_i \in E_G$ in G , and $R_\sigma = \sigma_1, \sigma_2, \dots, \sigma_k$ ($\sigma_i \in [1, \tau], \sigma_i \leq \sigma_j \forall i < j$) is a time schedule indicating when each edge of the path R is to be traversed. Then a journey $\mathcal{J} = (R, R_\sigma)$ in \mathcal{G} is defined. The concept of a journey is important in our research because it represents a path that a bundle must traverse in order to reach its final destination, and the size of the journey (i.e., the number of edges to be traversed during a journey between a pair of vertices) is usually related to the communication cost between the two nodes involved.

Definition 2.4.3. Delay time of a Journey: The delay time of a journey \mathcal{T} can be defined as $\sigma_k - \sigma_1$, where σ_k is the time when the last edge of the journey is traversed,

and σ_1 is the time when the first edge of the journey is traversed. This definition is relevant in the context of this thesis because a smaller time of a journey represents a more responsive management system.

Definition 2.4.4. Delivery ratio: Let N be the total amount of journey attempts towards vertices of the network, and $N_{\mathcal{J}}$ be the number of successful journeys on said network. The delivery ratio \mathcal{D} is defined as the fraction $N_{\mathcal{J}}/N$. One of the objectives of a VDTN management solution is to maximize the delivery ratio of management messages.

Definition 2.4.5. VDTN management architecture objective: The objective of the VDTN management architecture is defined as two-fold: to maximize the delivery ratio \mathcal{D} , and to limit the maximum delay of a journey to not exceed a constant k_i ($\mathcal{T}_{max} < k_i$), with $k_i \in K$, where K is the set of tolerated delays for each class of applications, and k_i is an element of this set regarding one specific class of applications.

2.5 Chapter Conclusions

This chapter presented concepts necessary for the proper comprehension of the remainder of this thesis. Network management techniques help network operators to keep their network performing as expected.

The Delay-Tolerant Network (DTN) architecture copes with frequent disconnections and long transmission delays that can occur in challenging environments. In Vehicular Ad Hoc Network (VANETs) the nodes are highly mobile and the network can extend itself over a vast area, which generally leads to connectivity issues between nodes. The usage of the DTN architecture in VANETs can solve the communication problems of these networks in scenarios of sparsely distributed nodes, and the combination of these two type of networks is commonly referred as Vehicular Delay-Tolerant Network (VDTN). VDTNs carry common characteristics with VANETs, but there are also aspects that can be found only in VDTNs, such as the necessity of providing specific solutions for the management of hardware and services that have delay-sensitive, like road safety applications. The particularities of Vehicular DTNs demand the design of new management techniques that will allow effective and efficient management specifically for these networks. Finally, the effective management of a VDTN was formalized using evolving graphs.

Chapter 3

Related Work

This chapter presents a review of the literature of the main areas related to this thesis: DTN management and Vehicular Network management. Clustering Algorithms are also reviewed due to the importance of these algorithms in the proposed management architecture.

3.1 Methodology

The search for the state of the art in this thesis used the prominent scientific search engines. Google Scholar¹ and Science Direct² engines were selected, since they are able to return results from several important scientific publishers, such as ACM, IEEE and Elsevier. Only works published in the last decade (i.e., between 2006 and 2016) were taken into account, in order to avoid reviewing research results that became obsolete as the result of newer published works providing improved versions of older results. Selected search queries for each surveyed area were employed systematically in the selected search engines, as following:

a) Delay-Tolerant Network Management: The results matching the query string below were taken into consideration:

((DELAY-TOLERANT NETWORK) OR (OPPORTUNISTIC NETWORK)) AND
((MANAGEMENT) OR (MONITORING) OR (CONFIGURATION))

More than 21,000 results were retrieved through the usage of the query string above. Works strictly regarding specific management tasks such as buffer management

¹<http://scholar.google.com>

²<http://www.sciencedirect.com>

only, power management only, among others, were dismissed for being too narrow to be pertinent to the objectives of this thesis.

b) Vehicular Network Management: The research query used was the following:

((VEHICULAR DELAY-TOLERANT NETWORK) OR (VANET) OR
(VEHICULAR NETWORK)) AND ((MANAGEMENT) OR (MONITORING) OR
(CONFIGURATION))

The usage of this query string resulted in more than 10,000 works being retrieved. Once again, management techniques that are applicable to a single aspect of vehicular networks, such as power management or mobility management were not taken into consideration for being out of the scope of this thesis.

c) Vehicular Network Clustering Algorithms: The research query used was the following:

((VANET) OR (VDTN) OR (VEHICULAR NETWORK)) AND (CLUSTERING)

In summary, clustering techniques designed for any variation of vehicular network were taken into consideration. The query string provided the retrieval of more than 3,000 results. Priority was given to works published in media with higher relevance, according to the Brazilian metric for measuring the quality of journals and conferences: Qualis. The related work research using the presented methodology was last updated on June 2016.

3.2 Delay-Tolerant Network Management

The ascension of the DTN research was followed by the perception that new network management techniques would be necessary for DTN. Since then, a number of proposals emerged in the literature. However, it is important to notice that the number of published works is relatively low, leaving plenty of research questions still open. This subsection reviews the works published so far regarding DTN management.

Birrane and Cole investigated DTN management employing a system engineering approach [Birrane and Cole, 2010]. The authors present implementation recommendations and current threads of research on: system architecture, data definition/usage, application architecture, and tool implementation. However, the authors do not provide any new concrete protocols or architecture for DTN management.

Peoples *et al.* proposed a network middleware called context-aware broker (CAB), supporting policies in deep space communication DTNs [Peoples et al., 2010]. The impact of the policy-driven model on transmissions, its ability to achieve autonomy and self-management, and the relationship between offered and received load were the aspects evaluated for the proposed middleware. However the authors' solution is presented only in a deep space communication scenario, without information on how their solution could be extended to other scenarios. In the opinion of this thesis' author, while the addition of a context-aware policy-based network management middleware could improve several management tasks in VDTNs, it would not satisfy near-real-time applications.

Sachin *et al.* extended the NETCONF to DTNs, to install, manipulate, and delete configurations in network devices [Sachin et al., 2010]. Their proposal was called Configuration Network Management Protocol (CNMP), and an initial prototype of this protocol was tested on a GNU Radio testbed. The authors discussed design trade-offs and their impact to the configuration of nodes and services in a DTN. They proposed a new NETCONF operation called `<verify>`, which verifies the correctness of operation in the nodes configuration environments. However, the design choices and implementation presented by Sachin *et al.* are still in an early stage. The research lacks details such as the specifics on the local policy rules checker, as well as a more extensive evaluation concerning how well their NETCONF extension scales in larger DTNs.

Papalambrou *et al.* proposed a method for monitoring DTNs based on the publisher-subscriber model [Papalambrou et al., 2011]. The proposed protocol is called Diagnostic Interplanetary Network Gateway (DING), and it was the first to implement DTN monitoring based on the publisher/subscriber model. In this protocol, the *subscriber* informs which data it wishes to receive. *Publishers* generate data, which is sent to the subscribed nodes. This paradigm has a much lower overhead in comparison with SNMP, since it requires only one subscription request for the reception of an arbitrary number of responses, while SNMP requires one request for each response. DING evaluation occurred in a small network called *planet-ece*, composed of three nodes based on the DTN2 implementation of the DTN Bundle Protocol [FALL and FARREL, 2008]. In this paper the authors gather monitoring data using scripts that interact with the operating system and the DTN2 implementation. The authors' proposal does not deal with the consequences of frequent disconnections and long delays to management operations. Moreover, it is not clear if the authors' solution scales well with the size of the DTN, since their evaluation process employed three nodes.

The Delay-Tolerant Network Management Protocol is proposed by the NASA

DTN program through an IETF Internet-Draft, and it is designed to support the network management functions of configuration, performance reporting, control, and administration [Birrane and Ramachandran, 2014]. DTNMP supports five desirable properties: Intelligent Push of Information, eliminating the need for round-trip data exchange in the management protocol; Small Message Sizes, requiring smaller contact time to be transmitted; Fine-grained, Flexible Data Identification, to precisely define what data is required; Asynchronous Operation, to avoid relying on session establishment or round-trip data exchange; and Compatibility with Low-Latency Network Management Protocols, to enable management interfaces between DTNs and other types of networks. Nonetheless, the document primary focuses are to present an overview of the DTNMP and to specify message flows and formats, without providing details on how the inner components of agents and managers should be implemented. Also, the DTNMP solution does not approach the specificities of VDTN management.

Torgerson [Torgerson, 2014] presented a suite of tools that provide network visualization, performance monitoring, and node control software for devices using the Interplanetary Overlay Network (ION) DTN implementation [Burleigh, 2007]. This suite, called Network Monitor & Control (NM&C) System, complements the DTNMP. The authors expected their suite of tools, together with DTNMP, to form the basis for flight operation using DTN. The limitation of this work is the fact that its proposals are tied to the deep space communication scenario. The authors do not provide information on how their proposals could be extended to other scenarios.

The Opportunistic Management Contact Estimator (OMCE) estimates the appropriate time to execute future management tasks considering distributed statistical analysis of past contacts [Nobre et al., 2014]. Additionally, the nodes share data with other nodes regarding their contacts to improve contact estimation using a P2P management overlay. Their solution was evaluated by both simulation experiments and a proof of concept implementation. Results showed that it is possible to improve the performance of monitoring tasks through the usage of the authors' proposals. However, this work relies on another overlay network on top of the DTN. The use of P2P over DTNs may increase the computational demands over the network nodes, which can lead to the authors' proposal being inadequate for scenarios where the devices have strict resource limitations. Unfortunately the authors did not provide insights on the computational costs of their proposal.

The Deutsches Zentrum Für Luft-und Raumfahrt (DLR) DTN working group proposed major changes to the current DTNMP reference implementation in order to better support commanding capability for the ION DTN reference implementation [Pierce-Mayer and Peinado, 2016]. The proposed changes include a strong-typing

Name	Management tasks	Target Scenario	Reference
CAB	Monitoring, Configuration and Alarms	Deep Space	[Peoples et al., 2010]
CNMP	Configuration	-	[Sachin et al., 2010]
DING 2011	Monitoring	-	[Papalambrou et al. 2011]
NM&C/DTNMP	Monitoring, Configuration and Data Visualization	Deep Space	[Torgerson, 2014] [Birrand and Ramachandran, 2014] [Pierce-Mayer and Peinado, 2016]
OMCE	Contact estimation for future management tasks	Opportunistic Networks	[Nobre et al., 2014]

Table 3.1. Main DTN management solutions

system, a RPC system, a middleware system, and a prototype implementation of an Abstract Data Model (ADM) that is responsible for network contact management. The authors managed to decouple the network management interface from any dependence on the DTN stack through command producers. These command producers take intermediate commands and outputs specific DTNMP commands which are required for a given DTN implementation or node type. However, the author’s proposals are strongly coupled to the ION reference implementation of DTN and its associated tools, which makes it harder to generalize these proposals to other scenarios.

Table 3.1 summarizes the main properties of the works regarding DTN management presented in this section. The table highlights the management tasks supported by the solutions, the type of scenario these solutions were designed for (if specified), and the references where the solutions are presented.

3.3 Vehicular Network Management

As already mentioned in this thesis, the problem of managing vehicular networks, like in the DTN management problem, brings new specificities that need to be approached by solutions specially designed to VDTNs. These works are reviewed in this subsection.

Mohinisudhan *et al.* incorporate SNMP-based control of vehicular components as well as performance monitoring for automobiles [Mohinisudhan et al., 2006]. The proposed management systems provides reliable, real-time communication. The framework uses the Controller Area Network (CAN) serial bus communications protocol to communicate between on-board SNMP agent embedded in a microprocessor with the manager that is usually located within the vehicle itself. While this proposal tackles monitoring and controlling diverse sensors and actuators inside a vehicle using a standardized framework, it is not applicable to inter-vehicle networks.

Ferreira *et al.* developed an SNMP-based management solution for VDTNs [Ferreira et al., 2014]. They claim that their solution supports load-related information collection, and provide a demonstration of this management application in a testbed.

Their work, however, does not control the delays involved in such transmissions, but instead it uses encapsulation of SNMP messages in DTN bundles.

Silva and Meira Junior proposed an algorithm for the deployment of stationary and mobile RSUs based on inputs of the density of vehicles along the road and the registered migration ratios among the urban cells of the network [Silva and Meira, 2015]. The authors also proposed a novel metric, called Delta Network, for measuring the connectivity experienced by vehicles. Although the authors' algorithm exploits vehicular network management, their techniques apply only to RSUs, leaving the vehicular nodes out of the scope of monitoring and configuration techniques.

Dias *et al.* designed a monitoring and configuration tool for VDTNs, called MoM [Dias et al., 2016]. This work considers a model of VDTN that employs out-of-band signal for providing separation between control and data plane. The employed VDTN model also uses two types of static nodes: terminal nodes, which act as access points to other nodes in the VDTN, and relay nodes, which have large store capacity and are usually placed in crossroads for increasing contact opportunities. MoM uses out-of-band signalling to collect statistical information about the performance of the nodes and to spread configuration messages across the network. Monitored statistics are addressed to terminal nodes, and whenever a problematic node is detected through the monitoring module of a terminal, it disseminates “advertisements” to other nodes of the VDTN in order to try to solve the problem. In this work, the authors rely on fixed infra-structure in the VDTN, which makes their solution more expensive in comparison to solutions that rely only in V2V communication. Moreover, although a network administrator is able to see in real-time data that has already arrived at one of the terminals, it does not mean the VDTN nodes are being monitored in real-time.

3.4 Vehicular Network Clustering Algorithms

Clustering algorithms are commonly employed in wireless networks in order to organize the network into clusters, which are groups of adjacent nodes selected by a predefined criterion. Clusters are usually formed by a number of cluster members and a cluster-head (CH), which acts as a local leader to perform the maintenance of said cluster. Clustering algorithms for Vehicular Networks are an important part of the management architecture proposed in Chapter 4. Thus, this section reviews the most relevant clustering techniques designed for Vehicular Networks. The vehicular network clustering problem is similar in many aspects to the mobile ad hoc network clustering problem in some aspects, but there are particularities exclusive to vehicular networks,

such as the possibility of these networks extending themselves over large areas, and the predictable movement of vehicular nodes, that are usually restricted to roads. These particularities can impact in the clusters formation process, being this the reason for this section focusing only on clustering algorithms designed for vehicular networks.

Kayis and Acarman proposed the CF-IVC (Clustering Formation for Inter-Vehicle Communication) for hierarchical inter-vehicle communication [Kayis and Acarman, 2007]. CF-IVC uses the vehicles' speed information to cluster together nodes with low relative mobility among themselves. The clustering formation process relies on information piggybacked in packet headers, turning the usage of specific clustering messages unnecessary. Vehicles belonging to different clusters that are close from each other use different Code Division Multiple Access (CDMA) orthogonal codes, in order to avoid collisions.

Su and Zhang proposed a clustering technique that aims to guarantee the delivery of real-time and nonreal-time data in V2V communication [Su and Zhang, 2007]. The proposed clustering algorithm takes vehicle movement direction as the main metric for determining the cluster members. The cluster-head is then used a) to coordinate the collection and delivery of real-time safety messages within the cluster; and b) to perform channel assignments for cluster-members transmitting/receiving nonreal-time traffic.

The Modified Distributed and Mobility-Adaptive Clustering (DMAC) minimizes cluster reconfiguration and cluster-head changes in VANETs [Wolny, 2008]. The algorithm uses periodic HELLO messages for recognizing neighbors and for disseminating neighbor weights (e.g., connectivity, energy level, etc.) and freshness values. The freshness is a clustering metric proposed by the authors where two nodes use the movement direction to estimate the communication time. This metric avoids re-clustering when the expected communication time is too short. MDMAC is also one of the few vehicular network clustering algorithms able to establish clusters with members distant N-hop one from another.

The Robust Mobility Adaptive Clustering (RMAC) employs relative node mobility inputs (speed, locations and direction of travel) for identifying 1-hop neighbors and to select optimal cluster-heads [Goonewardene et al., 2009]. With RMAC, a clustered node can perform the role of cluster-head, cluster member, or operate in Dual State, which means the node is both a cluster-head for its own cluster and cluster member for one or more other clusters. It is important to mention that the Dual State allows cluster overlapping, which makes RMAC one of the few clustering techniques with this feature. RMAC relies mostly on unicast packets synchronised by the cluster-heads for disseminating neighbour information, in order to support geographic routing. RMAC

also employs a Zone of interest, which is a circular area centred on a node and limited by some radius, in order to limit the neighbor table only to those inside the specified zone.

The Density Based Clustering (DBC) algorithm uses a clustering metric that considers density of connection graph, link quality and traffic conditions [Kuklinski and Wolny, 2009]. It employs classic reactive single path routing protocol in dense parts of the network, and uses multipath routing to increase the reliability of the sparse parts of the network.

Almalag and Weigle propose a vehicular network clustering technique that selects cluster-heads based on the lane where most of the traffic will flow [Almalag and Weigle, 2010]. This technique is meant to be used in urban scenarios with intersections, with the vehicles being able to determine its exact lane on the road. The authors also assume that the vehicles have an in-depth digital street map that includes lane information in order to process the data needed for the technique to work properly.

Wang and Lin propose three Passive Clustering (PC) based techniques for Vehicular Networks, called VPCs [Wang and Lin, 2010]. Each VPC considers one of the following routing metrics: vehicle density, link quality or link sustainability. VPCs operate at the logical link control sub-layer, and their main objective is to let cluster-heads and gateway candidates self-determine whether they require changing their current state or not when receiving route request packets.

Souza et al. introduces a beacon-based clustering algorithm that uses an Aggregate Local Mobility (ALM) metric, which aims to measure the cohesion of a group of nodes, for deciding whether to reorganize a cluster or not, in order to extend the lifetime of said cluster [Souza et al., 2010]. Additionally, a contention-based scheme is used for avoiding frequent re-organizations caused by a brief encounter of two cluster-heads.

The vehicular clustering based on the weighted clustering algorithm (VWCA) takes into consideration the number of neighbors based on dynamic transmission range, the direction of vehicles, the entropy, and the distrust value parameters (a value that represents the distrust of a vehicle towards one of its neighbours regarding messages forwarding) [Daeinabi et al., 2011]. Through this composition of metrics, VWCA improves the scalability, connectivity, and security performances of VANETs.

The Adaptable Mobility-Aware Clustering Algorithm based on Destination positions (AMACAD) takes into account a series of parameters related to the mobility pattern of the vehicles, in order to prolong cluster lifetime and reduce global overhead [Morales et al., 2011]. The parameters considered in order to arrange the clusters are current location, speed, relative destination and final destination of vehicles.

Zhang *et al.* introduced a multi-hop clustering scheme based on the relative

mobility between vehicles distant many hops one from another [Zhang et al., 2011]. Each vehicle periodically broadcasts beacon messages, and they use the delay between two successive beacon messages to estimate the relative mobility between them. The node to be chosen as the cluster-head is the one with the lowest aggregated mobility in its N-hop neighborhood, in order to achieve stable vehicle clusters.

Maslekar *et al.* proposed a direction based clustering algorithm to implement adaptive traffic lights. The algorithm features a cluster-head switching mechanism aimed to overcome the influence of overtaking within the clusters [Maslekar et al., 2011]. In this work, it is assumed that every vehicle in the network knows its own position using a GPS, and that it is equipped with digital maps for determining the direction of travel. The algorithm also acts as a platform to estimate the density of vehicles approaching intersection, enabling the realization of various efficiency and resource discovery applications in VANETs.

Maglaras and Katsaros proposed a clustering technique for vehicular networks based on force-directed algorithms [Maglaras and Katsaros, 2012]. These force-directed algorithms are inspired in physics to assign forces for the edges and the nodes in a network, in order to provide a representation of the network as a physical system. In this representation, the forces applied between vehicles reflect the ratio of divergence or convergence among them. The algorithm uses the location provided by GPS receivers, speed vector, node state and timestamp to calculate the forces.

Another approach for clustering in VANETs is proposed by Rawshedh and Mahmud [Rawashdeh and Mahmud, 2012]. This approach uses the speed difference, the location and the direction of the vehicles as the metric for the cluster formation process. Simulation results demonstrated that this technique can group fast moving vehicles on fast lanes in one cluster, and slow moving vehicles in another cluster. The admission of new members into existing clusters is conditioned to the candidate member having a relative speed within a pre-defined threshold.

Wahab et al. proposed an extension for the QoS-OLSR protocol, called VANET QoS-OLSR [Wahab et al., 2013]. This proposed clustering protocol considers a tradeoff between Quality of Service requirements and mobility by adding the velocity and the residual distance metrics to the QoS function. The protocol is composed of three components: QoS-based clustering using Ant Colony Optimization for the selection of Multipoint Relays (MPRs), MPR recovery algorithm for selecting alternative MPRs in case of link failure, and cheating prevention mechanism that relies on an encryption algorithm to avoid cheating during the MPRs selection.

The Affinity PROpagation for VEhicular networks (APROVE) employs a similarity function that uses the vehicles positions to calculate a combination of the negative

Name	Clustering inputs	Cluster joining criteria	Reference
CF-IVC	Speed	Relative speed within cluster thresholds	[Kayis and Acarman, 2007]
Su and Zhang, 2007	Signal strength	Vehicles moving in the same direction	[Su and Zhang, 2007]
MDMAC	Weight and freshness	Relative speed within thresholds	[Wolny, 2008]
RMAC	Speed, location and direction	Neighbours with highest precedence	[Goonewardene, 2009]
DBC	Distance between nodes, relative speed and link quality	Link quality above threshold	[Kuklinski and Wolny, 2009]
Almalag and Weigle, 2010	Traffic flow, speed, direction, location and lane	Vehicles moving in the same direction	[Almalag and Weigle, 2010]
Souza et al., 2010	Aggregate local mobility	Positive relative mobility metric	[Souza et al., 2010]
VPCs	Vehicle density, link quality or link sustainability	Suitability according to the metrics	[Wang and Lin, 2010]
VWCA	Distrust value, number of neighbors and direction	High entropy and low distrust	[Daeinabi et al., 2011]
AMACAD	Location, speed and destination	Density, speed and bandwidth within thresholds	[Morales et al., 2011]
Zhang et al. 2011	N-hop relative mobility	Least hops-distance and lowest relative mobility	[Zhang et al., 2011]
Maslekar et al. 2011	Position and direction determined in a map	Vehicles moving in the same direction	[Maslekar et al., 2011]
Maglaras and Katsaros, 2012	Location, speed vector	CH has bigger relative force	[Maglaras and Katsaros, 2012]
Rawshedh and Mahmud, 2012	Speed difference, location and direction	Relative speed within threshold	[Rawshedh and Mahmud, 2012]
VANET QoS-OLSR	Bandwidth, connectivity, residual distance and velocity	Suitability of QoS metrics	[Wahab et al., 2013]
APROVE	Current and future node position	Vehicles moving in the same direction	[Hassanabadi et al., 2014]
Fathian and Jafarian-Moghaddam, 2015	Speed, direction and location	Node within DEA frontier or distance to center of cluster	[Fathian and Jafarian-Moghaddam, 2015]

Table 3.2. Main clustering solutions for vehicular networks

Euclidean distance between node positions now and in the future, preventing vehicles moving in opposite directions to be clustered together [Hassanabadi et al., 2014]. The algorithm also minimizes relative mobility and distance between cluster-head and cluster members to produce clusters with long average member duration, long average cluster-head duration, and low average rate of cluster-head change.

Fathian and Jafarian-Moghaddam [Fathian and Jafarian-Moghaddam, 2015] proposed two new clustering algorithms: the Multi-objective Data Envelopment Analysis-based clustering algorithm, and the Ant Colony System-based clustering algorithm (ASVANET). The goals behind the proposed algorithms were to minimize vehicle transitions between clusters and to run in a limited timeframe. The proposed algorithms take into account vehicle speed, direction and location as the main clustering metrics.

Table 3.2 presents the clustering algorithms reviewed in this subsection, highlighting some of their main characteristics, namely: the inputs used by the algorithms during the cluster formation process, the criteria used by the algorithms specifying whether to allow a node to join a cluster or not, and the references for the works that introduce the listed clustering algorithms.

3.5 Chapter Conclusions

This chapter presented a literature review of Delay-Tolerant Network Management, Vehicular Network Management and Vehicular Network Clustering Algorithms. These topics were chosen due to their proximity to the theme of this thesis.

Despite some proposals for solving the DTN management problems being available in the literature, there is still plenty of space for more research in this area. First, the number of papers regarding DTN management is still limited. Moreover, the proposed DTN management solutions usually take into consideration specific DTN scenarios (e.g., space networks, interconnection of remote villages, vehicular networks, etc.), and some of these scenarios were little explored by the investigations currently available, such as vehicular networks.

Management solutions for Vehicular Networks are scarce in comparison to DTN in general. It is important to notice that there are challenges imposed by Vehicular Networks that are inherent to this specific type of network, specially the fast and predictable mobility of the nodes and the strict communication latency requirements that some applications impose. The impact of these challenges in management systems must be further studied, so more effective and reliable solutions for vehicular network management can be found.

This chapter presented a number of clustering techniques for vehicular networks. Each technique is based on a set of assumptions and desired characteristics, making each of them suitable for a certain number of scenarios. However there are still scenarios in the vehicular networks universe where none of the presented clustering techniques is fit for providing the desired results (Chapter 4 includes an example of such scenario). Hence, the design of new clustering algorithms for vehicular networks is still a valid research problem.

Chapter 4

The HE-MAN Architecture

This chapter presents a new architecture for the management of Vehicular Delay-Tolerant Networks. For a proper comprehension of the remainder of this chapter, it is important that some definitions are made clear.

Definition 4.1. A **network management architecture** is a collection of different components that interact with each other in order to provide the means necessary for a network to be managed. The most common components found in network management architectures are managers and agents.

Definition 4.2. A **manager** is a component of a network management architecture that manages other nodes in the network. A manager can either perform its tasks in an autonomous fashion, or act as an intermediate entity between managed nodes and network operators.

Definition 4.3. An **agent** is a component of a network management architecture that interacts with the pieces of hardware and software of network nodes in order to make said nodes manageable.

The architecture presented in this chapter is composed of a set of algorithms and definitions that a) organizes the topology of managers and agents; b) monitors devices and services in a way that is suited for VDTNs; and c) implements configuration techniques that deal with control commands being issued to vehicular nodes not only in situations where long delays are involved, but also in situations where the configuration must occur with strict time restraints (such as safety applications). The next section presents an overview of this architecture.

4.1 HiErarchical MANagement (HE-MAN) Architecture for Vehicular Delay-Tolerant Networks

VDTN management presents unique requirements, such as the provision of near real-time management capabilities in order to allow the management of delay-sensitive applications, like road safety systems. Therefore, the Hierarchical MANagement (HE-MAN) Architecture for Vehicular Delay-Tolerant Networks is proposed with the objective of fulfilling the VDTN management objective, as stated in Definition 2.4.5 presented in Chapter 2. The HE-MAN architecture can be defined as the following:

Definition 4.4. HE-MAN is an architecture designed for managing networks that experience frequent network partitioning and that need to run delay-sensitive systems (e.g., VDTNs) through the usage of a hierarchical organization of its components. The established hierarchy aims to detect connected group of nodes to establish local managers and to take advantage of local communication opportunities for achieving near real-time management. The architecture is composed of a set of managers, one Top-Level Manager for the entire VDTN and a set of Mid-Level Managers, one per group of connected nodes. The remainder components found in the HE-MAN architecture are agents, and protocols for organizing message exchanges and the logical organization of the VDTN's topology.

HE-MAN takes advantage of local communication opportunities (which is faster and more reliable), while avoiding remote communication whenever possible (slower and less reliable). This strategy allows the management of applications and devices with little tolerance to communication delay. On the other hand, applications that are not sensitive to delays can be managed via remote communication. In VDTN, the division of the management tasks in local and remote tasks can be accomplished with a hierarchical architecture, where Mid-Level Managers (MLMs) manage locally connected groups. In the scope of this work, a locally connected group exists when at least two nodes are within transmission range of each other. Vehicles that are not within the transmission range of any other node are considered isolated.

MLMs can perform tasks delegated by the VDTN's Top-Level Manager (TLM), which is a unique central entity that manages an entire vehicular network (e.g., the operational center of an enterprise that runs a fleet of vehicles). The TLM can be implemented by a single stationary node in the network, or it can be distributed implemented by permanently linked infrastructures, such as RSUs. All MLMs established

along the VDTN are directly subordinated to the TLM, which results in a 2-levels hierarchy of managers. Finally, HE-MAN managers adopt the publisher-subscriber model for monitoring tasks, proposed in the Diagnostic Interplanetary Network Gateway (DING) Protocol Papalambrou et al. [2011]. Figure 4.1 illustrates the organization of a hierarchical management architecture in a VDTN.

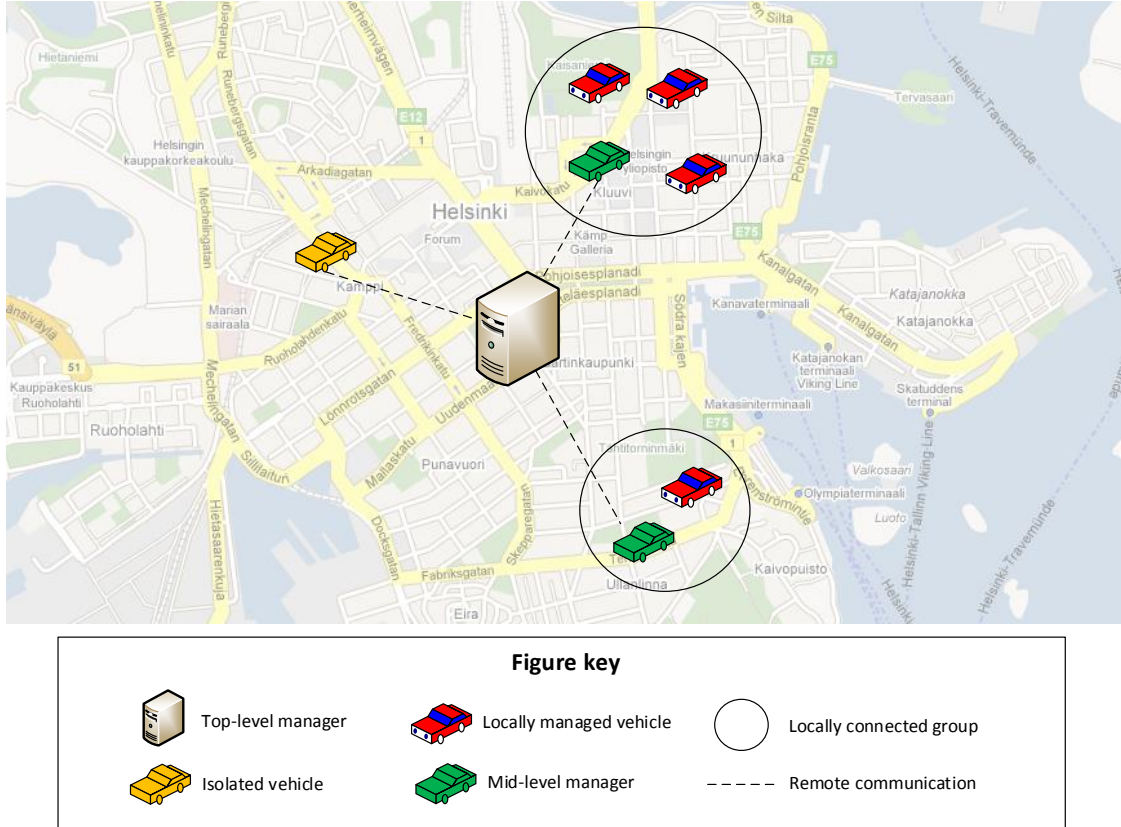


Figure 4.1. Hierarchical monitoring in a VDTN

The main software modules that are used by the nodes in the HE-MAN architecture are described below.

Top-Level Manager Module: this component keeps record of the status of the entire network, in order to support the decision-making process of network administrators. The TLM can either issue configuration commands directly to specific nodes, or it can delegate a configuration task for a MLM to redistribute the configuration among its cluster. A database is used to store historical data regarding the received monitoring data and the performed management tasks. The main TLM module attributions are the following:

- To store monitoring data sent by the network nodes (MLMs and agents);

- To issue new configuration commands from managers to any network node;
- To issue new management functionalities for MLMs;

Mid-Level Manager Module: the MLM module is responsible for implementing the management functions that this type of node must perform within the clusters. Therefore, this module must be able to process monitoring data sent by other members of the cluster, and to react to detected faults in a quick and reliable manner. Moreover, the MLM module needs to issue periodic messages to the TLM with monitoring data regarding its own cluster, and to redistribute configuration commands to other cluster's members whenever requested by the TLM.

The main functionalities of the MLM module are the following:

- To receive monitoring data from subscribed nodes;
- To treat detected faults, following pre-installed procedures;
- To redistribute configuration commands among its agents whenever ordered by the TLM.

Agent Module: this module employs the publisher-subscriber scheme for monitoring tasks. An implementation of a regular SNMP agent is used to access management objects within the VDTN nodes, in order to achieve a certain level of compatibility with management solutions that rely on SNMP. A database is also used in the vehicular nodes, in order to locally store management-related data that can be retrieved by external managers in case of any kind of data loss (e.g., bundles not delivered, data loss in the TLM's database, etc.).

The functionalities of the agent module are the following:

- To provide monitoring data to MLMs and TLM;
- To instantiate newly received configurations.

Both Mid-Level Manager Module and Agent Module are installed in vehicular nodes, enabling these nodes to perform any of these roles whenever needed. This provides flexibility to the network, making it easy to transform a common agent node into a MLM, or vice-versa. Figure 4.2 illustrates the organization of HE-MAN's software modules: the TLM module on the left-hand side, and the Agent and MLM modules on the right-hand side. These last two modules are grouped together in the figure due to the similarity between them, being differentiated only by the presence of the HE-MAN MLM or HE-MAN Agent as the core component.

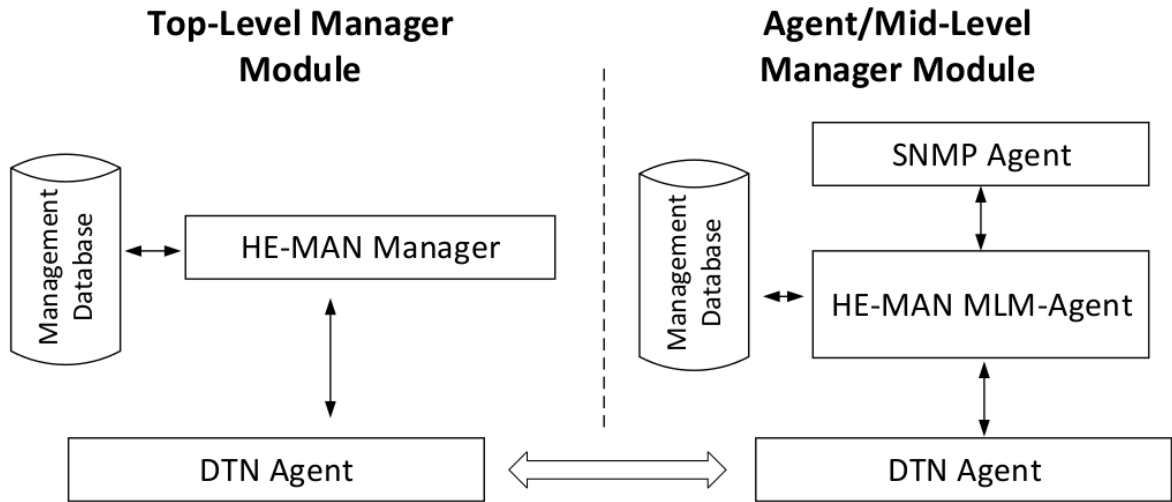


Figure 4.2. Main components of the HE-MAN architecture

The configuration of a hierarchical management system depends on the formation of stable groups of nodes. Every node that is part of the management network must have “connection awareness” to recognize its vicinity and determine whether the establishment of a local group is possible or not at a given moment. The group creation and MLM election process uses a clustering algorithm, which is described in the next section.

4.2 Clustering Algorithm

HE-MAN clustering algorithm maximizes the amount of time that a node is connected to a MLM, so this node can be locally monitored/configured as much as possible. Unfortunately, clustering algorithms designed for vehicular networks sometimes implement filters to prevent nodes from joining a group, in order to increase cluster stability, thus reducing the connection time [Vodopivec et al., 2012]. For instance, some algorithms do not allow vehicles travelling in opposing lanes to be members of a same cluster, due to the short amount of time they would be in contact with each other.

Regarding the clustering aspect of the HE-MAN architecture, the main definitions adopted in this thesis are the following.

Definition 4.5. A **cluster** is defined as a subset of two or more nodes that are directly reachable (one-hop communication) by one special element of the subset that acts as the leader of the cluster (also referred as cluster-head). In this thesis clusters can also be referred as local group or connected group.

Definition 4.6. The **cluster-head** is an unique node within a cluster that acts as the leader of the cluster. Hence, every other member of a cluster establishes some kind of communication with the cluster-head, resulting in the cluster-head ultimately defining the range of the cluster with its own communication range. In the scope of this thesis, the cluster-head is always the MLM of the cluster.

Definition 4.7. The **role** of a cluster node can be defined as a specific way this node interacts with the other members of the cluster. There are three possible roles that cluster nodes can perform, as further explained later in this section: MLM, agent or isolated.

Definition 4.8. A **beacon message** is defined as a type of message periodically issued by a VDTN node with the objective of sending to nearby nodes updates on some property of interest. For instance, a node can issue beacon messages informing to other nodes what role it is currently performing.

The next subsection presents a general description of the algorithm, explaining it with a high level of abstraction. Next, the various procedures that compose the clustering algorithm are presented and detailed.

4.2.1 General description of the algorithm

The HE-MAN algorithm relies on information of the connectivity history of nodes to create clusters with high stability. However, it does so without preventing nodes from joining certain groups, as other algorithms usually do. Furthermore, our clustering algorithm attempts to maximize the amount of time that a node spends connected to one of the VDTN clusters. To the best of our knowledge, an algorithm with such characteristics is not available in the literature.

In HE-MAN clustering algorithm, the detection of neighboring nodes is based on periodic beacon messages. Only neighbors within a 1-hop distance are detected, which means that each formed cluster will also encompass nodes that are 1-hop away from the cluster-head. This avoids the need for retransmission of beacon messages, which would increase the complexity and decrease the overall performance of the algorithm. Moreover, the HE-MAN clustering algorithm assumes that all nodes in the network are homogeneous.

A node can assume one of three roles: **isolated**, **agent** or **MLM**:

- **Isolated:** nodes that have no active contact with other nodes. Its objective is to identify opportunities of joining existing clusters or creating new clusters;

- **MLM**: the cluster-heads. Its main objectives are to maintain updated data about the current state of the cluster, and to initiate cluster-head transference operations whenever needed;
- **Agent**: it is a member of a cluster, but not its cluster-head. Nodes performing this role are constantly feeding the MLM (cluster-head) with monitoring data.

Beacon messages transport information concerning basic properties of a node (e.g., network address, vehicle identification, vehicle speed, etc.), as well as its **Connection Score** (connScore). The connScore is a metric that represents the history of connection opportunities that a node had, and it is used to determine if such node is the fittest for the MLM role in a cluster. This metric gives priority to nodes that are consistently establishing contacts with other nodes. Each node calculates its own connScore by sampling the number of active contacts at a given time, and then calculating the samples' Exponentially Weighted Moving Average (EWMA) [Roberts, 1959].

The EWMA is a statistic that processes historical data through a calculation that provides means for older values to be progressively “forgotten”. The calculated EWMA can have a long memory or a short memory, depending on the definition of a constant λ . The EWMA can be calculated using the equation below:

$$\hat{y}_t = \hat{y}_{t-1}\lambda + (1 - \lambda)y_t \quad (1)$$

The y_t variable is an observed value at time t , \hat{y}_{t-1} is the old calculated EWMA, and λ is a constant ($0 < \lambda < 1$) that determines the depth of memory of the EWMA [Hunter, 1986]. By employing the EWMA statistic in the proposed clustering algorithm, it is possible to control how much impact older samples of a node's connScore will have over new samples, which provides a final connScore value that reflects better the current connectivity state of a node. Algorithm 1 shows how the connScore metric is calculated by vehicular nodes.

Algorithm 1 calculateConnScore() function

```

1: function CALCULATECONNSCORE()
2:   previousConnScore  $\leftarrow$  getConnScore();
3:   activeConnNumber  $\leftarrow$  getActiveConnectionsNumber();
4:   return ( $\lambda * \textit{previousConnScore}$ ) + (( $1 - \lambda$ ) * activeConnNumber)
5: end function

```

4.2.2 Algorithms' details

The procedure *update*, described in Algorithm 2, is periodically executed (e.g., every second) by each node of the VDTN. The first thing this procedure does is to update

the value of the node's `connScore` (line 2). If the node is a member of a group (either as agent or MLM), it checks if it can still be considered connected to the group. In order to do so, agents check if the last received beacon from the MLM node is within an acceptable time window, and MLMs check if beacons from the managed agents arrived in an acceptable time window (line 5). These time windows are configurable parameters. If this check fails, the node changes its status to **isolated**. At the end of the procedure the node broadcasts to its neighborhood the beacon with its current up-to-date role and other relevant information (line 8).

Algorithm 2 `update()` procedure

```

1: procedure UPDATESTATUS()
2:   connScore  $\leftarrow$  calculateConnScore();
3:   status  $\leftarrow$  getStatus();
4:   if status=mlm or status=agent then
5:     validGroup  $\leftarrow$  checkIfValidGroup();
6:     if validGroup=false then
7:       setStatus(isolated);
8:   sendBeacon();
9: end procedure

```

Each type of node reacts differently to the reception of a beacon. Because of this, different procedures for handling received beacons are specified, one for each type of node (isolated, MLM and agent).

Algorithm 3 details how isolated nodes react when receiving a beacon. First of all, the recipient reads the information contained in the beacon to update the data about neighboring nodes, through the *updateNeighborhood()* call (line 2). Then, it checks if the received beacon was originated from a MLM or another isolated node (lines 3 and 7). If the received beacon was originated from a MLM, the node joins the MLM's cluster, changing its status to agent (line 5) and requesting that the MLM subscribes to it in order to receive management data (line 4). When the MLM receives the *subscriptionRequestTo* from the isolated node, it issues a *subscribe* message to indicate the inclusion of the node in the cluster, which is then replied with a *subscription ack* message. If the received beacon was originated from another isolated node, the recipient node creates a new cluster and runs the election function to determine which node will perform the MLM role (line 8).

It is important to notice that both nodes involved in the communication will run the election mechanism, since both of them sent an isolated beacon message upon the detection of the contact between them. In the election, each node compares its

connScore to the connScore of the others: whichever has the highest connScore assumes the role of MLM. In case of a tie, the node with the highest address wins.

Finally, the elected MLM subscribes to the other node of the cluster (line 10), in order to start monitoring them. The node that lost the election remains in the isolated role, waiting for the elected MLM to request the subscription. Cluster nodes, after receiving the subscription request, change their status to **agent** and reply the MLM with a *subscription ack* message. Figure 4.3 shows the behavior shown by an isolated node according to the different types of beacon messages it can receive.

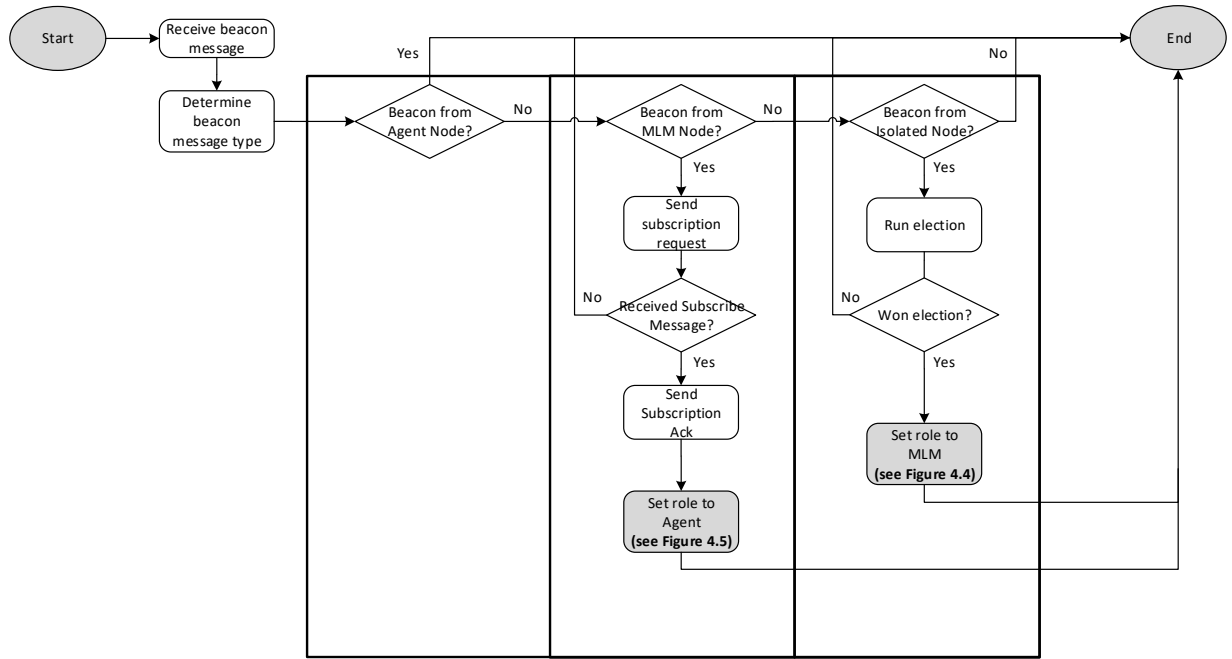


Figure 4.3. Behavior of isolated nodes according to the received type of beacon message

In HE-MAN's clustering algorithm, there are two possibilities for the MLM role to be transferred from one node to another:

- **Group merge:** a group merge occurs when MLMs of different clusters get within the range of each other for a period of time longer than a pre-configured threshold. In such case, the MLM with the highest connScore becomes the unique MLM of all the clusters involved in the merging operation, and the cluster established as the result of the merging is bounded by the transmission radius of the chosen MLM;
- **MLM transfer:** it occurs within a cluster, when an agent is connected to the cluster for a period of time longer than a pre-configure threshold, and it has a

Algorithm 3 isolatedBeaconHandler(beacon) procedure

```

1: procedure ISOLATEDBEACONHANDLER(BEACON)
2:   updateNeighborhood();
3:   if isMLMBeacon(beacon)=true then
4:     subscriptionRequestTo(getFrom(beacon));
5:     setStatus(agent);
6:   else
7:     if isIsolatedBeacon(beacon)=true then
8:       winner  $\leftarrow$  runElection(beacon);
9:       if winner=me then
10:        subscribeTo(getFrom(beacon));
11:        setStatus(MLM);
12: end procedure

```

connScore higher than the current MLM of the cluster. This operation transfers the MLM role to the agent in question, with the previous MLM changing back to the agent role.

Algorithm 4 dictates how MLMs react to beacons issued by other MLMs, which happens whenever the involved MLMs attempt to merge their respective clusters into a single larger cluster. Upon the reception of a beacon sent by another MLM, the recipient calls the *contTime* function to verify if the two MLMs are in contact to each other for an amount of time higher than a pre-configured threshold (line 4). The threshold is another configurable parameter in this clustering algorithm, and can be defined according to the needs of each VDTN using the proposed clustering technique. If the minimum threshold is crossed, then the procedure checks whether the connScore of the MLM that issued the beacon is higher than the connScore of the MLM that received the beacon (line 5). In case the last condition is true, then the MLM with the smaller connScore issues a *requestGroupMergeTo* message, which starts the groups merging procedure (line 6). The *requestGroupMergeTo* message contains a list of all the agents being managed by the MLM with the smaller connScore. When the other MLM receives this message, it issues subscription requests to the nodes of the received agents list, and sends a *Group Merge Ack* message back to the MLM with the smaller connScore, in order to release it from the MLM role. On the other hand, if the received beacon was issued by an agent, the MLM verifies if the connection to said agent is above a pre-configured threshold (line 8). If this verification is evaluated as true, the MLM checks if the connScore of the agent that issued the beacon is higher than its own connScore (line 9). If the agent has a higher connScore, then the MLM issues a *requestMLMTransferTo* message, which starts the transference of the MLM role to

the agent that issued the beacon (line 10). The different behaviors shown by MLMs in response to different types of received beacon messages are illustrated in Figure 4.4.

Algorithm 4 MLMBeaconHandler(beacon) procedure

```

1: procedure MLMBeaconHandler(BEACON)
2:   updateNeighborhood();
3:   if isMLMBeacon(beacon)=true then
4:     if contTime(getFrom(beacon))>=THRSHLD then
5:       if getConnScore(beacon)>connScore then
6:         requestGroupMergeTo(getFrom(beacon));
7:   if isAgentBeacon(beacon)=true then
8:     if contTime(getFrom(beacon))>=THRSHLD then
9:       if getConnScore(beacon)>connScore then
10:        requestMLMTransferTo(getFrom(beacon));
11: end procedure
  
```

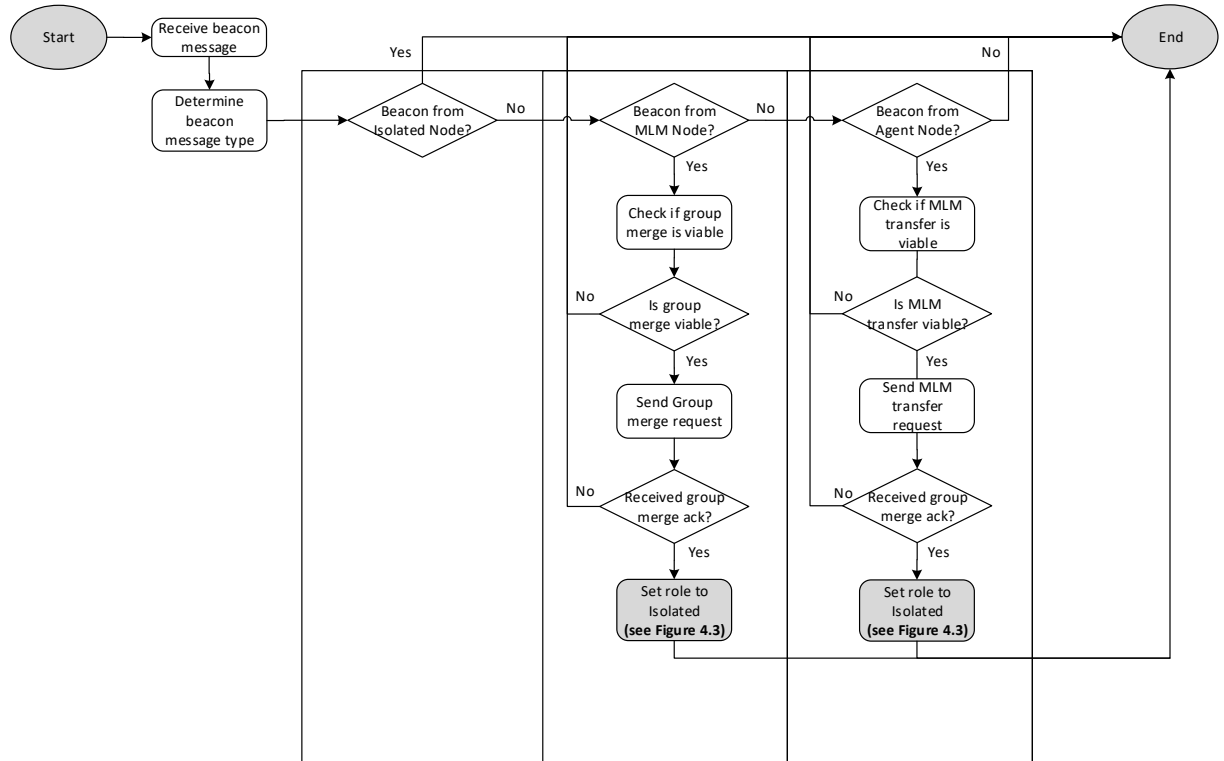


Figure 4.4. Behavior of MLMs according to the received type of beacon message

Finally, agent nodes do not have any special reaction to other beacons other than updating their data about the vicinity with the *updateNeighborhood* procedure. Agents respond to *subscription requests* and *subscription transfers* issued by the MLM

that are managing them, in order to realize subscriptions and, ultimately, to become manageable by the cluster’s MLM. Figure 4.5 shows how agent nodes react to the different types of beacon messages.

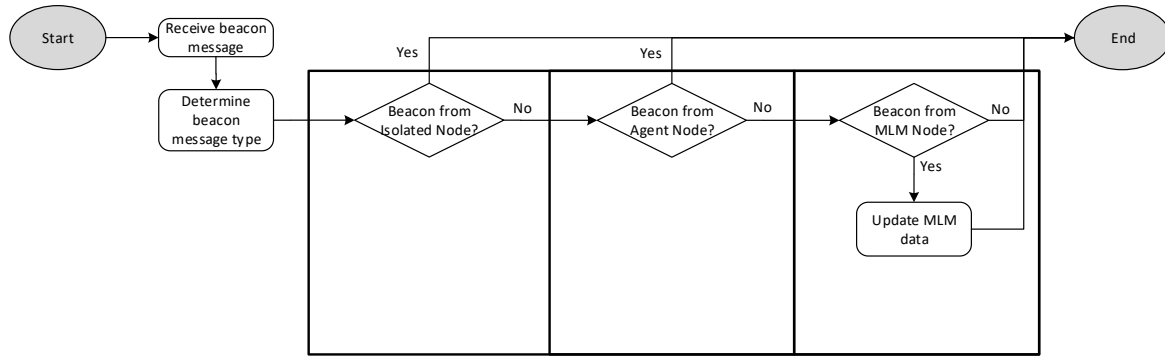


Figure 4.5. Behavior of agent nodes according to the received type of beacon message

4.3 Monitoring Support

HE-MAN employs the publisher-subscriber model for monitoring. The reason is to replace the polling-based monitoring used by SNMP with a less chatty approach, since in this model a manager subscribed to an agent once, and after that the manager is able to receive an arbitrary number of monitoring “responses”, periodically. It is important to notice that this scheme is different from traditional SNMP polling in the sense that each monitoring message issued by an agent requires a previous request message from the manager, which is difficult to be achieved in an environment with constant links disconnections due to the high mobility of nodes, such as in VDTNs.

4.3.1 Publisher/Subscriber Model

Traditional networks, such as the ones managed purely with SNMP, rely on periodic polling to monitor target devices. The “chatty” nature of polling-based monitoring does not fit VDTN management well, since this form of communication requires at least one round trip for each request, with both request and response messages suffering from long delays or even not being delivered at all. A feasible solution to this problem consists of replacing the polling mechanism with the publisher/subscriber model.

A publisher/subscriber based on DING [Papalambrou et al., 2011] is used in the HE-MAN architecture for monitoring tasks. In this model, presented in Figure 4.6,

the subscription message issued by the MLM is composed of two parts: a **schema** and a **schedule**. The **schema** is a static data model definition that describes what the manager wants to receive. It is a list of Object Identifiers (OIDs) that is verified by the publisher upon reception. After the publisher verifies what OIDs from the list it can provide to the subscriber, it will start to send the requested information in accordance to what is defined in the **schedule**. In this case, a **schedule** is a list of time intervals where agents are supposed to send management information back to managers. This **schedule** can have a single time interval that is applied to all OIDs requested in the schema, or one time interval for each OID in the schema. The time interval is specified in seconds. The publisher will periodically send scheduled frames back to the subscriber, containing a list of values regarding the management objects previously provided by the subscriber during the subscription process.

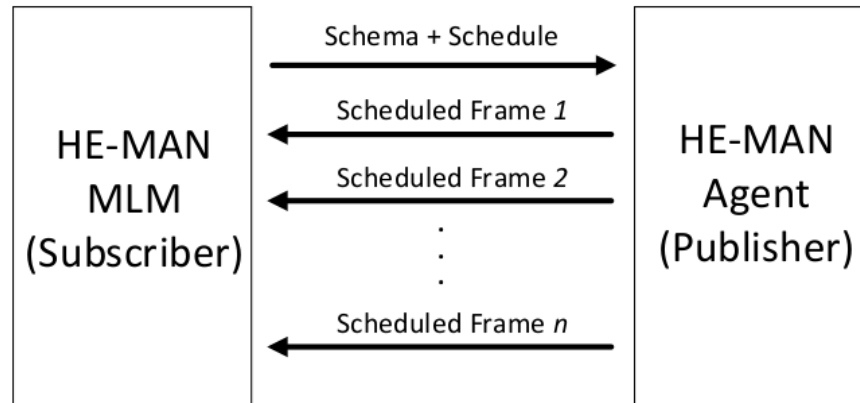


Figure 4.6. Messages exchange in the publisher-subscriber monitoring

4.3.2 Local and Remote Monitoring

In HE-MAN, the monitoring task is divided into local and remote monitoring. This division reflects the two different type of communication allowed by the hierarchical organization of the VDTN.

Local monitoring occurs whenever a node is a member of a cluster and it is being monitored by a MLM. This type of monitoring is suited for devices and applications that need to be monitored in near real-time, since the delays involved in local communication are a small fraction of the ones experienced in VDTN remote communication. The small delays experienced in local communication allow monitored objects to be verified more frequently, which is useful for the monitoring of critical network services and hardware components by a network manager.

On the other hand, **remote monitoring** occurs whenever the TLM is the destination of the monitoring data. In this type of monitoring the transmitted data is subject to long delivery delays and a smaller delivery success ratio. However, the main advantage of remote monitoring is the fact that management data originated from different parts of the VDTN is centralized in a single entity of the management system, which is the TLM. The centralization of “global” monitoring data in the TLM allows a better comprehension of the network as a whole, which can support better management decisions by an automated system or a human operator.

A MLM can also periodically aggregate monitoring data received in a certain time window in order to send to the VDTN’s TLM a single bundle with data regarding its whole cluster. The TLM can activate or deactivate the aggregation functionality by issuing configuration commands to the MLMs (HE-MAN’s support to configuration commands is explained in the next subsection). The main advantage provided by this feature is the reduced number of bundles in the VDTN at a given moment with monitoring data. One possible aggregation that could be implemented in the nodes is the calculation of averages and standard deviation of a given OID that deals with numeric values. In this case, the MLM would gather the values of said OID issued by the cluster’s agents, calculate the average and standard deviation regarding the received set of values and then sending to the TLM only these two calculated results, instead of a list of all values issued by each cluster member.

4.4 Configuration Support

Just as with the monitoring tasks, VDTNs’ connectivity limitations offer challenges to configuration tasks. Frequent link disconnections usually lead to configuration commands arriving too late at their destination, or not arriving at all. Because of this, management systems designed for VDTNs must provide means for the network to be able to be configured without relying exclusively on configuration commands being issued through the Bundle Protocol Layer.

The HE-MAN architecture divides configuration-related operations into two categories: local configuration and remote configuration. Local configuration is preferred whenever the decision making process can be carried out exclusively using local information gathered by the group’s MLM. On the other hand, when the configuration depends on information regarding the VDTN as a whole, then remote configuration should be employed. The subsections below provide further details on how local and remote configuration work in the HE-MAN architecture.

4.4.1 Local Configuration

In HE-MAN, a configuration operation is local when a MLM issues configuration commands to one of the cluster members managed by it. **Local configuration** is suited for scenarios where delay-sensitive devices and/or services must be configured, requiring new configurations to be transmitted in near real-time. The significant short delays and higher delivery success ratio of local communication allows the MLM to detect issues almost instantly, similarly to what happens in regular connected networks. Because of this, it is possible to implement complex interactive configuration processes involving successive monitoring operations followed by configuration commands being issued. This kind of configuration cannot be achieved through the regular delay-tolerant communication used in VDTNs. A possible application for this type of configuration process is presented in Subsection 4.5.1.

4.4.2 Remote Configuration

Remote configuration occurs when the VDTN's TLM is the entity issuing configuration commands. Similarly to the case of remote monitoring, remote configuration is performed in a central point of the network where monitoring data concerning the entire network is stored, which usually leads to more well-founded management decisions. However, HE-MAN is able to deal with a configuration command issued by the TLM arriving at its destination too late to be still valid, due to the long transmission delays. Because of this, in HE-MAN a TLM is able to create configuration scripts containing a set of configuration commands that are executed only if certain conditions are met. These conditions are meant to assess if the issue being approached is still valid when the configuration message reaches its final destination. The structure of the remote configuration message can be described as follows:

- **Conditional expression:** configuration messages carry a conditional expression, primarily to check if the state of the targeted node is still the same. Each expression is composed of a reference to an Object Identifier (OID), a relational operator ($<$, $>$, $==$, $<=$, $>=$, or $!=$) and a constant. Relation expressions can be connected by **AND** and **OR** operators, in order to enable the evaluation of multiple conditions in the destination node;
- **Cluster configuration flag:** when the TLM issues a configuration message to a known MLM, a special flag can be set to inform the MLM that the configuration must be relayed to all cluster members. The flag is ignored in case the

targeted node is not the MLM, to prevent the situation where one cluster receives a configuration meant to be applied to another cluster;

- **Configuration commands list:** the configuration commands to be executed when the conditions within the configuration message are met. If the cluster configuration flag is active, the MLM will locally retransmit the configuration commands to all nodes in its cluster.

It is important to notice that the configuration scripts defined in this thesis can also be seen as a management policy, which is defined by the Policy-Based Network Management (PBNM) approach [Strassner, 2003], since they represent a set of actions to be taken if a certain condition is met. However, the actions performed through a configuration script are restricted to the commands listed in the script, without other actions being automatically devised, as one would find in some implementations of management policies.

Remote configuration scripts are disseminated in the VDTN via a DTN routing algorithm. Nodes will stop forwarding the remote configuration script in one of these two situations: a) the message arrived at its destination, or b) the message reached a MLM that is currently managing the final destination node. Whenever situation *b* occurs, the MLM just delivers the configuration script to the final destination using local communication.

In order to illustrate how the configuration scripts described above could be applied in a vehicular network environment, consider the case where the VDTN vehicular nodes run a set of applications that trigger message exchanges whenever the vehicles get in communication range of one another. Now imagine that a large number of these vehicles got clustered together, and due to the high amount of exchanged messages the nodes' buffers run out of space. This problem is reported by the nodes to the cluster MLM, which afterwards notifies the problem to the TLM through a remote report message. A network administrator decides that some of the applications running in the vehicular nodes should be closed in an attempt to solve the buffer space problem, so the TLM issues the following message to the MLM of the group where the problem was originated.

Conditional expression	<code>vdtn.mlm.average_cluster_buffer_usage (1.3.6.1.3.40.3.5) > 0.7</code>
Cluster configuration flag	<code>true</code>
Configuration commands list	<code>vdtn.applications.social_chat.active (1.3.6.1.3.40.5.82.1) = false</code>
	<code>vdtn.applications.advertisements.active (1.3.6.1.3.40.5.91.1) = false</code>

The condition expression of the message above instructs the MLM to first verify if the managed object `vdtn.mlm.average_cluster_buffer_usage`, with OID 1.3.6.1.3.40.3.5, is still above 70% of usage, in order to avoid an unnecessary termination of applications. If the condition is true, then the node checks the cluster configuration flag, which is marked as true, meaning the configuration commands should be relayed to all cluster members. Finally, the configuration commands list deactivates two applications, represented by the managed objects `vdtn.applications.social_chat` (OID 1.3.6.1.3.40.5.82) and `vdtn.applications.advertisements` (OID 1.3.6.1.3.40.5.91).

4.5 Case Studies

This section presents three case studies for network management in order to illustrate the applications of the techniques presented in this chapter. Each case study includes a description of the applications considered in the scenario, the issues that the management operations face in such scenario and how the HE-MAN architecture tries to solve those issues. The case studies presented in this chapter are meant to be seen as a qualitative analysis of HE-MAN's solutions for network management. A quantitative analysis can be found in Chapter 5.

4.5.1 Case Study One: Platoon of Autonomous Vehicles

Consider the case of an army that transports equipment and troops using autonomous terrestrial vehicles. Moreover, these vehicles are grouped into platoons that move in a standardized way, where all vehicles of the group move at consensual speed while maintaining a desired space between adjacent vehicles [Horowitz and Varaiya, 2000]. coordination between the vehicles is necessary in order for the platoon to move as a cohesive unit. This coordination is done through a series of adjustments over their travel direction and speed, in order to respect the accorded platoon movement speed and direction, as well as the spacing between vehicles. It is not hard to notice that this coordination needs to be carried out with near real-time communication, otherwise the vehicles in the platoon would break the platoon formation because of laggy coordination commands.

The autonomous vehicles platoon scenario can benefit from the HE-MAN architecture in a few ways. First, the establishment of a coordinator in the platoon would be transparent, since the proposed clustering technique used in HE-MAN elects a MLM (cluster-head) whenever two or more nodes are in communication range of each other. Then, the MLM can assume the coordination of the platoon by subscribing itself for

receiving monitoring data from the other vehicles of the group, using the local monitoring support. By having the required amount of monitoring data in a timely fashion, the MLM will then be able to issue configuration commands to the platoon's vehicles, such as speed and direction changes, so the platoon's formation can be maintained during the entirety of the operation. Finally, the VDTN's TLM can be installed in an operational command center, which will disseminate new configuration scripts (i.e., a set of configuration commands) to specific platoons with reduced overhead (in comparison to a pure flooding strategy).

4.5.2 Case Study Two: Bus Fleet Monitoring

For this second case study, consider a company that owns a bus fleet that transports passengers from one city to another. Each bus of the fleet is equipped with an OBU and a set of sensors placed on vehicles' parts that the company wishes to monitor (e.g., fuel tank, tires, etc.). Let us also assume that some of the cities the buses travel to, as well as some of the travelled roads, do not have access to any sort of mobile connection to the Internet. The company wishes to use a fleet management system that aims to collect data from the vehicles of the fleet. The company wants to monitor the trips carried out by the buses (e.g., number of passengers, number of unscheduled stops, etc.), and the "health" of the vehicles themselves (e.g., fuel consumption, tire pressure, etc.). Because of the nature of the data, the company does not have to receive the monitoring data in real-time - there is room for some delay being tolerated, although it is preferable that the data arrives in the shortest time. The scenario used in this case study is inspired by the land transports monitoring system in development by the Brazilian National Agency of Land Transports (ANTT) [Agência Nacional de Transportes Terrestres, 2014].

The main benefit the HE-MAN architecture can offer to this scenario is the remote monitoring feature, which takes advantage of the hierarchical organization of the VDTN to improve the usage of network resources. Whenever vehicles are clustered together, MLMs subscribe to the other members of the cluster to gather monitoring data regarding performed trips and the performance of the relevant vehicle's parts. This monitoring data is not immediately sent to the TLM, but instead they are stored in the MLMs until the time scheduled to generate remote reports arrives. Then, each MLM calculates statistics from the received monitoring data (e.g., average, standard deviation, etc.), and send these statistics in a single message to the TLM. This reduces the amount of monitoring messages being transmitted through DTN communication, consequently reducing the usage of buffer space in the vehicular nodes.

4.5.3 Case Study Three: Transmission Status Propagation System

In this last case study, let us consider a system that allows drivers to share data regarding the conditions of the transmissions and medium access of portions of the network ahead (e.g., statistics on message collisions, transmission rate being used by vehicular nodes, etc.), without relying on any sort of mobile communication technology (e.g., 3G/4G Internet access) other than the VDTN itself. This type of system is particularly useful in areas where there is no coverage of mobile Internet, or this coverage is not reliable enough, such as in deserts or rural areas in underdeveloped countries. The main idea behind this Transmission Status Propagation System is that whenever two or more vehicles get into communication range from each other, they exchange data regarding the transmission status perceived by the nodes. Each vehicle acquires this transmission status data by monitoring the access to the medium in its surroundings. Figure 4.7 illustrates how vehicles can acquire transmission status information and pass it on to other vehicles: in a) a vehicle detects the occurrence of a high concentration of message collisions in the portion of the network located in opposing lane, and carries this information with it; in b) the vehicle that previously detected the high concentration of message collisions joins the group of other vehicles travelling in the opposing lane, and during this contact the information is transmitted to the group's MLM.

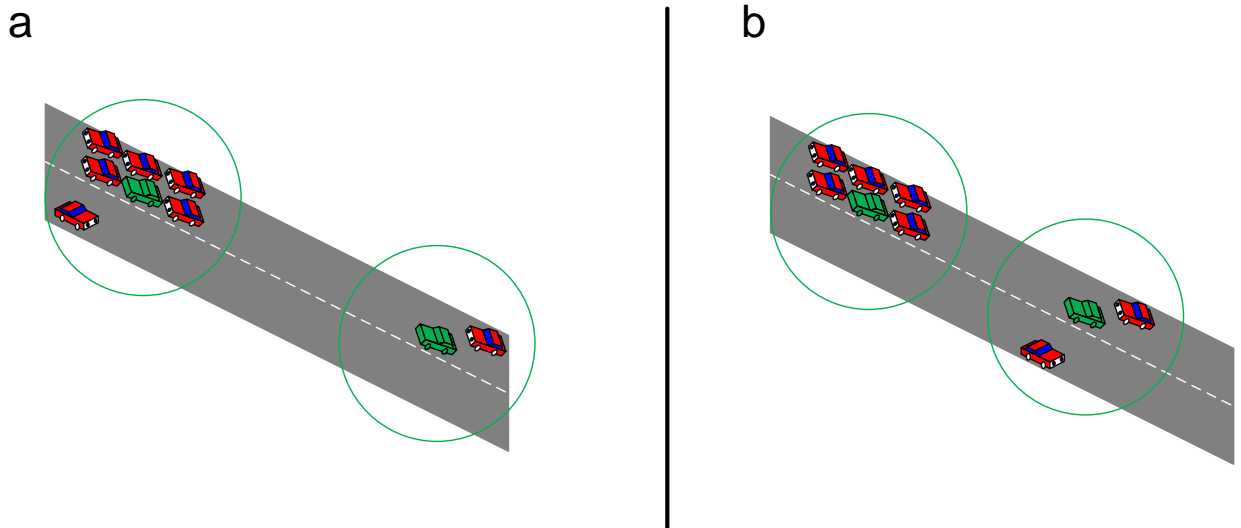


Figure 4.7. Transmission Status Propagation System supported by HE-MAN

This type of scenario benefits from the HE-MAN architecture due to its clustering algorithm and its ability to use MLMs to improve management tasks. When the vehicle

carrying the message collisions data gets in range with another cluster (Figure 4.7b) it immediately joins said cluster to transmit the data to the MLM. It is important to highlight that it is thanks to the inclusive nature of the HE-MAN's clustering algorithm that this vehicle can immediately join a cluster, even when it is travelling in the opposite direction of the cluster it is joining. Then, the MLM realizes through the received data that a traffic jam is possibly occurring ahead, which results in the MLM using local configuration to issue commands to the cluster members travelling towards the possible traffic jam, in order to deactivate non-essential VDTN applications (e.g., entertainment applications). This quickly deployed measure helps the vehicular nodes of this cluster to deal with the challenging communication environment they are about to face ahead.

4.6 Chapter Conclusions

This chapter introduced the HiErarchical MANagement (HE-MAN) Architecture, which uses the hierarchical approach to improve the reliability and the efficiency of management tasks. In the proposed hierarchical architecture, a Top-Level Manager (TLM) manages the entire network, delegating tasks to Mid-Level Managers (MLMs), which act as local managers in connected groups. The agents are the nodes managed either by a MLM or directly by the TLM.

A new clustering algorithm for VDTNs was proposed in order to implement a logical hierarchical topology of the vehicular nodes. This clustering algorithm maximizes the amount of time a node spends connected to one of the clusters, without using any sort of “filter” to prevent nodes from joining clusters for the sake of clusters' stability. The proposed clustering technique relies on periodic beacon messages, and it uses the nodes' connectivity history and nodes' addresses to elect cluster-heads. The MLM is always the cluster-head of each VDTN cluster.

Local monitoring allows near real-time management in a bounded portion of the network, allowing more complex and delay-sensitive applications/services, which would not be possible without a hierarchical network. Remote monitoring, in its turn, allows the TLM to monitor specific nodes of the VDTN without flooding the network. Both local and remote monitoring employ the publisher/subscriber model to reduce the amount of messages needed during monitoring tasks.

Connected groups of nodes also allow MLMs to perform more complex and delay-sensitive configuration tasks, through the usage of the proposed local configuration technique. Moreover, the TLM supports remote configuration tasks, which reduces the risk of a node being wrongly configured because of previous monitoring data that

became outdated due to the long transmission delays involved in VDTNs.

Finally, case studies are presented in order to illustrate how the proposed monitoring and configuration techniques would perform in some VDTN scenarios. These case studies also represent a qualitative analysis of the proposed techniques, while quantitative analyses will be provided in the next chapter.

Chapter 5

Evaluation: Methodology and Preparation

The research described in this thesis presents a set of algorithms for VDTN management. These algorithms need to undergo validation in order to check whether they are performing as designed or not. Section 4.5 presented a qualitative evaluation through case studies. This chapter presents the methodology used during the quantitative evaluation and discusses the achieved results.

5.1 Methodology

The methodology described to evaluate the proposed architecture is composed of a sequence of steps. These steps were the following:

1. Establishment of the objectives;
2. Selection of the metrics;
3. Selection of mobility traces;
4. Configuration of the simulation environment;
5. Execution of the simulations;
6. Results analysis.

The steps between the establishment of the objectives and the configuration of the simulation environment are explained in sections 5.2 to 5.5, while the remaining steps can be found in sections 5.6 to 5.8.

The HE-MAN architecture encompasses three main algorithms, each one approaching one of the following functionalities: **clustering**, **monitoring** and **configuration**. Therefore, three groups of experiments were designed in order to evaluate the proposed algorithms. Each group has its own characteristics (e.g., objectives, metrics, parameters, etc.), because the evaluated algorithms are designed to operate in different situations. The type of experiments chosen for the evaluation of the proposed algorithms are computer simulations. The main advantage of using a simulated VDTN for the evaluation is the benefit of a relatively high degree of realism without the high cost of implementing an actual vehicular network. The usage of testbeds with real vehicles provides the most realistic scenario for testing. However, the cost of acquiring the needed components for the testbed (e.g., vehicles, on-board units, roadside units, etc.) and the complexity of maintaining such environment under permanent control for an effective evaluation, especially in large-scale networks, is too high to be afforded in the context of the research presented in this thesis.

5.2 Objectives

Each group of experiments verifies whether the proposed algorithms are effectively overcoming the limitations they were made to do. The defined objectives follow:

5.2.1 Objectives for the clustering algorithm

1. To determine the best value for the configurable parameters used in HE-MAN's clustering algorithm;
2. To verify if the proposed algorithm can increase the amount of time nodes stay connected to a cluster;
3. To evaluate the characteristics of the clusters formed through the proposed algorithm, such as size, duration and stability;
4. To evaluate the quality of the communication within the clusters (i.e., communication delay and delivery ratio).

5.2.2 Objectives for HE-MAN monitoring

1. To verify if local monitoring can increase the delivery ratio and reduce the delivery delay of monitoring messages;

2. To evaluate the performance of remote monitoring;
3. To measure how the size of the nodes' buffer influences the performance of monitoring operations;
4. To investigate how different routing techniques influence the performance of monitoring operations.

5.2.3 Objectives for HE-MAN configuration

1. To verify the feasibility of the distribution of remote configurations to agents and MLMs.

5.3 Metrics

Metrics were selected for each group of experiments in order to quantify important aspects of the evaluated algorithms. This subsection introduces the selected metrics, along with their objective.

Clustering algorithm

1. *Cluster-connected time*: to evaluate the ability of HE-MAN's clustering algorithm to increase the time nodes spend connected to the established clusters;
2. *Cluster life time*: to measure the amount of time between the creation and the destruction of a cluster. Clusters with a long life time are desirable because the benefits provided by local communication within the clusters are made available for longer period of times;
3. *Number of created clusters*: this metric infers the overall logical organization of a network caused by the clustering algorithm being used, as well as the overhead of the clustering strategy;
4. *Cluster size*: the number of cluster members, including the cluster-head, provides insight on the reach of the created clusters;
5. *Number of cluster-head transferences per cluster*: a cluster-head transfer occurs when the cluster-head role is passed from one node to another. Stability is a desirable property for clustering algorithms, in order to avoid the overhead and possible interruption of functionalities caused by cluster-head transference operations;

6. *Local notification delay*: the interval between the creation of a message at a source node and its reception by the destination node within a local group. This metric verifies the freshness of local communication. The local notification delay is measured through ping messages among agents and the cluster's MLM;
7. *Local notification delivery ratio*: is the relation between the number of locally received messages and the total number of locally sent messages. This metric indicates the reliability of communication within clusters.

HE-MAN monitoring

8. *Notification average delay*: this metric measures the time a monitoring message takes to travel from a source to a manager, aiming to assess the ability of the evaluated proposals to cope with near real-time monitoring. A more formal statement of this metric can be found in the Definition 2.4.3 of Chapter 2;
9. *Average delivery ratio*: denotes the relation between delivered messages and total number of sent messages. This metric evaluates the reliability of the communication with the VDTN managers. The Definition 2.4.4 of Chapter 2 provides a formalization for this metric.

HE-MAN configuration

10. *Configuration command delivery probability*: to evaluate the efficacy of the configuration distribution schemes in getting the messages from the main VDTN manager to their final destination;
11. *Average delivery time for the configuration commands*: tests the freshness of the configuration messages received by the network nodes, in order to allow a more responsive and reliable management environment.

5.4 Mobility trace selection

In order to obtain as much realism from the experiments as possible, real mobility traces were employed. Mobility modeling is an important aspect of experiments with mobile networks, which intends to describe how the nodes move while the experiments are being carried out. Traces generated from real traffic monitoring provide a level of realism that is not achieved through synthetic models, since traces replicate the movement of vehicles as seen in a real world observation while synthetic models calculate the movement of nodes based on mathematical models.

Mobility traces collected from the Chicago Transportation Authority (CTA) Bus Tracker API were chosen for this research [Doering et al., 2010]¹. The buses that compose the fleet periodically report their position to a central server at CTA, which then becomes available to the general public. The traces were collected during 18 days in November 2009, and the generated database uses the UTM (Universal Transverse Mercator) system for geographical coordinates. Figure 5.1 shows a snapshot of the positions of buses (red dots) and bus stops (black dots) at a given moment captured in the trace files.

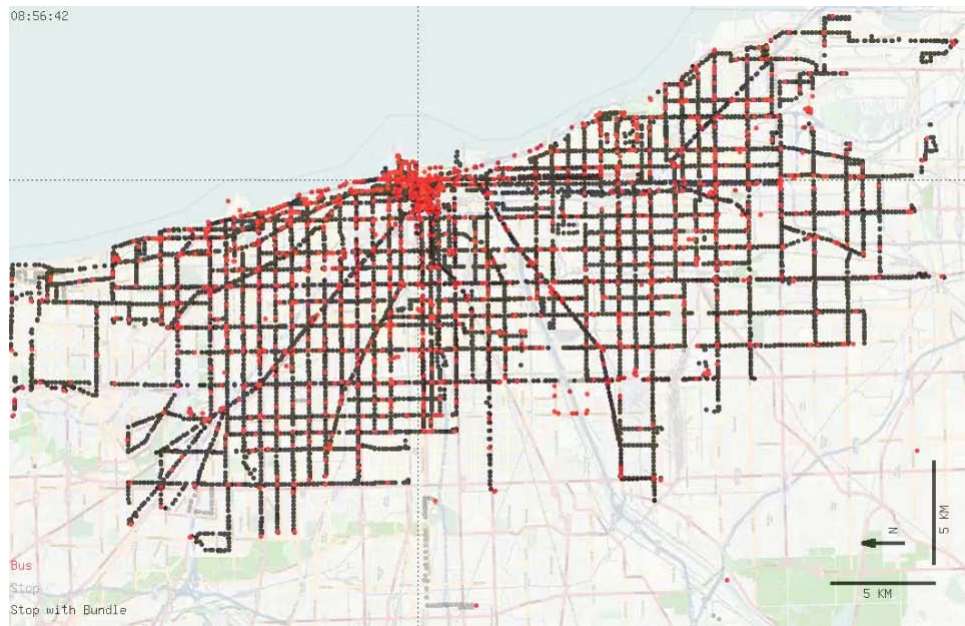


Figure 5.1. Snapshot of the buses (red dots) and bus stops (black dots) in the traffic trace [Doering et al., 2010].

One of the main advantages in adopting the traffic trace from the Chicago Transportation system is the fact that its nodes are sparsely distributed throughout the network area, which leads to a high number of partitioning of the network. This feature allows the simulation of scenarios that fit the challenges the HE-MAN architecture intends to overcome. Moreover, this specific trace contains information of a number of nodes higher than other traces considered in this research, which allows a better evaluation concerning the scalability of the solutions that are simulated. Finally, the long duration of the vehicles movement described in the traces allows simulations that last

¹As of September 2016, the mobility traces can be found at <https://www.ibr.cs.tu-bs.de/users/mdoering/bustraces/>

longer, making it viable to study effects that can appear in the network after prolonged usage of the simulated solutions.

Figure 5.2 presents a cumulative distribution function (CDF) of the duration of the contacts within the traces' nodes. The CDF shows that 25% of the contacts last less than 48 seconds, 50% of the contacts last less than 81 seconds, and 75% of the contacts last less than 129 seconds. The highest contact time registered is 2464 seconds long. Based on this data, one can conclude that although contacts are volatile, most of them are long enough to allow the exchange of significant amounts of data.

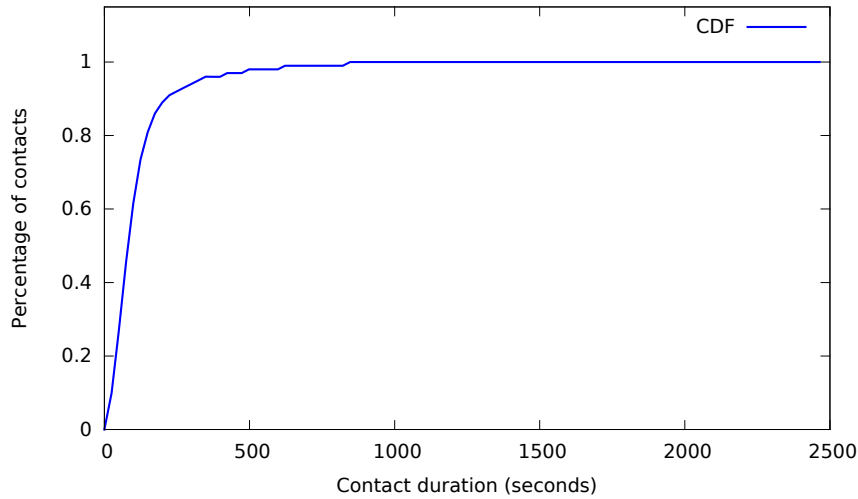


Figure 5.2. Cumulative distribution function (CDF) of contact duration within the trace files

In this research, the original trace files went through a data sanitization process, where a set of nodes with sample rate inferior to 30 seconds was selected. The reasoning behind this selection is to provide to the experiments a detailed description of the vehicles' movement, excluding large intervals without data about the location of a vehicle, which could compromise the realism of the nodes' movement. At the end of this process, 410 nodes were chosen, and the resulting trace file contains over 4 hours of mobility data. Finally, this trace file was divided into 4 traces with approximately 1 hour (3708s) of mobility data each. The number of vehicles in the simulations ranged from 100 to 400, and the total simulation area was approximately 30Km x 45Km ($1350Km^2$).

5.5 Simulation environment and configuration

The Opportunistic Network Environment (ONE) simulator [Keränen et al., 2009], version 1.4.1, was chosen to carry out the evaluations in this research. This is a DTN-specific simulator written in Java, and it is currently a well-established simulator that was used in a number of recent and important works regarding VDTN [Wang et al., 2015] [Xia et al., 2015].

Regarding the parameters used in the simulations, they are divided in general parameters and specific parameters. General parameters are the ones employed in all experiments realized in this thesis, while the specific parameters apply only to individual groups of experiments.

Table 5.1 presents the general parameters used in the simulations. Vehicles communicate with the network using the Dedicated Short Range Communications (DSRC) technology, which has a typical communication range of 300 meters [Bai and Krishnan, 2006]. The radios are configured to transmit data at 10Mbps, which is within the typical data rate range of DSRC [IEE, 2014]. All communications are half-duplex with constant bit-rate. The VDTN nodes are also equipped with persistent storage that provides 512MB for storing bundles. Simulation results are computed only after the first 100 seconds of the simulations, in order to allow a warmup period that allows the network to reach its steady state.

Table 5.1. General parameters of the simulations

Parameter	Value
Transmission Range	300m (DSRC)
Transmission Speed	10 Mbps
Buffer Size	512 MB
Available Routing Algorithms	Epidemic with oracle; Epidemic; Spray and Wait; Prophet
HE-MAN's clustering beacon interval	1 second

During the execution of the simulations there is no background traffic. The management traffic generated by the nodes follow the rules defined in the simulated applications only, and it is not based in any sort of traffic traces because, as far as this thesis' author is aware, such traces are not available in the literature.

For each evaluated metric the simulations are repeated four times, with each time using a different traffic trace having 3708 seconds of duration. The results are calculated with a confidence interval of 95%.

The simulations can employ four different routing algorithms, all available in The One Simulator, according to the targets of the simulations. The discard policy

employed by the routing algorithms is the removal of oldest messages first.

- **Epidemic:** the Epidemic routing algorithm provides high delivery ratios and low communication delays by making nodes distribute copies of messages to all other nodes in their local vicinity, until one copy of the message eventually reaches its final destination [Vahdat et al., 2000]. Despite the relatively high probability of messages being delivered to their destination with Epidemic routing, the high overhead caused by the amount of copies can sometimes degrade the overall performance of the network;
- **Spray and Wait:** it limits the number of message copies in the network, stopping the flood process when the limit of copies is reached [Spyropoulos et al., 2005]. This algorithm is more viable for real-life vehicular networks that have limited network resources, demanding an efficient usage of these resources. Whenever Spray and Wait was used in the simulations, the binary mode is activated, and the initial number of copies for a message is 15% of the number of vehicles in each scenario, in order to make the configuration of this algorithm compatible with previous works in the area [Soares et al., 2012];
- **Prophet:** the Probabilistic ROuting Protocol using History of Encounters and Transitivity (Prophet), as its name suggest, is a routing algorithm that make use of the predictability of nodes movement to improve the routing process [Lindgren et al., 2003]. It employs a probabilistic metric called **delivery predictability** was created, which estimates the likelihood of a node to be able to deliver a message to another node;
- **Epidemic Routing with Oracle:** this algorithm is an optimized version of the Epidemic Protocol, where bandwidth is unlimited and message sizes are negligible [Nelson and Kravets, 2010]. The Epidemic with Oracle can be considered a best case scenario in DTN routing, and serves as an optimal DTN routing algorithm.

This research employs the taxonomy proposed by Chaintreau *et al.*, which classifies opportunistic routing algorithms into **oblivious forwarding algorithms** and **informed forwarding algorithms** [Chaintreau et al., 2007]. Oblivious algorithms do not employ information about other nodes (e.g., identity of devices that are met, recent history of contacts, time of day, etc.) for guiding its forwarding process, while informed algorithms makes use of this type of information to decide to which nodes to forward messages. The Epidemic and Spray and Wait algorithm are two representatives of oblivious algorithms: the first represents an upper bound of delivery performance, but

with high consumption of network resources, while the latter represents a lower bound of delivery performance, but with low consumption of network resources. Prophet, in its turn, is a representative of an informed algorithm. As for the Epidemic with Oracle, it was chosen to assess how close other routing algorithms are to a best-case routing scenario.

As for the specific parameters, they are presented below according to the group of experiments they belong to.

5.5.1 Clustering algorithm

The performance of HE-MAN's clustering algorithm is compared to the performance of another related clustering algorithm. MDMAC, which is first presented in Chapter 3 of this thesis, was chosen as a base of comparison because it is one of the most recent algorithms that share an important design principle with the clustering technique designed with HE-MAN: they both use information on the movement of the vehicles to decide how to cluster network nodes. Both HE-MAN's clustering algorithm and MDMAC are based on periodical broadcasting of beacons to detect topology changes. Moreover, MDMAC is a relatively simple algorithm that is described with plenty of details in the literature, which allows an easy implementation. The decision making of both MDMAC and HE-MAN's clustering algorithms relies only on the vehicles' movement speed and direction, and it does not require other sources of information, such as GPS or information about the roads' lanes. Regarding the differences between these two clustering techniques, MDMAC is a representative of a large group of algorithms that employ criteria for determining if a node will be able to join a cluster or not, for the sake of increased cluster stability. This design differs from our proposed clustering algorithm, which maximizes the time where vehicles are members of clusters, without preventing vehicles to join clusters. This difference between HE-MAN's clustering algorithm and MDMAC allows for an interesting comparison on how filtering the nodes that are within communication range of a cluster can affect the performance of the hierarchical management scheme adopted in the HE-MAN architecture.

In HE-MAN's clustering, the minimum contact time threshold necessary for MLM transfer within cluster and cluster merging operations was determined through the analysis of the contact duration in the mobility traces, previously presented in Figure 5.2. Considering that approximately 50% of the contacts have a duration superior to 80 seconds, it was established that the minimum contact duration of 80 seconds must occur in order to MLM transfers and clusters merging to be considered by HE-MAN's clustering algorithm.

In summary, the following parameters were used for HE-MAN's clustering (an empirical evaluation of the parameters values can be found in Subsection 6.1.1):

- The time threshold where agent nodes tolerate the absence of received MLM beacons without abandoning their current cluster is set to 3 seconds;
- A minimum time threshold of 80 seconds of continuous contact was defined as a condition for the start of MLM transferences inside clusters and clusters merging operations;
- The value of lambda in the EWMA used for calculating the connection score metric was set to 0.2.

The MDMAC algorithm was configured with the same values as the paper where the algorithm was originally presented [Wolny, 2008]. The MDMAC algorithm has three parameters. The first parameter is the ANGLE THRESHOLD, which determines a maximum allowed angle between the direction vector of two nodes in order to determine whether these two nodes can be part of the same cluster. The other two parameters are the FRESHNESS constant, which represents an initial estimation of the time of contact between two nodes, and the FRESHNESS THRESHOLD, which determines a minimum contact time before the node is accepted as a member of a cluster. The MDMAC configuration that was used is the following:

- The ANGLE THRESHOLD constant used in the algorithm was set to 45 degrees;
- The FRESHNESS value was set to 30 beats (30 seconds) and the FRESHNESS THRESHOLD value was set to 2 beats (2 seconds).

The simulations evaluating the clustering algorithms employ the parameters presented in Table 5.2.

Table 5.2. Simulation parameters for the VDTN clustering techniques

Parameter	Value
HE-MAN's lambda	0.2
HE-MAN's beacon wait maximum threshold	3 seconds
HE-MAN's continuous connection minimum threshold	80 seconds
Employed routing algorithm	Epidemic Routing
MDMAC's beacon interval	1 second
MDMAC's ANGLE THRESHOLD	45 degrees
MDMAC's FRESHNESS	30 beats (30 seconds)
MDMAC's FRESHNESS THRESHOLD	2 beats (2 seconds)

5.5.2 HE-MAN monitoring

The performance of HE-MAN's monitoring solution was evaluated in comparison with a hypothetical VDTN monitoring solution, based on the classic centralized monitoring approach widely used in traditional networks. This is due to the lack of existing monitoring solutions for VDTNs at the time of the evaluation. Both HE-MAN's monitoring approach and the baseline solution, called **Simple VDTN Monitoring**, are described as the following:

HE-MAN's monitoring solution: In this scenario, the VDTN is organized with a TLM placed in a fixed position at the center of the simulation area, and the vehicular nodes moving around the TLM, either within the established clusters (as MLMs or managed agents) or travelling in an isolated manner. The MLMs are elected using the HE-MAN's clustering algorithms. The publisher-subscriber monitoring model is employed in the communications between managers and agents, with agents issuing monitoring messages to subscribed MLMs whenever an issue is detected. Also, MLMs aggregate the remote reports issued by the event logging system of the vehicles within the local group in order to forward to the TLM a single remote report bundle, in order to reduce the overhead of messages travelling through the VDTN.

Simple VDTN monitoring: This scenario, inspired in traditional network management approaches, employs a single manager at the center of the simulation area monitoring the VDTN. For the sake of fairness, the publisher-subscriber approach will also be employed in this scenario.

On top of the two network management strategies described above, two different applications were implemented to be run by the vehicular nodes during the simulations. The main purpose of these applications is to generate the messages that will be analyzed by the simulated management solutions, triggering events such as a node issuing a fault notification message to its manager. The two applications used in the monitoring-related simulations are the following:

- The first is a safety application called *Collision Avoidance*. It uses sensors installed in the vehicles to monitor its surroundings. Collision avoidance beacon messages are sent every 100ms (which is the typical beacon interval for vehicular safety applications [Kargl et al., 2008]), and each message contains data such as the vehicles' current position, speed, and direction. A beacon message can become incorrect during its transmission, due to a faulty sensor generating invalid measurements in the source vehicle, for example. Whenever a vehicular node receives one of these incorrect beacon messages, the receiver issues a fault notification message to the cluster's MLM, in order to allow the MLM to react to the

problem (e.g., issuing commands to correct the problem, notifying other nodes of the cluster of the existence of a faulty vehicular node, etc.). This application evaluates the ability of the monitoring solutions for working effectively in situations involving high data rates and a demand for low response times (typically in the order of hundreds of milliseconds).

- The second application is an **Event Logging** system used for the operational management of fleets. This application ensures safety and operational efficiency, providing warnings for broken parts, unscheduled stops during a travel, or vehicles exceeding a maximum speed threshold. The data collected by the event logging system is sent periodically to a manager of the VDTN. The event logging system tests how the monitoring solutions deal with non-real-time and low data rate applications, which accepts delays in the order of hours or even a few days. In our simulations, this application reports data concerning distances traveled and registered speeds for each vehicle.

Parameter values for the monitoring evaluations can be found in Table 5.3. The size of the beacon messages issued by the Collision Avoidance system was 64 bytes, while the size of the fault notification messages sent by the VDTN monitoring system whenever a faulty beacon message is detected was 1024 bytes [Kato et al., 2013] [Biswas et al., 2006]. The probability of a beacon from the Collision Avoidance system being corrupted is 10%. This intentionally high probability was chosen in order to test the ability of the evaluated monitoring systems to cope with a relatively demanding situation. The remote reports issued by the Event Logging system have 512KB of size (hypothetical size relative to 30 minutes of monitoring data of the cluster's members), and are sent by the vehicular nodes to a network manager every 30 minutes. Messages sent to remote nodes using DTN routing (Epidemic with Oracle, Spray and Wait or Prophet) had their Time-To-Live (TTL) parameter set to 60 minutes. The TTL value represents a maximum delay the bundles can tolerate, and it was set to 60 minutes in order to approximately match the total simulation time, which is 3708 seconds.

5.5.3 HE-MAN configuration

The specific parameters regarding the simulation of the VDTN configuration scenarios are presented in Table 5.4. The configuration message size generated by the main manager in both scenarios is 10KB (hypothetical size of a configuration script containing a set of conditions and configuration commands). After the warmup phase, the simulation time is divided in configuration generation ticks, with each tick occurring every

Table 5.3. Simulation parameters used for the monitoring evaluation

Parameter	Value
Collision avoidance beacon size	64 bytes
Fault notification messages size	1024 bytes
Fault probability for collision avoidance beacons	10%
Remote report messages interval	30 minutes
Remote report messages size	512KB
Used routing algorithms	Epidemic with Oracle; Epidemic; Spray and Wait; Prophet
TTL for Remote report bundles	60 minutes

5 minutes. Immediately after a configuration generation tick, the TLM generates 10 configuration messages to 10 different vehicles. The vehicles are randomized in every configuration generation tick. The Epidemic with Oracle routing was used, in order to evaluate the performance of the distribution of configuration messages in a best-case scenario routing wise. Finally, bundles carrying configuration messages have the TTL configured to 60 minutes.

Table 5.4. Simulation parameters used for evaluating VDTN configuration techniques

Parameter	Value
Configuration message size	10KB
New configurations generation tick time	5 minutes
Number of new configurations generated per tick	10
Used routing algorithms	Epidemic with Oracle
TTL for configuration bundles	60 minutes

5.6 Chapter Conclusions

This chapter presented the methodology and preparation process for the quantitative evaluation of the main algorithms that are part of the HE-MAN architecture. Three main groups of evaluations were defined, according to the main functionality of their algorithms: clustering, monitoring and configuration. For each group, objectives, metrics and the main parameters used in the experiments were defined. A DTN-specific simulator was employed together with vehicular traffic mobility traces, in order to provide simulations realistic enough to draw significant conclusions, while maintaining the whole simulation process cost-effective. In the next chapter the simulation results are presented and analyzed.

Chapter 6

Evaluation: Results and Analyses

This chapter presents and analyzes the results of the experiments defined in Chapter 5. Once more, the experiments are grouped in three main groups: clustering algorithms, monitoring, and configuration.

6.1 Clustering algorithm evaluation

The clustering evaluation is divided into two parts: the first part aims to determine the best possible values for the parameters in HE-MAN's clustering algorithm. The second part evaluates the performance of the proposed clustering algorithm in comparison with MDMAC.

6.1.1 Tuning of the configurable parameters in HE-MAN's clustering algorithm

As presented in Chapter 4, HE-MAN's clustering algorithm has three configurable parameters: a minimum contact time threshold necessary for MLM transferences within clusters and cluster merging operations, a maximum time threshold without the reception of beacon messages for a contact to be considered extinct (contact timeout), and the EWMA's lambda (λ) attenuator value used for the calculation of the connection score metric. The minimum contact time threshold for MLM transferences is already explained in Subsection 5.5.1.

The two remainder configuration parameters are evaluated next. The lambda used in the calculation of the connection score varied from 0.05 to 0.9. As for the contact timeout, it varied from 1 second to 5 seconds. These two parameters will be analyzed according to the metrics defined for clustering algorithms, in order to

determine ideal values for them. Finally, the simulated network has 400 nodes, which is the densest scenario considered in this thesis.

Results and analyses

The first analyzed metric is the average time which nodes spend connected to clusters (metric 1), which is illustrated in Figure 6.1. As the value of lambda varied, the average cluster-connected time presented little variation regardless of the contact timeout for contact loss. The highest average cluster-connected time was registered when lambda was 0.7 for 1 second contact timeout, 0.2 for 3 seconds contact timeout, and 0.1 for 5 seconds contact timeout.

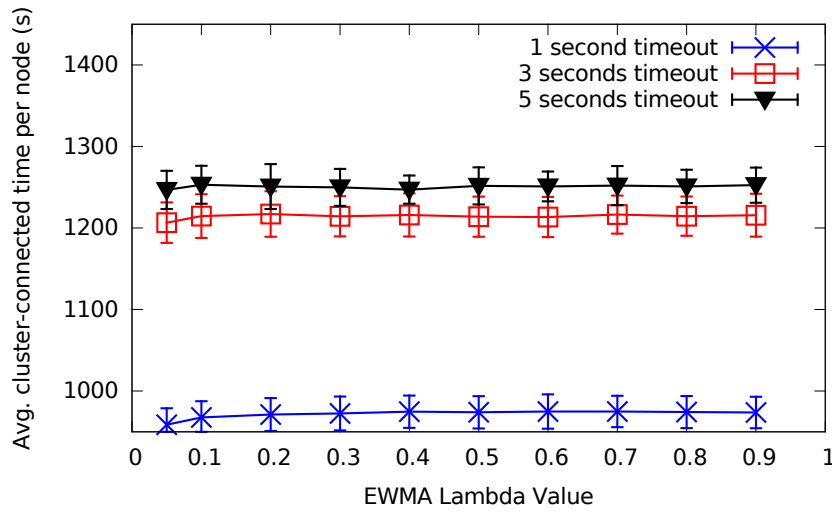


Figure 6.1. Average time that nodes spend connected to clusters

Next, the average lifetime of the created clusters was evaluated (metric 2), as presented in Figure 6.2. Once more, the results presented little variation, with the clusters lasting approximately 8 to 9 seconds when the contact timeout is 1 second, 50 to 53 seconds when the contact timeout is 3 seconds, and 57 to 60 seconds when the contact timeout is 5 seconds. The longest cluster lifetime values are achieved when lambda is 0.4 for 1 second contact timeout. With the contact timeout at 3 or 5 seconds, the longest cluster lifetime is achieved when lambda is 0.05. This difference caused by the values of the contact timeouts can be explained by the higher tolerance that clusters have to member disconnection as this timeout increases. Therefore, the probability of a cluster existing for a longer period of time is higher.

The average number of formed clusters (metric 3) for every performed simulation is represented in Figure 6.3. When the contact timeout for a contact to be considered lost is 1 second, clusters tend to be destroyed and created very frequently because nodes

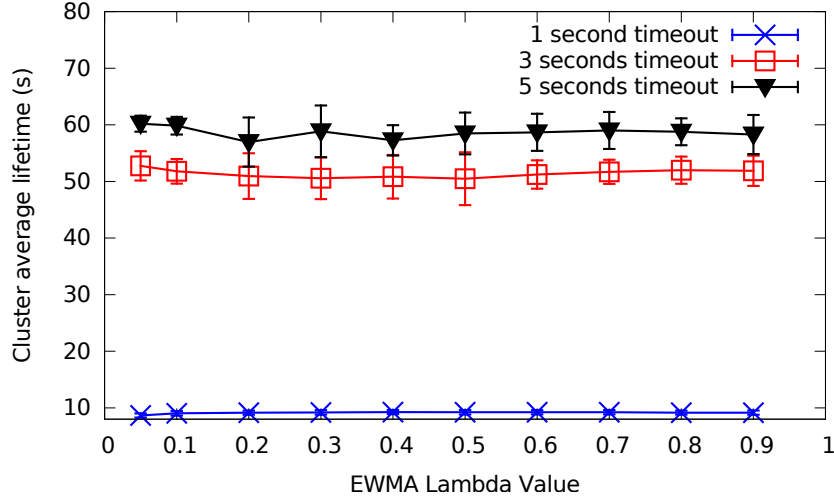


Figure 6.2. Average lifetime of a cluster

can be disconnected for a brief amount of time (up to 1 second) just to be reconnected a moment later. The graph shows that the number of created clusters when the contact timeout is 1 second is around 18000, regardless of the value of lambda. On the other hand, a longer tolerance regarding the disconnection of nodes reduces the chances of unnecessary reclustering, but if the contact timeout for a contact to be undone is too high, cluster members can wrongly consider other nodes to still be part of the cluster and try to communicate with them, when such communication would not be possible anymore. When the contact timeout is 3 seconds, the number of created clusters is around 4500, while the number of created clusters when the contact timeout is 5 seconds is around 4000.

The average size of the clusters (metric 4) is represented in Figure 6.4. The size of the clusters provides insights on the complexity of the communication within the clusters formed with the proposed algorithm. The graph shows that the size of the clusters remained almost the same, hovering around 2 members, regardless of the value of lambda or the contact timeout for contacts to be considered lost. The biggest average cluster size is seen when the contact timeout for a contact to be considered lost is 3 or 5 seconds, which is expected since this allows the membership of the cluster nodes to be more resilient to brief disconnections. Although clusters with size larger than two were registered, they were vastly outnumbered by clusters with size two, which brought the averages to the values seen in the graph. The main reason behind this is the fact that the nodes in the mobility trace are mostly scattered around the simulation scenario, with few zones of concentration, which favors smaller clusters rather than bigger ones.

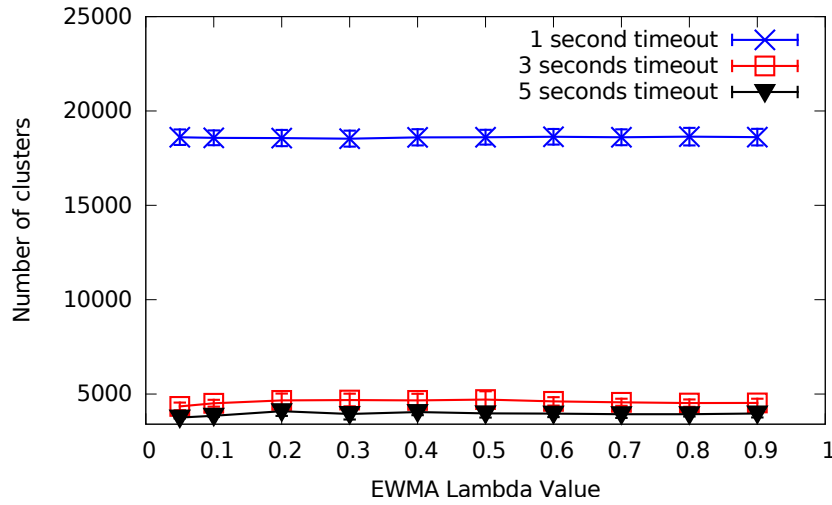


Figure 6.3. Average number of clusters formed during the experiments

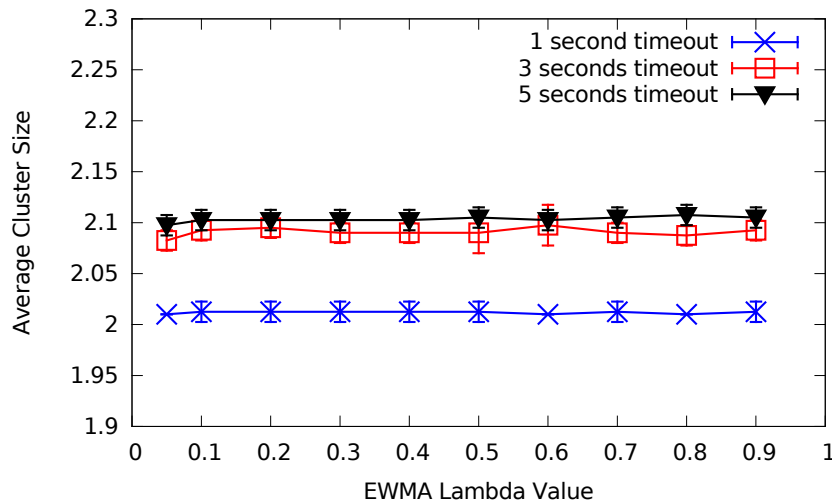


Figure 6.4. Average size of clusters formed during the experiments

The average number of cluster-head transferences per cluster (metric 5) is presented in Figure 6.5. In the case of HE-MAN's clustering algorithm, a cluster-head transference happens when the MLM role within a cluster is transferred to one of the cluster's agents, or when two clusters are merged into a single cluster. The lambda variable has a bigger impact on this metric in comparison to the other metrics, and results show a tendency of the number of average cluster-head transferences per cluster to go down as the value of lambda goes up. The exception occurs when the contact timeout for a contact to be considered lost is 1 second, which leads to almost zero

cluster-head transferences per cluster occurring. The reasoning for this is that when the contact timeout is 1 second, the clusters are too unstable for cluster-head transference operations to take place. The highest amount of cluster-head transferences was registered when lambda was 0.05 for contact timeouts equal to 3 or 5 seconds, while the least amount of cluster-head transferences occurred when lambda was 1.0. However, the average number of cluster-head transferences per cluster ranged from 0 to 0.08, which can be considered reasonably low regardless of the value of lambda or contact timeout for contacts to be considered lost.

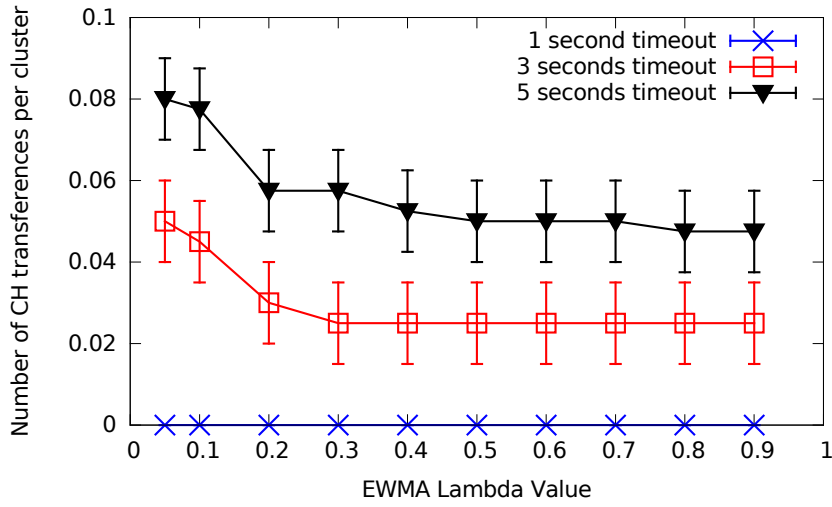


Figure 6.5. Number of cluster-head transferences per cluster

Figure 6.6 shows the results concerning the local notification delay (metric 6). The analysis of the graph shows that the contact timeout for a contact to be considered lost has a higher influence over the local notification delay. The average delay is approximately constant as the value of lambda is increased, being around 0.97 second when the contact timeout is 1 second, around 1.1 second when the contact timeout is 3 seconds, and around 1.3 second when the contact timeout is 5 seconds. The main limiting factor regarding local notification delay is message queueing in the cluster's nodes, preventing messages to be delivered as soon as they are created. Nonetheless, local notification delay revolves around one second.

Results on the measurement of local notification delivery ratios (metric 7) are illustrated in Figure 6.7. Varying the value of lambda imposes little influence over the obtained delivery ratio. On the other hand, varying the contact timeout for a contact to be considered lost results in more significant variations of the local delivery ratio. The average delivery ratio revolves around 97 percent when the contact timeout is

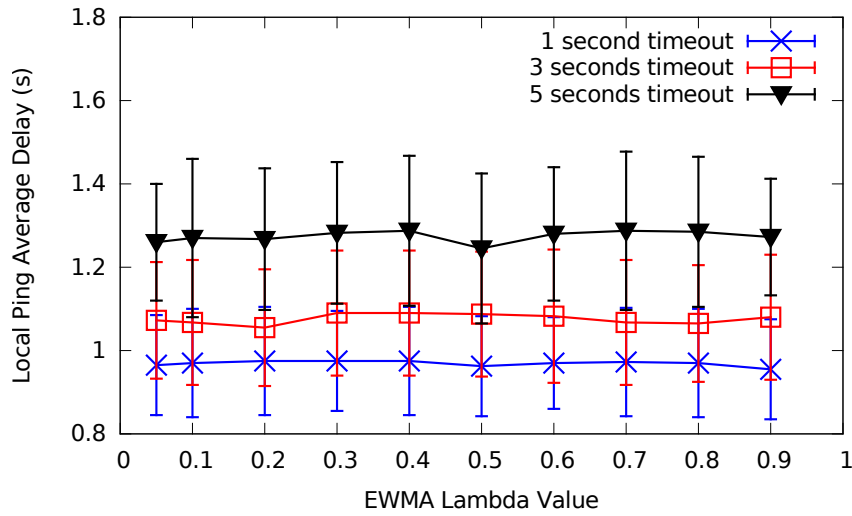


Figure 6.6. Notification delay in local communication

1 second, around 96 percent when the contact timeout is 3 seconds, and around 92 percent when the contact timeout is 5 seconds. The main reason behind the losses in the delivery ratio resides mostly on agents unable to deliver messages to the cluster-head, or vice-versa, because the destination is no longer in communication range of the other nodes. This issue is aggravated by a higher contact timeout, since it results in more cluster members assuming that other members are still within communication range, when said members may already have left the cluster's communication range.

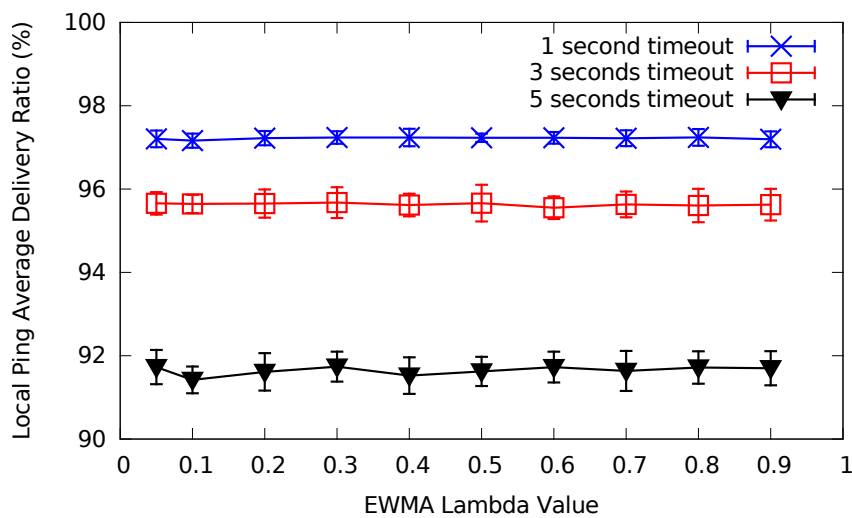


Figure 6.7. Notification delivery ratio in local communication

Conclusions

Considering the results presented in this subsection, it was decided that the lambda value to be used in the other simulations of this thesis is **0.2**. One of the reasons for this is the importance of maximizing the time nodes spend connected to one of the VDTN clusters. The longer a node stays connected to a cluster, the longer it benefits from local management. It is also important to notice that the small variations seen in the results of some metrics with different values of lambda do not mean that the value of lambda has little impact over the clustering process. What it means is that the vast majority of the clusters did not exceed two members in this particular trace. The variation in the metrics' results will increase as different values of lambda are provided if the overall number of clusters' members increase, which is achieved in denser networks.

As for the contact timeout for a contact to be considered lost, the limit of **3 seconds** will be employed in the simulations, since it provides a good balance between cluster stability and performance of local communication within the clusters (concerning transmission delay and delivery ratio). Finally, a minimum contact duration of **80 seconds** must exist for MLM transfers or clusters merges to happen. Once more, Table 5.2 summarizes the selected values for the HE-MAN's configurable parameters.

6.1.2 HE-MAN's clustering algorithm performance analysis

Figure 6.8 shows the average amount of time that a node spends connected to one of the network clusters. Since HE-MAN's clustering algorithm does not employ any sort of filter for a node to join a cluster, it performs better in this metric than the MDMAC algorithm. Moreover, as the density of the network increases, the difference of performance between HE-MAN's clustering and MDMAC also increases. The average connection time achieved by HE-MAN's clustering algorithm varies from 434 seconds in simulations with 100 nodes to 1213 seconds in simulations with 400 nodes, while the results achieved with MDMAC ranged from 202 seconds in simulations with 100 nodes to 581 seconds in simulations with 400 nodes. The MDMAC algorithm presents less variation in the results, which can be explained by the careful selection of the nodes that are able to join the network clusters.

Figure 6.9 illustrates the average lifetime of the clusters formed with HE-MAN's and MDMAC's algorithms. A cluster is created when at least two nodes get in the communication range of each other and a CH is elected. On the other hand, a cluster is destroyed when all members of a cluster lose communication with the cluster-head. This metric is used in the investigation of the stability of the clusters produced by the

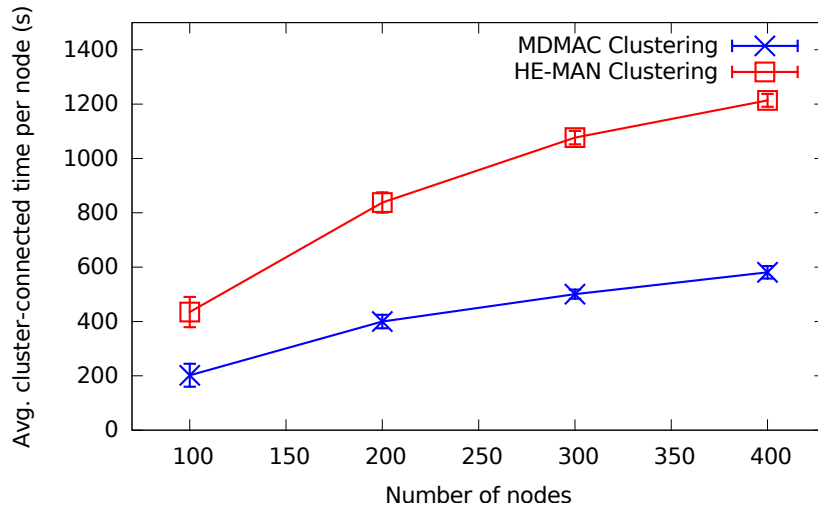


Figure 6.8. Average time that nodes spend connected to clusters

studied algorithms. Because of the filtering, MDMAC tends to perform better on this metric. The figure shows that, indeed, MDMAC's clusters have a higher lifetime in the simulations with 100 nodes, in comparison to HE-MAN's clustering. However, this difference is reduced in the scenarios with 200, 300 and 400 nodes. This shows that the MDMAC's filtering of nodes trying to join clusters is more effective in sparser networks, with this effectiveness being reduced as the network becomes denser. Clusters' lifetime in MDMAC's algorithm ranges from 112 seconds (100 nodes) down to 50 seconds (400 nodes), while in HE-MAN it ranges from 80 seconds (100 nodes) down to 50 seconds (400 nodes).

The average number of clusters created with both algorithms is presented in Figure 6.10. The graph shows that the amount of clusters is always higher in HE-MAN's algorithm than MDMAC's. The reasoning for this behavior resides one more time on the filtering process that MDMAC imposes over nodes trying to join/create clusters, limiting the number of clusters. On the other hand, HE-MAN's algorithm creates more clusters because whenever two or more nodes make contact with one another they are able to establish a cluster. With MDMAC the number of created clusters per simulation varies from 97 clusters in the simulations with 100 nodes to 2527 clusters in the simulations with 400 nodes. HE-MAN's algorithm, in its turn, created from an average of 283 clusters in the simulations with 100 nodes to 4707 clusters in the simulations with 400 nodes.

The average size of the created clusters (total number of nodes within a cluster, including the cluster-head) is shown in Figure 6.11. The differences in both algorithms

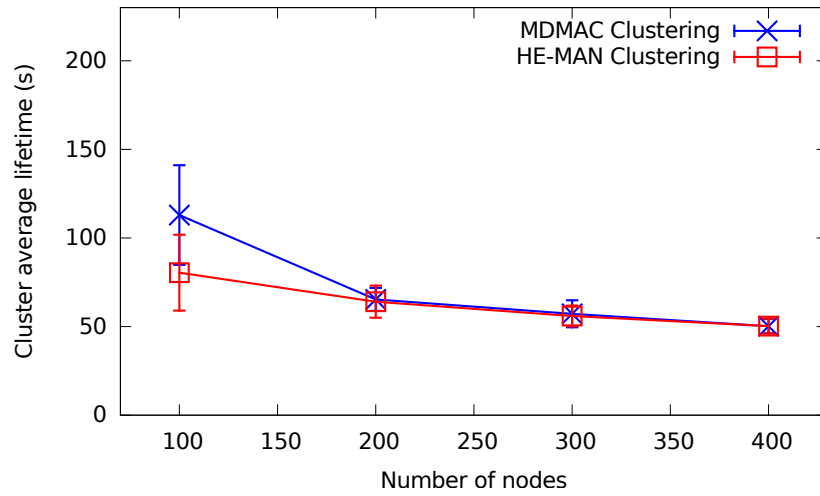


Figure 6.9. Average lifetime of a cluster

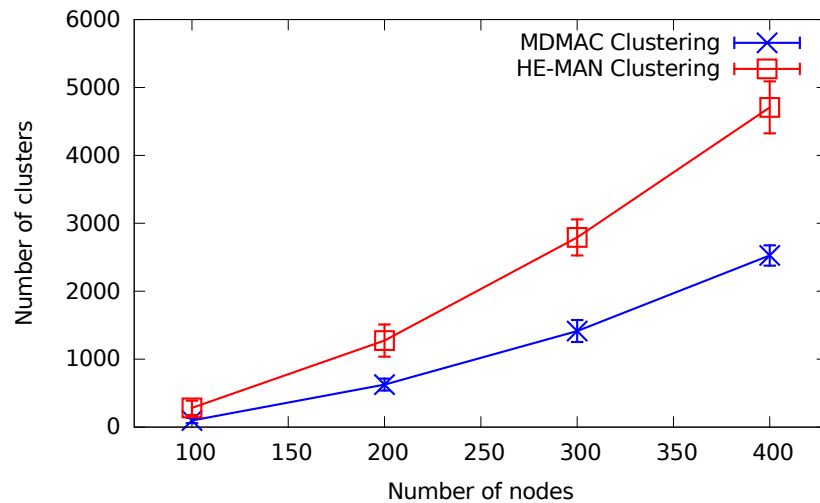


Figure 6.10. Average number of clusters formed during the experiments

are small, with HE-MAN and MDMAC forming clusters with approximately 2 nodes in all simulated scenarios (from 100 to 400 nodes). The main reason for the limited size of the clusters is due to the sparse nature of the mobility trace used in the simulations. Although there are some regions where the concentration of nodes is higher, most of the simulations are composed by sparsely distributed nodes, which hinders the creation of a higher number of clusters with more than two members.

The cluster-head transferances per cluster is an indicator of how stable a cluster is, since well picked clusterheads usually do not need to be changed in short periods of time.

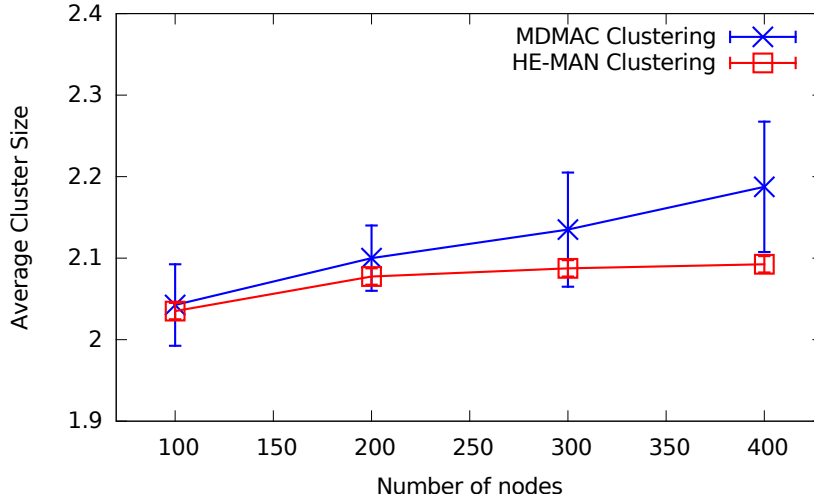


Figure 6.11. Average size of clusters formed during the experiments

Figure 6.12 shows that HE-MAN's clustering algorithm needs fewer CH changes per cluster in comparison to MDMAC. HE-MAN's algorithm registered from approximately 0.03 to 0.04 cluster-head transferences per cluster in all simulated scenarios, while MDMAC registered from 1.04 cluster-head changes per cluster (100 nodes scenario) to 1.44 cluster-head changes per cluster (400 nodes scenario). The probable explanation for this difference is the usage of a minimum connection time threshold in HE-MAN. This threshold restricts cluster-head transferences only to the cases where the cluster-head and cluster-head candidate are connected for a significant amount of time, which drastically reduces the overall number of cluster-head transferences. On the other hand, MDMAC allows cluster-head changes whenever the weight of the cluster-head candidate is higher than the weight of the current cluster-head, and the freshness metrics and the angle between the movement vectors of the current cluster-head and the candidate cluster-head are within the pre-established thresholds.

Figure 6.13 shows the average delay achieved through the clustering algorithms during intra-cluster communication. For both algorithms the average delay was typically inferior to 2 seconds, yet HE-MAN's clustering managed to perform better in this metric than MDMAC. While the local communication delay with MDMAC ranged from 0.3 second (100 nodes) to 1.9 second (400 nodes), in HE-MAN's this delay ranged from 0.3 second (100 nodes) to 1.1 second (400 nodes). The reason for the better performance of HE-MAN over MDMAC resides on the fact that MDMAC takes longer to realize when a node is not connected to a cluster anymore, eventually being able to deliver messages to nodes that spent a few seconds disconnected from the cluster,

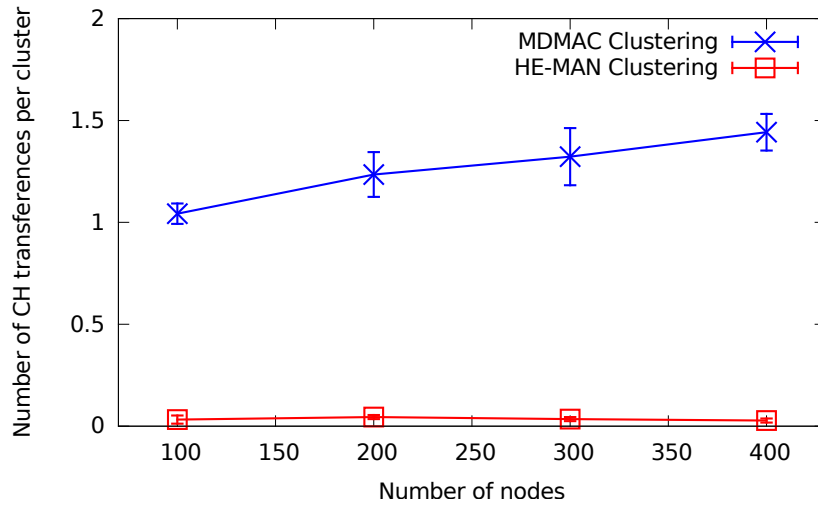


Figure 6.12. Number of CH transferences per cluster

which increases the average delay.

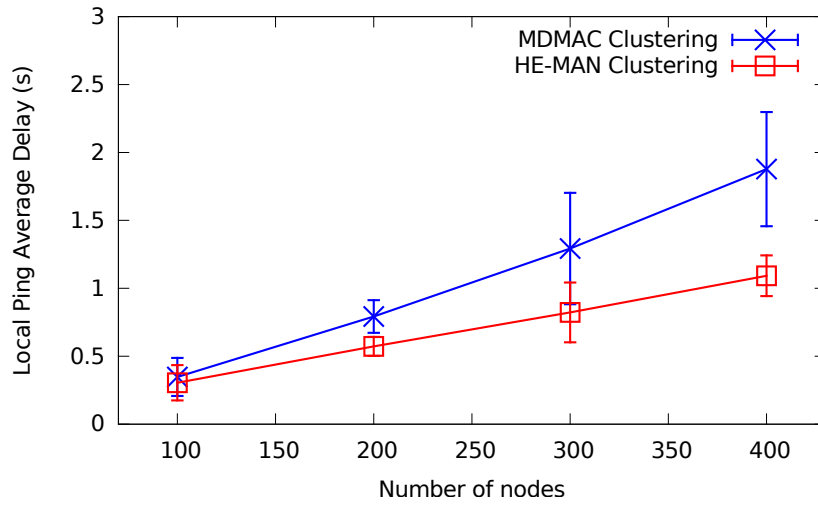


Figure 6.13. Notification delay in local communication

The delivery ratio in local communication is represented in Figure 6.14. Once more, HE-MAN's clustering algorithm and MDMAC presented similar performance in more sparse scenarios, while HE-MAN achieved better delivery ratio in denser scenarios. With MDMAC, the registered delivery ratios ranged from 96% (100 nodes) down to 87% (400 nodes), while HE-MAN's clustering algorithm registered delivery ratios from 97% (100 nodes) down to 96% (400 nodes). Once more, the explanation for this difference can be found on the higher tolerance MDMAC for temporary disconnections.

While waiting for another member to reconnect, a cluster member may issue new messages to the disconnected node expecting to be able to deliver these messages, when the expected reconnection of the missing member can simply not occur. This problem is less recurring in HE-MAN's clustering algorithm due to its smaller tolerance for cluster members that become unreachable.

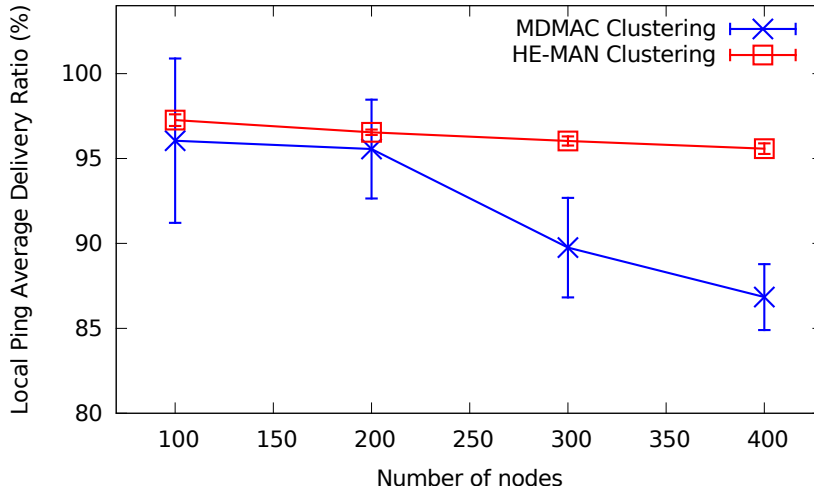


Figure 6.14. Notification delivery ratio in local communication

The results presented in this subsection show that the proposed HE-MAN's clustering algorithm increases the amount of time nodes spend being members of clusters in comparison to MDMAC. Moreover, the clusters formed with HE-MAN's clustering algorithm were stable enough to keep the number of cluster-head transferances and the local communication delay lower than the MDMAC, while being able to provide a higher delivery ratio. On the other hand, the number of created clusters is considerably higher in the HE-MAN's clustering algorithm in comparison to MDMAC, which can lead to a higher overhead related to the messages issued during clusters creation.

6.2 Monitoring solution evaluation

This section presents an evaluation of the impact that the size of the buffer in the nodes has on the overall communication performance of the network, a comparison of the communication performance between a centralized monitoring approach and the proposed HE-MAN monitoring scheme, and the impact that DTN routing algorithms can have over the evaluated monitoring solutions.

6.2.1 Analysis of the impact of the size of buffer in HE-MAN Monitoring

Before comparing the proposed monitoring scheme of HE-MAN architecture to another monitoring solution, it is important to discover how the size of the buffer impacts the communication delay and delivery ratio. To this end, the proposed HE-MAN monitoring scheme was simulated with different buffer sizes, from 8MB up to 512MB. The number of nodes was fixed at 400, in order to provide a high load of messages. Moreover, the routing algorithm that among the DTN algorithms has the highest requirements for network resources was used: the Epidemic routing. The communication delay and delivery ratio for both local and remote communication were measured, in order to discover the point where the performance of HE-MAN monitoring starts to decay due to lack of free buffer space.

Figure 6.15 shows how the local communication delay related to the transmission of fault notification messages behaves with different buffer sizes. The delay measured in the experiments is almost constant, with very little variation, as shown in the graph. The notification delay with the tested buffer sizes ranged from 1.45s to 1.49s. The reason for this small variation is the fact that the nodes involved in the communication of fault notifications are constantly connected to each other within the local groups, which allows the messages to be readily delivered, without the need for them to be stored in the buffer for long periods of time. Naturally there are situations where the connection between nodes in local groups can be broken for brief periods of time, due to the mobility of the nodes in the simulation, which results in some higher notification delays being registered, rising the average of these delays.

The delays involved in the transmission of remote reports to the network's TLM are represented in Figure 6.16. When the buffer size varied from 512MB down to 64MB, the average delay of remote reports remains constant, around 952 seconds (a little over 15 minutes). However, starting from 32MB down to 8MB, the average delay declines up to approximately 647 seconds (almost 11 minutes). The explanation for this reduction is the fact that with buffer size smaller than 64MB less remote reports are being delivered to the TLM, especially the ones that would take long to reach their destination. Therefore, only the reports that needed to spend less time stored in the nodes' buffers managed to reach the TLM.

The average delivery ratios concerning the local transmissions of fault notifications are represented in Figure 6.17. The delivery ratio remained approximately constant at 98.5%, regardless of the size of the buffers. Once more, local messages spend little time in the nodes' buffers in comparison to messages sent through remote

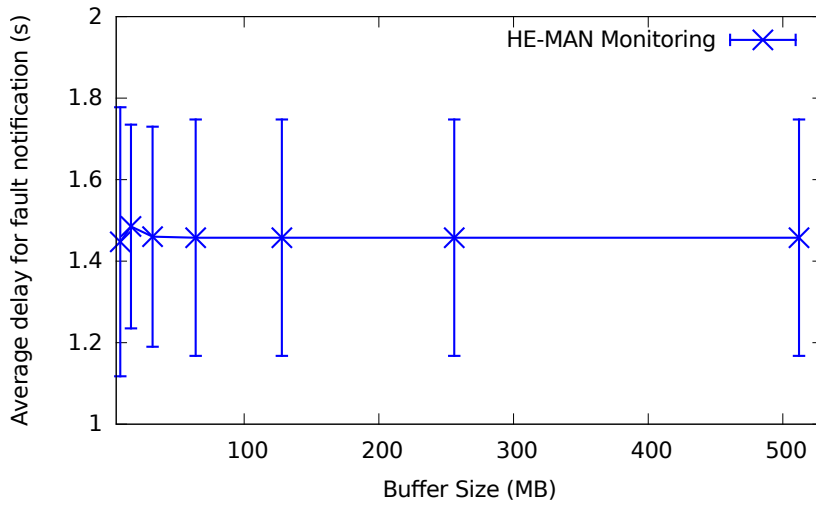


Figure 6.15. Average local notification delay with different buffer sizes

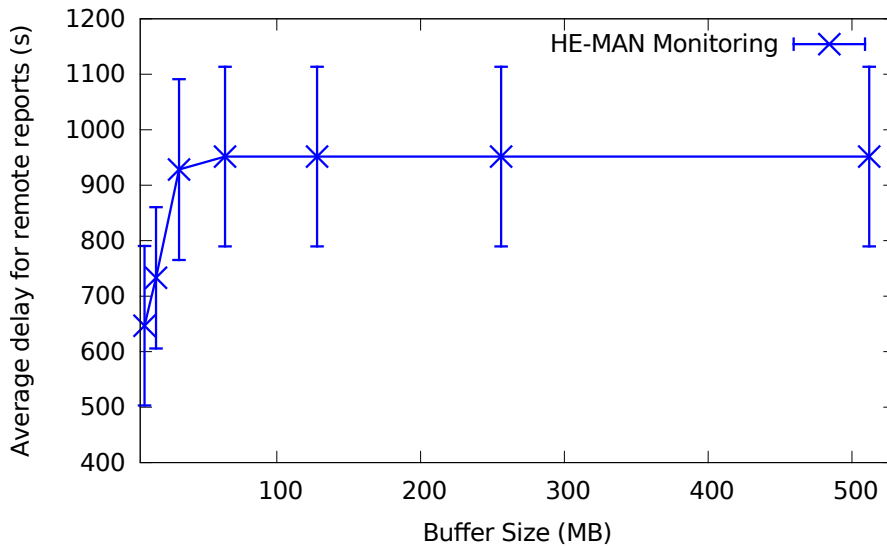


Figure 6.16. Remote reports average delay with different buffer sizes

communication, due to the fact that nodes in local groups are usually connected to each other. Therefore, even the smallest buffer sizes were able to store all the necessary messages long enough for them to be delivered to their destination.

Figure 6.18 shows the delivery ratio for remote reports sent to the VDTN's TLM by the MLMs. When the buffer size is between 64MB and 512MB the average delivery ratio remains approximately constant, around 12.72%. However, the average delivery ratio starts to fall when the buffer size ranges from 32MB down to 8MB, getting as

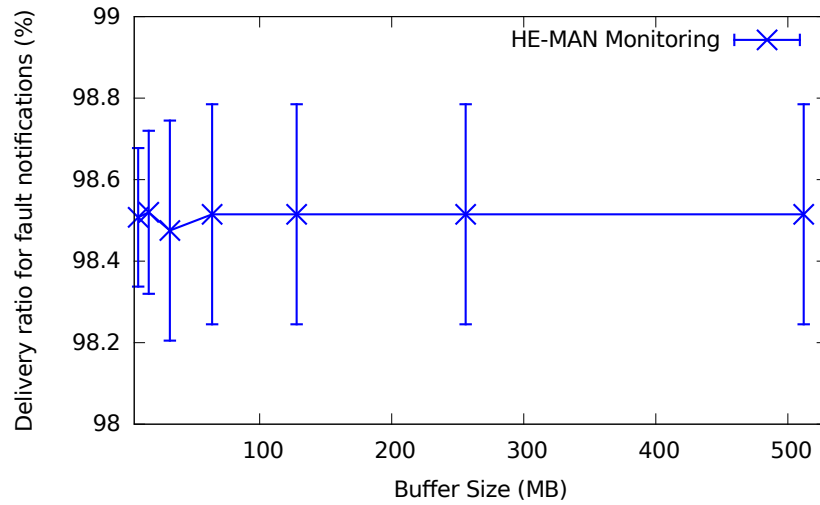


Figure 6.17. Fault notification delivery ratio with different buffer sizes

low as 6.9%. This behavior shows that, in the simulated scenario, messages start to be discarded due to the nodes' buffer being full if the buffer size is 32MB or less.

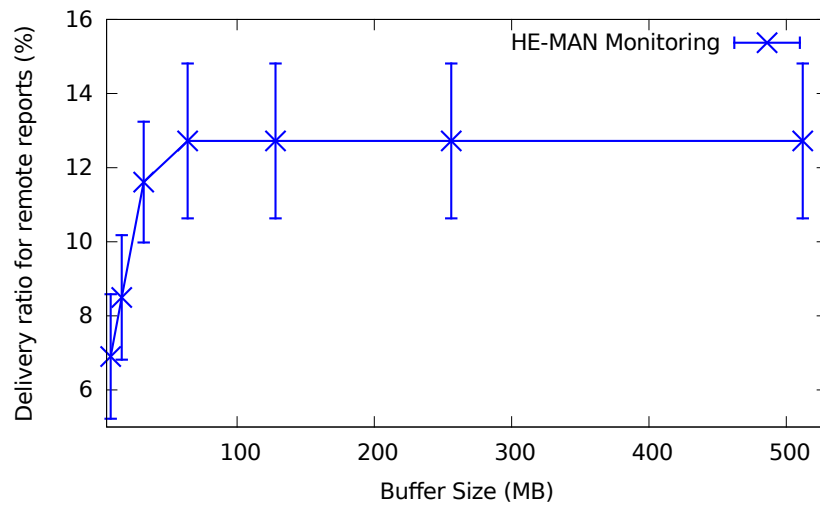


Figure 6.18. Remote reports delivery ratio with different buffer sizes

The results obtained from the experiments with different buffer sizes showed that, in the densest scenario considered in the experiments, a buffer size equal or above 64MB is enough to prevent the discard of messages due to a full buffer. As expected, limited buffers tend to affect only remote communication, as local communication does not force messages to spend long periods of time stored in buffers due to the constant availability of connections.

6.2.2 Performance analysis of HE-MAN Monitoring

In this subsection, the performance of the HE-MAN monitoring support is compared to the performance of the Simple VDTN monitoring. At first, the only routing algorithm that is considered is the Epidemic with Oracle, which provides the best performance routing-wise for both monitoring approaches. The objective of the analyses in this subsection is to highlight the main differences between the studied approaches.

Figure 6.19 shows the notification delay values obtained with the Collision Avoidance Application. In this scenario, a monitoring notification message takes from 15 minutes up to 20 minutes in average to reach its destination in the Simple VDTN Monitoring scenario. On the other hand, the average notification delay is between 0 and 0.05 seconds in the monitoring solution proposed with the HE-MAN's architecture. This is an expected behavior since the communication occurs within a connected group due to the hierarchical organization of the network.

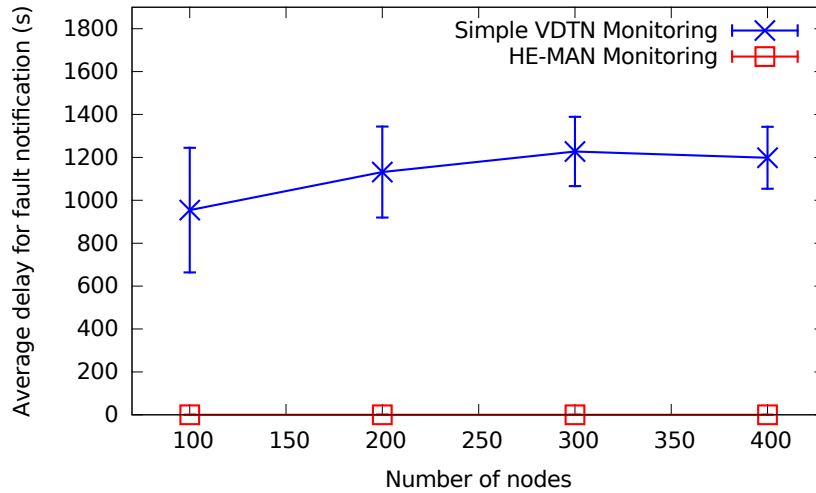


Figure 6.19. Fault notification average delay

Figure 6.20 presents the notification delay values achieved with the Event Logging application. A remote report generated within this application takes from 13 minutes to 15 minutes on average to reach its destination in the Simple VDTN Monitoring solution. The notification delay increases as the VDTN gets denser, as one would expect since denser networks provide more opportunities for messages getting to their final destination using more hops, which results in longer delays and higher delivery ratios. In the HE-MAN's monitoring solution this delay is between 11% and 16% higher than the ones observed in the Simple VDTN Monitoring scenario. This difference happens

because of the aggregation process that occurs in the MLMs, which reduces the number of remote report messages in the VDTN.

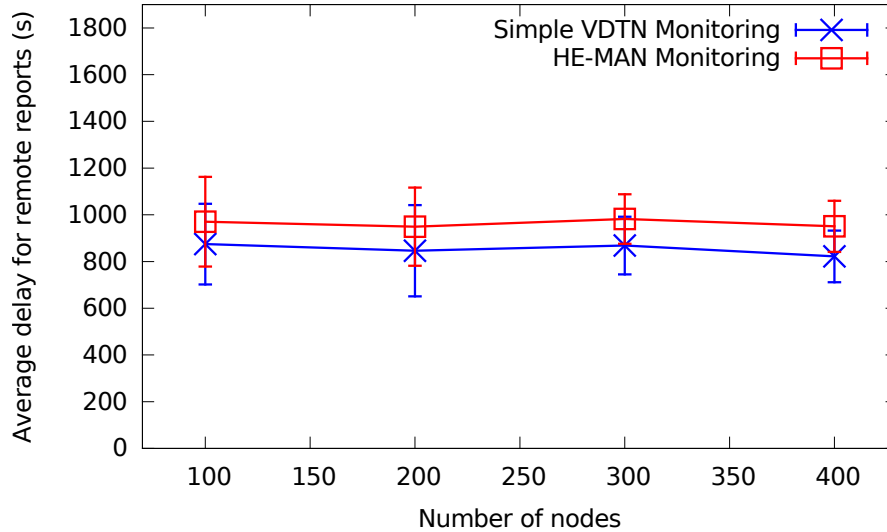


Figure 6.20. Remote reports average delay

The message delivery ratio was the second metric studied in the simulations with the VDTN monitoring solutions. Figure 6.21 shows the delivery ratio obtained with the local communication of the Collision Avoidance applications. The delivery ratio in the HE-MAN's monitoring solution was always superior to 98.5%, while in the Simple Monitoring Solution the delivery ratio was always inferior to 40%. The increase in the density of the VDTN benefits the fault notification delivery ratio of the Simple Monitoring solution due to higher availability of forwarding opportunities.

As for the delivery ratio achieved by the Event Logging application, shown in Figure 6.22, the observed average in the Simple VDTN Monitoring solution was always inferior to 18%. With the HE-MAN's monitoring solution, the average delivery ratio remained inferior to 15%. Again, the inferior delivery rate observed in the HE-MAN's approach can be explained by the aggregation of reports in the MLMs.

6.2.3 Analysis of the impact of DTN routing algorithms over VDTN monitoring

This subsection presents and analyzes the performance of HE-MAN's support to network monitoring, in comparison with the Simple VDTN Monitoring. Here, other DTN

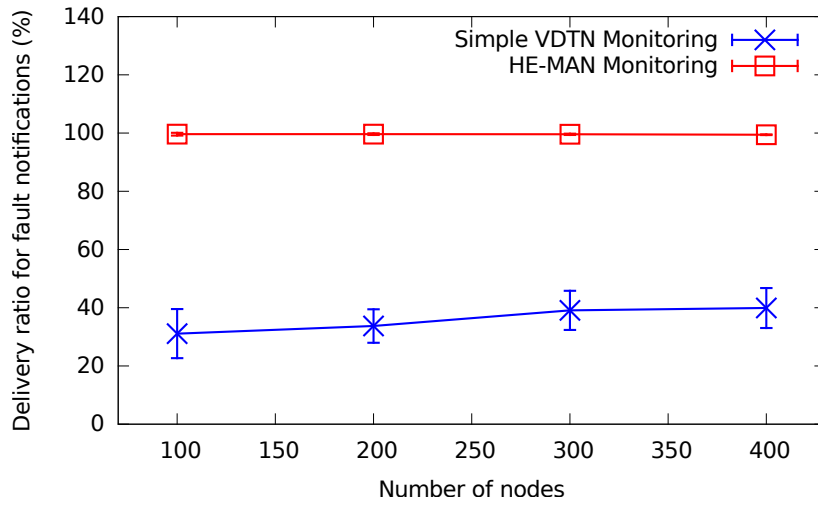


Figure 6.21. Fault notification delivery

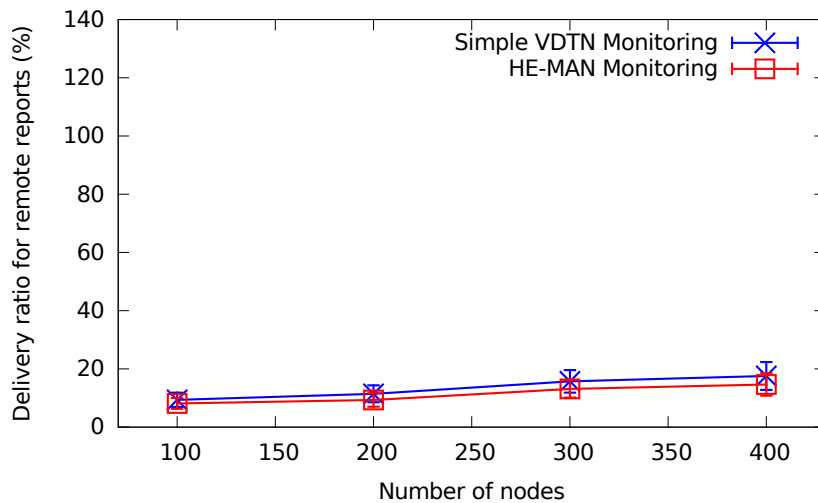


Figure 6.22. Remote reports delivery ratio

routing algorithms are taken into account, in order to assess the impact that routing has on the performance of the simulated monitoring schemes.

The notification delays obtained with the Collision Avoidance Application are shown in Figure 6.23. The spaces where bars did not appear mean that the respective value is zero or close to zero. It is easy to identify the significant difference between HE-MAN's hierarchical solution and the Simple VDTN Solution. Although there are some differences in the averages obtained through the used routing algorithms, these differences are subtle. The differences are more clear in the sparser simulated scenarios

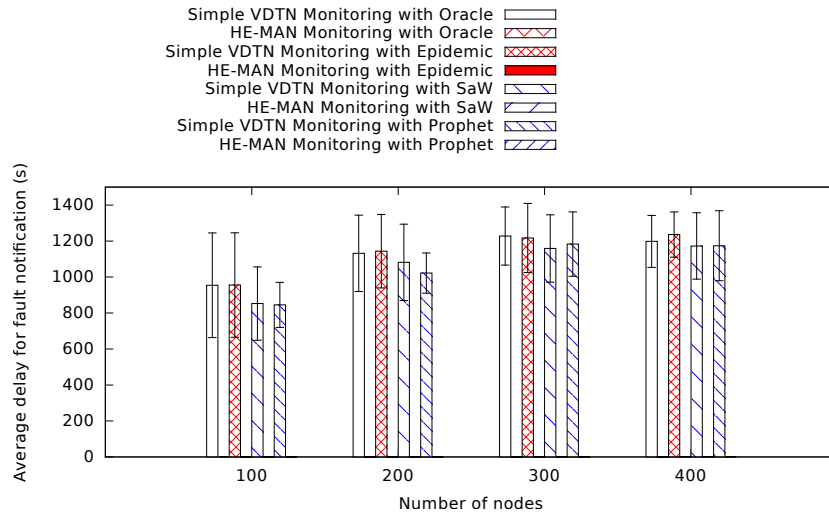


Figure 6.23. Fault notification average delay

(i.e., with 100 and 200 nodes).

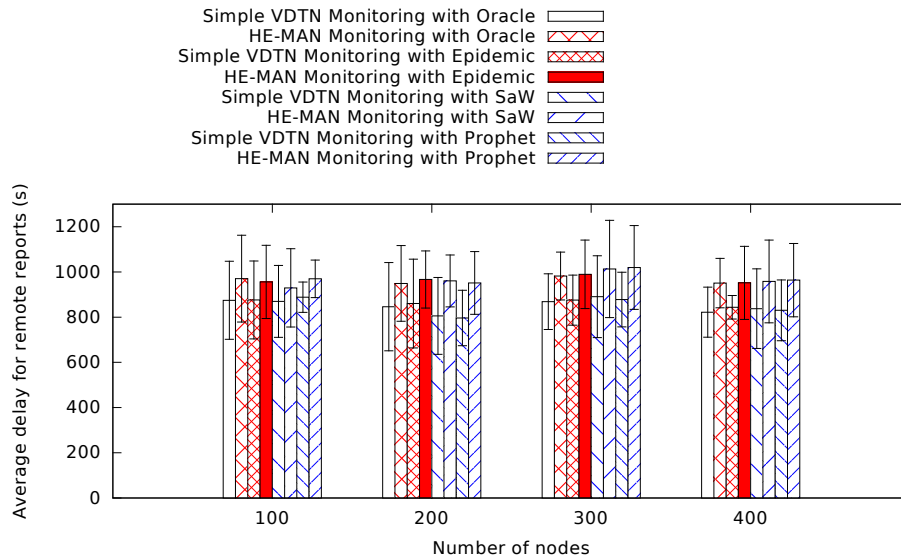


Figure 6.24. Remote reports average delay

Figure 6.24 presents the notification delay registered through different routing techniques with the Event Logging application. Once more, the difference between HE-MAN's Hierarchical Solution and the Simple VDTN Monitoring are easily spotted, with HE-MAN presenting higher delivery delay. The routing algorithms had little impact over the performance of the evaluated monitoring approaches.

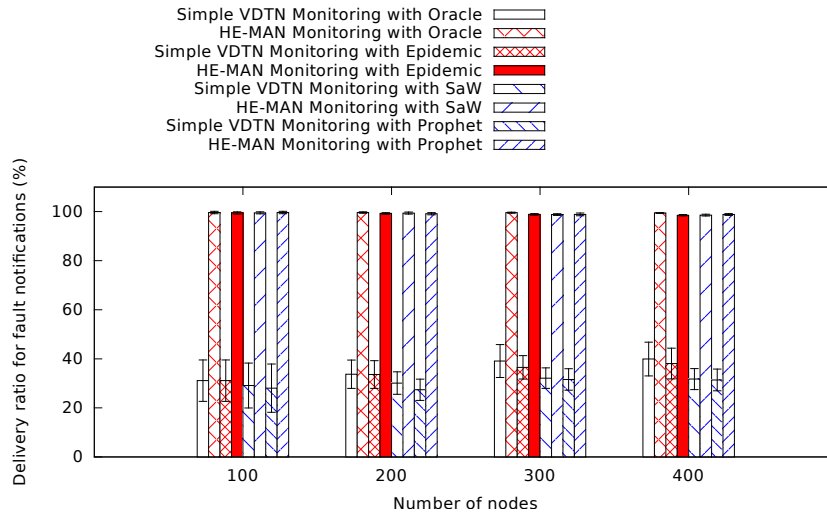


Figure 6.25. Fault notification delivery

Figure 6.25 shows the delivery ratio obtained with the local communication of the Collision Avoidance applications. For all tested DTN routing algorithms, there is a clear difference between the HE-MAN's Hierarchical Monitoring and the Simple VDTN Monitoring, with HE-MAN being able to deliver near 100% of the local notifications.

Finally, Figure 6.26 shows that the average delivery ratio of Remote Reports in the Simple VDTN Monitoring solution was consistently higher than the HE-MAN's Hierarchical Monitoring. Nonetheless, different routing algorithms had limited impact on the results. The difference between different routing techniques tends to become more perceptible as the VDTN gets denser.

The results presented in this appendix showed that different DTN routing algorithms have small impact on the overall performance of the studied monitoring approaches. Because of this, the monitoring scheme provided by the HE-MAN architecture should not impose limits to the choices of DTN routing techniques applicable to VDTNs.

6.3 Configuration solution evaluation

The support for configuration operations provided by the HE-MAN's architecture is divided into two categories: local configuration and remote configuration, as presented in Chapter 4. The communication pattern of local configuration is the same of HE-MAN's local monitoring, which involves messages doing round-trips between a cluster's MLM and its agents, using local communication. Since the performance of local com-

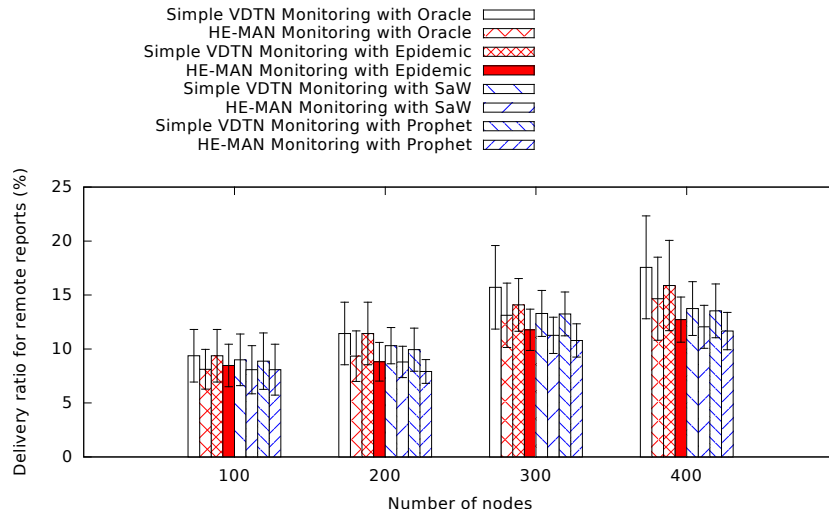


Figure 6.26. Remote reports delivery ratio

munication was already evaluated in the last section, in this section only the remote configuration support will be studied.

The first metric is the delivery delay, which represents how fast a configuration message can reach its destination. Figure 6.27 shows that the delivery delay tends to increase as the network becomes denser. The explanation to this is the increase of forwarding opportunities that comes with a denser network - a higher number of forwards frequently results in a higher communication delay. The delay was approximately 986 seconds (approximately 16 minutes) in the scenario with 100 nodes, which was increased to approximately 1843 seconds (approximately 30 minutes) in the scenario with 400 nodes. These messages that are delivered after a high number of forwarding have a high associated delay, which results in the total average delivery delay being increased.

The second metric is the delivery ratio of configuration messages, aiming to assess the probability of a configuration being actually delivered to its destination. Figure 6.28 shows that the delivery ratio of remote configuration messages can be as low as 10.25% in a VDTN with 100 nodes, and as high as 30.5% in a VDTN with 400 nodes. The increase of the delivery ratio in denser VDTN is expected due to the increase of forwarding opportunities for the configuration messages.

It is important to highlight that the achieved results in the experiments with HE-MAN's remote configuration were obtained with the usage of the Epidemic with Oracle routing protocol, which provides the highest possible delivery ratio and infinite buffer size, with each message being delivered as soon as theoretically possible. This

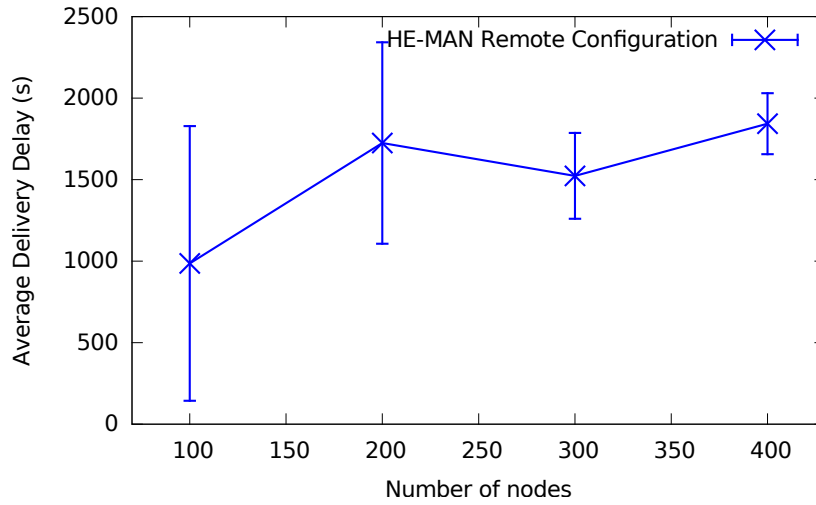


Figure 6.27. Delivery delay for configuration messages

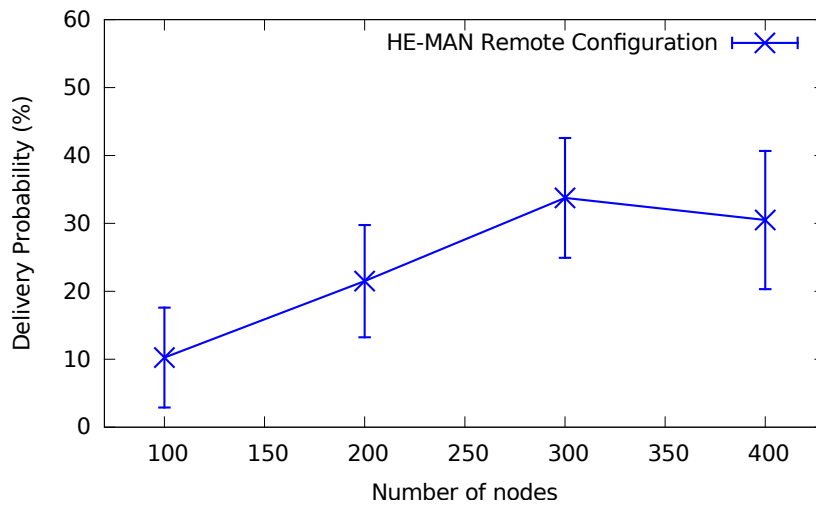


Figure 6.28. Delivery probability for configuration messages

fact emphasizes the existence of a hard limit for remote configuration viable. That also reinforces the necessity for local configuration techniques whenever possible, leaving remote configuration as a last resource.

6.4 Chapter Conclusions

In this chapter the results obtained from the simulations were presented and analyzed. The simulations with the algorithms revealed that the HE-MAN's clustering approach

was able to form clusters that are more inclusive to vehicular nodes in comparison to a similar clustering algorithm used as the baseline. The increase of the time nodes spent connected to clusters in HE-MAN was achieved without excessively sacrificing the stability of said clusters. The monitoring techniques designed for the HE-MAN architecture take advantage of the hierarchical organization of the VDTN to drastically reduce the communication delay and increase the message delivery ratio, which then enables the usage of near real-time monitoring techniques within the clusters. As for the HE-MAN's VDTN remote configuration approach, simulations pointed out that while this approach is possible, it has significant delays, as well as low delivery ratio, especially in sparse networks. Therefore, remote configuration should be used only as a last resort, giving preference to local configuration.

The next chapter presents the main conclusions drawn from this thesis, summarizing the lessons learnt, the achieved results and the limitations of this work. Finally, future directions of this research are presented and discussed.

Chapter 7

Conclusions and Future Work

VDTN is a research field that continues to attract attention from both the academia and the industry, with new solutions being designed every year in order to make this type of network more and more viable. However, even in face of the developments experienced by the VDTN area, only few works try to solve the network management problems that are characteristic to these networks, as shown in Chapter 3 of this thesis. The small amount of research on network management techniques for VDTNs emphasizes the importance of this thesis.

Most of the management requirements that apply to regular DTN also apply to VDTN, such as the long delays for a network manager to receive notifications from remote nodes, or the possibility of a configuration message arriving at its destination too late to be effective. Moreover, VDTNs impose challenges to management systems that are specific of this type of network. For instance, vehicular safety applications are very sensitive to delays, and this characteristic affects the requirements of a VDTN management system, since failing to effectively detect faults in safety systems can result in road accidents. Considering that frequent link disruptions and long delivery delays are unavoidable when it comes to remote communication in VDTNs, one of the conclusions of this research is the necessity of the management system to be aware of the current network topology at any given time, in order to be able to take advantage of local communication. This strategy is possibly the only way to a VDTN management system be able to perform near-real-time monitoring and configuration, although limited only to connected portions of the network.

This thesis proposed an architecture designed for establishing a hierarchical management of the network. The proposed architecture, called HE-MAN, provides means for connected groups detection and clusters formation, establishing a Mid-Level Manager (MLM) for each connected group of vehicular nodes. The group detection and

MLM election processes are made possible through a new clustering algorithm that was designed specifically for the HE-MAN architecture. Once the management topology is hierarchically organized, our proposed techniques for monitoring and configuration take advantage of local communication for optimizing the management tasks, aiming to improve the overall responsiveness and reliability of the management systems.

Simulations showed that our proposed clustering algorithm can organize connected groups successfully, having as one of the main objectives the maximization of the amount of time a node spends connected to VDTN clusters. By doing so, nodes can be locally monitored and configured by MLMs for a maximized amount of time, making these management tasks more reliable and efficient. Simulation results pointed that HE-MAN's clustering algorithm successfully improved the total time a vehicle spends connected to the clusters without completely sacrificing the stability of the clusters. It was also demonstrated in the simulations that the HE-MAN architecture have made possible, within the clusters, delivery ratios and notification delays compatible with the ones found in connected networks. Such results allow the usage of near-real-time monitoring techniques inside connected groups, for the monitoring of devices and software that are delay-sensitive (e.g., road safety systems). Also, MLMs were able to gather monitoring data from the agents in their respective clusters for forwarding it to the TLM, achieving delivery ratio and notification delay only slightly inferior to the considered baseline, which consisted of every agent sending their monitoring data directly to the central manager of the VDTN. The monitoring data aggregation that occurs in the MLMs help reducing the amount of bundles crossing the VDTN towards the TLM. Finally, the proposed remote configuration scheme allows the TLM to issue commands to remote nodes of the VDTN while having the means for verifying whether the situation of the node upon the reception of the new configuration remains consistent to the knowledge the TLM has about it or not.

In summary, the main contributions of this research work are:

1. a review and analysis of the literature related to VDTN management (Objective 1);
2. a formalization of the problem and goals of VDTN management (Objective 2);
3. the design of an architecture suitable for the management of VDTNs (Objective 3);
4. the design of monitoring algorithms suitable for local and remote communication (Objective 4);

5. the design of configuration algorithms suitable for local and remote communication (Objective 5);
6. qualitative and quantitative evaluations of the algorithms proposed for VDTN management (Objective 6)

7.1 Limitations

There are limitations regarding the research that need consideration. First of all, this thesis investigated Fault Management and Configuration Management, which leaves plenty of room for the development of a more complete architecture for VDTN management that is able to explore other management dimensions (e.g., Accounting Management, Performance Management, and Security Management).

Moreover, many of the scenarios used in this thesis as base of comparison are hypothetical, which reduces the level of realism of the evaluation. However, this happens due to the lack of real data about operational VDTNs at the moment this thesis was written.

Finally, not all characteristics found in actual hardware devices and software implementations are covered by simulators, which needs to be accomplished outside of a simulation environment.

7.2 On the Applicability of the Proposals on Different Scenarios

All the contributions provided by this thesis took into consideration the VDTN scenario. However, there are other scenarios where such contributions may be applicable. Since some dominant characteristics of VDTNs are high mobility of nodes, existence of dense and sparse regions in the network, and the opportunistic nature of the contacts, networks sharing these same characteristics may be candidates for the re-use of the results presented in this thesis.

Some scenarios where this thesis could be applied are the following:

- **Emergency Networks:** this type of network shares some important characteristics with VDTNs, such as the natural occurrence of groups of nodes (usually as a result of first responder teams moving together), the existence of a central entity managing the devices and services deployed in the disaster mitigation zone, and

the opportunistic nature of contacts due to the somewhat unpredictable movement of the network's nodes [Calarco and Casoni, 2011];

- **Pocket Switched Networks (PSNs):** this type of network aims to enable network services for mobile users through their personal devices (e.g., mobile phones, laptops, PDAs, etc.) even when they are not in reach of Internet connectivity islands [Hui et al., 2005]. Moreover, contacts in PSNs are opportunistic, and the nodes are likely to form islands of connectivity (e.g., grouping resulting from people gathering at some point of interest). All these characteristics turn PSNs candidates for the network management solutions found in HE-MAN's architecture.

On the other hand, some examples of scenarios that are unlikely to be suitable for using the contributions presented in this thesis are the following:

- **VANETs (MANETs):** if the vehicular network being considered does not have network partition of its topology, making it behave as a conventional MANET, then problems like long notification delays in monitoring operations or configuration messages arriving too late at their destination would not exist, rendering the usage of the HE-MAN architecture pointless;
- **Deep Space Networks:** despite this type of network suffering from frequent link disconnections and long propagation delays, its nodes are typically travelling in an isolated manner. This behavior does not allow the formation of locally connected groups, which is one of the assumptions of HE-MAN.

7.3 Future Work

Considering the current limitations of the HE-MAN architecture, and the possibilities of extending the applicability of the solution proposed in this research to other networking scenarios, there is a number of future research that is being considered. First, since HE-MAN's current functionalities are focused on Fault Management and Configuration Management, the architecture could be expanded in order to encompass other dimensions, such as Accounting Management, Performance Management and Security Management. Second, the development of a prototype implementation of HE-MAN in a testbed would allow more realistic results on the performance of the architecture in a real-life environment and provide insights for improvements to HE-MAN. Finally, new studies concerning the usage of HE-MAN in delay-tolerant networking scenarios

other than VDTNs would be desirable, with the objective of making HE-MAN a more general solution.

7.4 Publications

This section lists the publications that were direct results of the doctorate research described in this thesis.

- Salvador, E. M., Macedo, D. F., Nogueira, J. M., Almeida, V. D. D., Granville, L. Z. (2016) Hierarchy-based Monitoring of Vehicular Delay-Tolerant Networks. *In Proceedings of the 13th IEEE Annual Consumer Communications and Networking Conference (CCNC 2016)*. Las Vegas, 2016.
- Salvador, E. M., Mota, V., Macedo, D. F., Nogueira, J. M., Nobre, J. C., Duarte, P. A. P. R., Granville, L. Z., Tarouco, L. M. R. (2014) Uma Avaliação de Abordagens de Distribuição para Gerenciamento de Redes Tolerantes a Atrasos e Desconexões. *In Anais do XV Workshop de Testes e Tolerância a Falhas (WTF 2014)*, Florianópolis, 2014.
- Salvador, E. M., Macedo, D. F., Nogueira, J. M. (2011a) Gerência Autônoma de Redes DTN. *In I Workshop on Autonomic Distributed Systems (WoSIDA 2011)*, Campo Grande, 2011.
- Salvador, E. M., Macedo, D. F., Nogueira, J. M., Nobre, J. C., Duarte, P. A. P. R., Granville, L. Z., Tarouco, L. M. R. (2011b) Work-in-progress: Towards an Architecture for Managing Delay-Tolerant Emergency Networks. *In Proceedings on the 3rd Extreme Conference on Communication (Extremecom 2011)*, Manaus, 2011.

Bibliography

- (2014). IEEE Guide for Wireless Access in Vehicular Environments (WAVE) - Architecture. *IEEE Std 1609.0-2013*, pages 1–78.
- Agência Nacional de Transportes Terrestres (2014). Automated Monitoring of the Operation of Interstate and International Land Passengers Transport Services - MONITRIIP (In Portuguese: ‘Monitoramento Automatizado da Operação dos Serviços de Transporte Rodoviário Interestadual e Internacional de Passageiros - MONITRIIP’. <http://www.antt.gov.br>.
- Almalag, M. S. and Weigle, M. C. (2010). Using traffic flow for cluster formation in vehicular ad-hoc networks. In *Local Computer Networks (LCN), 2010 IEEE 35th Conference on*, pages 631–636. ISSN 0742-1303.
- Bai, F., Elbatt, T., Hollan, G., Krishnan, H., and Sadekar, V. (2006). Towards characterizing and classifying communication-based automotive applications from a wireless networking perspective. In *Proceedings of IEEE Workshop on Automotive Networking and Applications (AutoNet)*.
- Bai, F. and Krishnan, H. (2006). Reliability Analysis of DSRC Wireless Communication for Vehicle Safety Applications. In *2006 IEEE Intelligent Transportation Systems Conference*, pages 355–362. ISSN 2153-0009.
- Barba, C., Mateos, M., Soto, P., Mezher, A., and Igartua, M. (2012). Smart city for VANETs using warning messages, traffic statistics and intelligent traffic lights. In *Intelligent Vehicles Symposium (IV), 2012 IEEE*, pages 902–907. ISSN 1931-0587.
- Birrane, E. and Cole, R. G. (2010). Management of Disruption-Tolerant Networks: A Systems Engineering Approach. In *SpaceOps Conference*.
- Birrane, E. and Ramachandran, V. (2014). Delay Tolerant Network Management Protocol. draft-irtf-dtnrg-dtnmp-01.

- Biswas, S., Tatchikou, R., and Dion, F. (2006). Vehicle-to-vehicle wireless communication protocols for enhancing highway traffic safety. *IEEE Communications Magazine*, 44(1):74–82. ISSN 0163-6804.
- Burleigh, S. (2007). Interplanetary Overlay Network: An Implementation of the DTN Bundle Protocol. In *Consumer Communications and Networking Conference, 2007. CCNC 2007. 4th IEEE*, pages 222–226.
- Calarco, G. and Casoni, M. (2011). Virtual Networks and Software Router approach for Wireless Emergency Networks Design. In *Proc. of IEEE 73rd Vehicular Technology Conference 2011*.
- Case, J. D., Fedor, M. L., and Schoffstal, J. D. (1990). Simple Network Management Protocol (SNMP). RFC 1157.
- Cerf, V., Burleigh, S., Hooke, A., Torgerson, L., Durst, R., Scott, K., Fall, K., and Weiss, H. (2007). Delay-Tolerant Networking Architecture. RFC 4838 (Informational).
- Cerroni, W., Moro, G., Pasolini, R., and Ramilli, M. (2015). Decentralized detection of network attacks through P2P data clustering of SNMP data. *Computers & Security*, 52:1–16. ISSN 01674048.
- Chaintreau, A., Hui, P., Crowcroft, J., Diot, C., Gass, R., and Scott, J. (2007). Impact of Human Mobility on Opportunistic Forwarding Algorithms. *IEEE Transactions on Mobile Computing*, 6(6):606–620. ISSN 1536-1233.
- Clemm, A. (2006). *Network Management Fundamentals*. Cisco Press. ISBN 1587201372.
- Conti, M. and Giordano, S. (2014). Mobile ad hoc networking: milestones, challenges, and new research directions. *IEEE Communications Magazine*, 52(1):85–96. ISSN 0163-6804.
- Crowcroft, J., Yoneki, E., Hui, P., and Henderson, T. (2008). Promoting Tolerance for Delay Tolerant Network Research. *SIGCOMM Comput. Commun. Rev.*, 38(5):63–68. ISSN 0146-4833.
- Daeinabi, A., Rahbar, A. G. P., and Khademzadeh, A. (2011). VWCA: An efficient clustering algorithm in vehicular ad hoc networks. *Journal of Network and Computer Applications*, 34(1):207–222. ISSN 1084-8045.

- Dias, J. A. F. F., Rodrigues, J. J. P. C., de Paz, J. F., and Corchado, J. M. (2016). MoM - a real time monitoring and management tool to improve the performance of Vehicular Delay Tolerant Networks. In *2016 Eighth International Conference on Ubiquitous and Future Networks (ICUFN)*, pages 1071–1076.
- Ding, J. (2009). *Advances in network management*. CRC press.
- Doering, M., Pögel, T., Pöttner, W.-B., and Wolf, L. (2010). A New Mobility Trace for Realistic Large-scale Simulation of Bus-based DTNs. In *Proceedings of the 5th ACM Workshop on Challenged Networks, CHANTS '10*, pages 71--74, New York, NY, USA. ACM.
- Eichler, S. (2007). Performance Evaluation of the IEEE 802.11p WAVE Communication Standard. In *Vehicular Technology Conference, 2007. VTC-2007 Fall. 2007 IEEE 66th*, pages 2199–2203. ISSN 1090-3038.
- Fall, K. (2003). A Delay-tolerant Network Architecture for Challenged Internets. In *Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, SIGCOMM '03*, pages 27--34, New York, NY, USA. ACM.
- FALL, K. and FARREL, S. (2008). DTN: An Architectural Retrospective. *IEEE Journal on Selected Areas in Communications*, 26(5):828–836.
- Fathian, M. and Jafarian-Moghaddam, A. R. (2015). New clustering algorithms for vehicular ad-hoc network in a highway communication environment. *Wireless Networks*, 21(8):2765--2780. ISSN 1572-8196.
- Ferreira, A. (2002). On models and algorithms for dynamic communication networks: The case for evolving graphs. In *Proceedings of 4e rencontres francophones sur les Aspects Algorithmiques des Télécommunications (ALGOTEL'2002)*, pages 155–161.
- Ferreira, B. F., Rodrigues, J. J., Dias, J. a. a., and Isento, J. a. N. (2014). Man4VDTN - A network management solution for vehicular delay-tolerant networks. *Computer Communications*, 39:3--10. ISSN 01403664.
- George, S. M., Zhou, W., Chenji, H., Won, M., Lee, Y. O., Pazarloglou, A., Stoleru, R., and Barooah, P. (2010). DistressNet: a wireless ad hoc and sensor network architecture for situation management in disaster response. *IEEE Communications Magazine*, 48(3):128--136.

- Goldszmidt, G. (1993). Distributed system management via elastic servers. In *Systems Management, 1993., Proceedings of the IEEE First International Workshop on*, pages 31–35.
- Goonewardene, R. T., Ali, F. H., and Stipidis, E. (2009). Robust mobility adaptive clustering scheme with support for geographic routing for vehicular ad hoc networks. *IET Intelligent Transport Systems*, 3(2):148–158. ISSN 1751-956X.
- Grasic, S. and Lindgren, A. (2014). Revisiting a remote village scenario and its DTN routing objective. *Computer Communications*, 48:133–140.
- Hartenstein, H. and Laberteaux, K. (2008). A tutorial survey on vehicular ad hoc networks. *Communications Magazine, IEEE*, 46(6):164–171. ISSN 0163-6804.
- Hassanabadi, B., Shea, C., Zhang, L., and Valaee, S. (2014). Clustering in Vehicular Ad Hoc Networks using Affinity Propagation. *Ad Hoc Networks*, 13, Part B:535–548. ISSN 1570-8705.
- Horowitz, R. and Varaiya, P. (2000). Control design of an automated highway system. *Proceedings of the IEEE*, 88(7):913–925. ISSN 0018-9219.
- Hui, P., Chaintreau, A., Scott, J., Gass, R., Crowcroft, J., and Diot, C. (2005). Pocket Switched Networks and Human Mobility in Conference Environments. In *Proceedings of the 2005 ACM SIGCOMM Workshop on Delay-tolerant Networking*, WDTN '05, pages 244–251, New York, NY, USA. ACM.
- Hunter, J. S. (1986). The Exponentially Weighted Moving Average. *Journal of Quality Technology*, 18(4):203–210.
- Hussain, S. A. and Gurkan, D. (2011). Management and Plug and Play of Sensor Networks Using SNMP. *IEEE Transactions on Instrumentation and Measurement*, 60(5):1830–1837. ISSN 0018-9456.
- Ivansic, W. (2009). DTN Network Management Requirements. Internet Draft (Informational).
- Kahani, M. and Beadle, H. W. P. (1997). Decentralised Approaches for Network Management. *SIGCOMM Comput. Commun. Rev.*, 27(3):36–47. ISSN 0146-4833.
- Karagiannis, G., Altintas, O., Ekici, E., Heijenk, G. J., Jarupan, B., Lin, K., and Weil, T. (2011). Vehicular Networking: A Survey and Tutorial on Requirements, Architectures, Challenges, Standards and Solutions. *IEEE Communications Surveys & Tutorials*, 13(4):584–616.

- Kargl, F., Papadimitratos, P., Buttyan, L., Müter, M., Schoch, E., Wiedersheim, B., Thong, T. V., Calandriello, G., Held, A., Kung, A., and Hubaux, J. P. (2008). Secure vehicular communication systems: implementation, performance, and research challenges. *IEEE Communications Magazine*, 46(11):110–118. ISSN 0163-6804.
- Kato, S., Hiltunen, M., Joshi, K., and Schlichting, R. (2013). Enabling vehicular safety applications over LTE networks. In *2013 International Conference on Connected Vehicles and Expo (ICCVE)*, pages 747–752. ISSN 2378-1289.
- Kayis, O. and Acarman, T. (2007). Clustering Formation for Inter-Vehicle Communication. In *Intelligent Transportation Systems Conference, 2007. ITSC 2007. IEEE*, pages 636–641.
- Keränen, A., Ott, J., and Kärkkäinen, T. (2009). The ONE Simulator for DTN Protocol Evaluation. In *2nd International Conference on Simulation Tools and Techniques*.
- Kuklinski, S. and Wolny, G. (2009). Density based clustering algorithm for VANETs. In *Testbeds and Research Infrastructures for the Development of Networks Communities and Workshops, 2009. TridentCom 2009. 5th International Conference on*, pages 1–6.
- Lindgren, A., Doria, A., and Schelén, O. (2003). Probabilistic Routing in Intermittently Connected Networks. *SIGMOBILE Mob. Comput. Commun. Rev.*, 7(3):19–20. ISSN 1559-1662.
- Liu, K., Ng, J. K. Y., Lee, V. C. S., Son, S. H., and Stojmenovic, I. (2016). Cooperative Data Scheduling in Hybrid Vehicular Ad Hoc Networks: VANET as a Software Defined Network. *IEEE/ACM Transactions on Networking*, 24(3):1759–1773. ISSN 1063-6692.
- Maglaras, L. A. and Katsaros, D. (2012). Distributed clustering in vehicular networks. In *Wireless and Mobile Computing, Networking and Communications (WiMob), 2012 IEEE 8th International Conference on*, pages 593–599. ISSN 2160-4886.
- Martin-Flatin, J.-P., Znaty, S., and Habaux, J.-P. (1999). A Survey of Distributed Enterprise Network and Systems Management Paradigms. *Journal of Network and Systems Management*, 7(1):9–26. ISSN 1064-7570.
- Maslekar, N., Boussedjra, M., Mouzna, J., and Labiod, H. (2011). A stable clustering algorithm for efficiency applications in VANETs. In *Wireless Communications and Mobile Computing Conference (IWCMC), 2011 7th International*, pages 1188–1193.

- McCloghrie, K. and Rose, M. (1988). Management Information Base for Network Management of TCP/IP-based internets. RFC 1066 (Informational).
- Mohinisudhan, G., Bhosale, S. K., and Chaudhari, B. S. (2006). Reliable On-board and Remote Vehicular Network Management for Hybrid Automobiles. In *Electric and Hybrid Vehicles, 2006. ICEHV '06. IEEE Conference on*, pages 1–4.
- Morales, M. M. C., Hong, C. S., and Bang, Y. C. (2011). An Adaptable Mobility-Aware Clustering Algorithm in vehicular networks. In *Network Operations and Management Symposium (APNOMS), 2011 13th Asia-Pacific*, pages 1–6.
- National Highway Traffic Safety Administration (2014). Federal Motor Vehicle Safety Standards: Vehicle-to-Vehicle (V2V) Communications. http://www.nhtsa.gov/staticfiles/rulemaking/pdf/V2V/V2V-ANPRM_081514.pdf.
- Nelson, S. C. and Kravets, R. (2010). Achieving Anycast in DTNs by Enhancing Existing Unicast Protocols. In *Proceedings of the 5th ACM Workshop on Challenged Networks*, CHANTS '10, pages 63–70, New York, NY, USA. ACM.
- Nobre, J., Duarte, P., Granville, L., Tarouco, L., and Bertinatto, F. (2014). On using P2P technology to enable opportunistic management in DTNs through statistical estimation. In *IEEE International Conference on Communications (ICC)*, pages 3124–3129.
- Papadimitratos, P., Fortelle, A. D. L., Evenssen, K., Brignolo, R., and Cosenza, S. (2009). Vehicular communication systems: Enabling technologies, applications, and future outlook on intelligent transportation. *IEEE Communications Magazine*, 47(11):84–95. ISSN 0163-6804.
- Papalambrou, A., Voyiatzis, A., Serpanos, D., and Soufrilas, P. (2011). Monitoring of a DTN2 network. In *Baltic Congress on Future Internet Communications (BCFIC Riga)*, pages 116–119.
- Peoples, C., Parr, G., Scotney, B., and Moore, A. (2010). Context-aware policy-based framework for self-management in delay-tolerant networks: A case study for deep space exploration. *IEEE Communications Magazine*, 48(7):102–109. ISSN 0163-6804.
- Pereira, P. R., Casaca, A., Rodrigues, J. J. P. C., Soares, V. N. G. J., Triay, J., and Cervello-Pastor, C. (2012). From Delay-Tolerant Networks to Vehicular Delay-Tolerant Networks. *IEEE Communications Surveys & Tutorials*, 14(4):1166–1182. ISSN 1553-877X.

- Pierce-Mayer, J. and Peinado, O. (2016). DTN Network Management. In *SpaceOps Conference*.
- Rawashdeh, Z. Y. and Mahmud, S. M. (2012). A novel algorithm to form stable clusters in vehicular ad hoc networks on highways. *EURASIP Journal on Wireless Communications and Networking*, 2012(1):1--13. ISSN 1687-1499.
- Roberts, S. W. (1959). Control chart tests based on geometric moving averages. *Technometrics*, 1:239--251.
- Sachin, K., Mishra, A., and Cole, R. G. (2010). Configuration Management for DTNs. In *IEEE Globecom Workshops*, pages 589--594.
- Schoenwaelder, J., Quittek, J., and Kappler, C. (2000). Building Distributed Management Applications with the IETF ScriptMIB. *IEEE Journal on Selected Areas in Communications*, 18(5):702--714. ISSN 0733-8716.
- Scott, K. and Burleigh, S. (2007). Bundle Protocol Specification. RFC 5050. [S.l.]: Internet Engineering Task Force, Network Working Group.
- Silva, C. M. and Meira, W. (2015). Managing infrastructure-based vehicular networks. In *2015 16th IEEE International Conference on Mobile Data Management*, volume 2, pages 19--22. IEEE.
- Soares, V., Rodrigues, J., Dias, J., and Isento, J. (2012). Performance analysis of routing protocols for vehicular delay-tolerant networks. In *Software, Telecommunications and Computer Networks (SoftCOM), 2012 20th International Conference on*, pages 1--5.
- Souza, E., Nikolaidis, I., and Gburzynski, P. (2010). "a new aggregate local mobility (alm) clustering algorithm for vanets". In *Communications (ICC), 2010 IEEE International Conference on*, pages 1--5. ISSN 1550-3607.
- Spyropoulos, T., Psounis, K., and Raghavendra, C. S. (2005). Spray and Wait: An Efficient Routing Scheme for Intermittently Connected Mobile Networks. In *Proceedings of the 2005 ACM SIGCOMM Workshop on Delay-tolerant Networking*, WDTN '05, pages 252--259, New York, NY, USA. ACM.
- Stallings, W. (1998). *SNMP, SNMPV2, Snmpv3, and RMON 1 and 2*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 3rd edition. ISBN 0201485346.

- Strassner, J. (2003). *Policy-based network management: solutions for the next generation*. Morgan Kaufmann.
- Su, H. and Zhang, X. (2007). Clustering-Based Multichannel MAC Protocols for QoS Provisionings Over Vehicular Ad Hoc Networks. *IEEE Transactions on Vehicular Technology*, 56(6):3309–3323. ISSN 0018-9545.
- Torgerson, J. L. (2014). Network Monitor and Control of Disruption-Tolerant Networks. In *SpaceOps Conference*.
- Uzcategui, R. A., Sucre, A. J. D., and Acosta-Marum, G. (2009). Wave: A tutorial. *IEEE Communications Magazine*, 47(5):126–133. ISSN 0163-6804.
- Vahdat, A., Becker, D., et al. (2000). Epidemic routing for partially connected ad hoc networks.
- Vodopivec, S., Bester, J., and Kos, A. (2012). A survey on clustering algorithms for vehicular ad-hoc networks. In *Telecommunications and Signal Processing (TSP), 2012 35th International Conference on*, pages 52–56.
- Wahab, O. A., Otrók, H., and Mourad, A. (2013). VANET QoS-OLSR: QoS-based clustering protocol for Vehicular Ad hoc Networks. *Computer Communications*, 36(13):1422 – 1435. ISSN 0140-3664.
- Wang, F., Xu, Y., Zhang, H., Zhang, Y., and Zhu, L. (2015). 2FLIP: A Two-Factor Lightweight Privacy Preserving Authentication Scheme for VANET. *Vehicular Technology, IEEE Transactions on*, PP(99). ISSN 0018-9545.
- Wang, S.-S. and Lin, Y.-S. (2010). Performance evaluation of passive clustering based techniques for inter-vehicle communications. In *Wireless and Optical Communications Conference (WOCC), 2010 19th Annual*, pages 1–5.
- Wolny, G. (2008). Modified DMAC Clustering Algorithm for VANETs. In *Systems and Networks Communications, 2008. ICSNC '08. 3rd International Conference on*, pages 268–273.
- Xia, F., Liu, L., Li, J., Ahmed, A., Yang, L., and Ma, J. (2015). BEEINFO: Interest-Based Forwarding Using Artificial Bee Colony for Socially Aware Networking. *Vehicular Technology, IEEE Transactions on*, 64(3):1188–1200. ISSN 0018-9545.
- Xuan, B. B., Ferreira, A., and Jarry, A. (2003). Evolving graphs and least cost journeys in dynamic networks. In *WiOpt'03: Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks*, pages 10--pages.

- Zeadally, S., Hunt, R., Chen, Y.-S., Irwin, A., and Hassan, A. (2012). Vehicular ad hoc networks (VANETS): status, results, and challenges. *Telecommunication Systems*, 50(4):217--241. ISSN 1572-9451.
- Zhang, M. and Wolff, R. (2010). A Border Node Based Routing Protocol for Partially Connected Vehicular Ad Hoc Networks. *Journal of Communications*, 5(2).
- Zhang, Z., Boukerche, A., and Pazzi, R. (2011). 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*, MobiWac '11, pages 19--26, New York, NY, USA. ACM.