Universidade de Aveiro Departamento de Electrónica, Telecomunicações
2008 e Informática

**Rui Pedro
Ferreira da Costa**

Test and Measurement Environments For VANETs and MANETs

Cenários de Teste e Medidas para VANETs e MANETs

"Don't gain the world and lose your soul, wisdom is better than silver or gold..."

— Bob Marley

Universidade de Aveiro Departamento de Electrónica, Telecomunicações
2008 e Informática

Rui Pedro
Ferreira da Costa

Test and Measurement Environments For VANETs and MANETs

Cenários de Teste e Medidas para VANETs e MANETs

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requesitos necessários à obtenção do grau de Mestre em Engenharia de Computadores e Telemática, realizada sob a orientação científica de Susana Sargento e Rui Aguiar, Professores do Departamento de Electrónica, Telecomunicações e Informática da Universidade de Aveiro.

o júri / the jury

presidente / president	**Doutor José Luis Guimarães Oliveira** Professor Associado da Universidade de Aveiro (por delegação da Reitora da Universidade de Aveiro)
vogais / examiners committee	**Doutor Manuel Alberto Pereira Ricardo** Professor Associado do Departamento de Engenharia Electrotécnica e de Computadores da Faculdade de Engenharia da Universidade do Porto
	Doutora Susana Isabel Barreto de Miranda Sargento Professora Auxiliar Convidada da Universidade de Aveiro (orientador)
	Doutor Rui Luís Andrade Aguiar Professor Auxiliar da Universidade de Aveiro (co-orientador)

agradecimentos / acknowledgements

During the half-year I spent at NEC Network Labs in Heidelberg, I gained not only professional but also personal experience.
For this I would like to especially thank my supervisor at NEC Doctor Wenhui Zhang for his splendid orientation and support.
I would also like to extend my thanks to the IVC group members Dr. Andreas Festag and Eng. Roberto Baldessari for their direct or indirect contribution to this thesis. Also, I want to express my gratitude to all the friends and colleagues at NEC that helped me in one way or another during the time I spent there.
Also I would like to thank to Prof. Doctor Rui Aguiar and Prof. Doctor Susana Sargento for both their orientation and the chance to do my part of my Master Thesis in a business environment.
Finally I would like to appreciate the continuous support of my parents, brother and closest friends, during the development of this thesis and especially during my time abroad.

Resumo

A crescente necessidade por parte dos utilizadores em obterem acesso à Internet "em qualquer lugar e qualquer momento" tem incentivado investigação e desenvolvimento de abordagens capazes de resolver esta questão. Um dos maiores obstáculos em fornecer uma solução de acesso ubíquo à Internet tem sido a gestão de mobilidade.

Nesta Tese de Mestrado ir-se-á implementar uma das soluções correntemente em desenvolvimento, integrando-a num ambiente MANET. A plataforma resultante pode ser posteriormente avaliada e até retiradas medidas de desempenho, podendo-se ainda tirar conclusões importantes sobre como um ambiente MANET se comporta numa plataforma de mobilidade global e acesso ubíquo.

Na área de Sistemas de Transporte Inteligentes, tópicos como eficiência de tráfego e segurança dos utilizadores têm-se mostrado muito populares e deram início a pesquisa extensiva em Redes Veículares *Ad-Hoc* (*VANETs*). Métodos tradicionais para investigação e desenvolvimento como testes com protótipos ou simulação computacional têm sido largamente usados. No entanto, os testes com protótipos são usualmente muito caros e a simulação computacional tem falta de precisão em ambientes sem fios.

Esta Tese de Mestrado tem também por objectivo construir uma solução híbrida que combine os métodos de emulação e simulação. A solução proposta será implementada num *testbed* para *VANETs*. O *testbed* resultante irá permitir que multiplas instâncias de programas de *routing* reais possam ser executadas sobre um ambiente simulado computacionalmente. Assim poderão ser também retiradas elacções sobre o seu desempenho em características como o consumo de recursos e escalabilidade.

Abstract

The growing need from users to have internet access “whenever and wherever” has driven research to devise several approaches to cope with this issue. One of the greatest challenges in providing ubiquitous internet access has been the management of mobility.
In this Master Thesis a solution currently under development, will be implemented integrating a MANET environment. The resulting testbed can later be evaluated and it's performance measured, drawing important conclusions about how a MANET environment behaves in a global mobility and ubiquitous access framework.
In the area of Intelligent Transportation System traffic efficiency and safety for users have become very popular topics and have triggered extensive research in Vehicular Ad-Hoc Networks (VANETs). Traditional methods for reaserch and development like field testing and simulation have been used. But field testing is usually very expensive expensive and simulation lacks accuracy in wireless environments.
This Master Thesis also aims to provide a hybrid solution that combines the simulation and emulation methods. The proposed solution is implemented in a testbed for VANETs. The resulting testbed would allow multiple real routing instances to run simultaneously on a simulated environment. And to provide performance measures such as resource consumption and scalability.

Table of Contents

List of Figures

List of Tables

List of Acronyms

A4C Authentication, Authorization, Accounting, Auditing an Charging

AODV Ad-Hoc On-Demand Distance Vector protocol

AP Access Point

API Application Programming Interface

AR Access Router

ARP Address Resolution Protocol

AU Application Unit

cAR Current Access Router

C2C-CC Car 2 Car Communication Consortium

cGW Current Gateway

CN Correspondent Node

CoA Care-of-Address

Daidalos Designing Advanced network Interfaces for the Delivery and Administration of Location independent, Optimised personal Services

Daidalos I Daidalos project phase I

Daidalos II Daidalos project phase II

DARPA Defense Advanced Research Projects Agency

DHCP Dynamic Host Configuration Protocol

DVB Digital Video Broadcasting

FIFO First-In First-Out stack policy

GMD Global Mobility Domain

GMM	Global Mobility Management
GMP	Global Mobility Protocol
GPS	Global Positioning System
GRREP	Gateway Route Reply
GW	Gateway
HA	Home Agent
HAWAII	Handoff-Aware Wireless Access Internet Infrastructure
IIS	Intelligent Interface Selection
IMUNES	Integrated Multiprotocol Network Emulator / Simulator
L2	Layer 2 - Link Layer
LMA	Local Mobility Anchor
LMD	Local Mobility Domain
LMM	Local Mobility Management
LMP	Local Mobility Protocol
MAC	Media Access Control
MAG	Mobility Anchor Gateway
MANET	Mobile Ad-Hoc Network
MIH	Media Independent Handover
MIHF	Media Independent Handover Function
MIP-MANET	Mobile IP for Mobile Ad-Hoc Networks
MIPv6	Mobile IPv6
MIWU	MIP-MANET Interworking Unit
MMARP	Multicast MANET Routing Protocol
MN	Mobile Node
MT	Mobile Terminal
MTC	Mobile Terminal Controller
MW	MANET Wrapper

nAR New/Next Access Router

NCTUns National Chiao Tung University network simulator

NetLMM Network-Based Local Mobility Management

nGW New/Next Gateway

NoW Network-on-Wheels project

NoWd Network-on-Wheels demonstrator

NS-2 Network Simulator 2

OBU On-Board Unit

OLSR Optimized Link State Routing protocol

OLSRd Optimized Link State Routing protocol Daemon

PAA PANA Client

PAC PANA Agent

PANA Protocol for carrying Authentication for Network Access

PBR Position-Based Routing

PoA Point-of.Access

QoS Quality-of-Service

RA Routing Advertisement

RAL Radio Access Layer

RADVD Router Advertisement Deamon

RPSC Routing Protocol Selector and Controller

RSU Road Side Unit

SSID Service Set Identifier

TCP Transmission Control Protocol

UDP User Datagram Protocol

ULA Unique Local IPv6 Unicast Address

UMTS Universal Mobile Telecommunications System

VANET Vehicular Ad-Hoc Network

VID	Virtual Identity
VIDAuth	Virtual Identity Authenticator
VIP	Virtual Interface Proxy
VoIP	Voice over IP
WiMAX	Worldwide Interoperability for Microwave Access
WLAN	Wireless Local Area Network

Chapter 1

Overview

1.1 Introduction

Wireless communications have greatly evolved in the last few decades. Mobility has posed a challenge for communications in several aspects.

Mobile Ad-Hoc Networks (MANETs) are nowadays one of the most interesting research fields in networking. This type of networks have very specific characteristcs. The great mobility of the nodes and consequent dynamic topology of this type of networks pose some unique and interesting challenges. Some of these challenges are for instance network routing, integration with infrastructured networks and mobility management.

Nowadays, regular mobile nodes can support different access network technologies. They can therefore use multiple links and different technologies to access the internet. With the help and support of an infrastructured network, MANETs can also belong to this multiplicity of access technologies and allow it's nodes to freely move between networks and moreover, use whichever access technology is better suited. From a mobility management point of view this introduces new relevant issues. In this thesis a solution for the seamless mobility of MANET nodes is proposed and will be addressed by the development of a real testbed environment.

Vehicular Ad-hoc Networks (VANETs) as a specific type of Mobile Networks also have inherited many of the challenging aspects of MANETs. Intensive research has been done to cope with these challenges and improve vehicular communications. Usually solutions are designed based on research using traditional methods like field testing and simulation. These methods however have drawbacks, for instance, field testing is expensive and simlation lacks accuracy especially in wireless environments. Therefore a hybrid approach method combining simulation with emulation will be used for research and development for VANETs, in this thesis.

1.2 Focus of This Thesis

The focus of this thesis is in fact divided in two parts. This is due to the fact that a part of the thesis was developed in a business environment in cooperation with `NEC Deutschland`, while the other was developed in a more academic environment in association with `Instituto de Telecomunicações - Aveiro`. Each part refers to a different project with different architectures and also different approaches. Regardless of the differences in origin or development environment, the goal of both parts resides in the development of testbeds in order to test and measure the performance of VANETs and MANETs.

The first part of this master thesis is related to the Network on Wheels (NoW) project [3]. One of the main goals of the NoW project is the development of a communication platform for VANETs. It is a German research project, which is supported by the Federal Ministry of Education and Research (BMBF).

In the project it is proposed an implementation of an experimental platform for performance evaluation in realistic environments. An application part of this project is the NoW demonstrator which supports the network layer.

In the first part of this master thesis, an integrated testbed for test and measurement in a VANET environment is designed and developed.

In order to test multiple simultaneous nodes in a VANET simulated environment, the use of the NoW demonstrator as a tool to provide Positioned-Based Routing is required. The testbed also includes another tool, a simulator/emulator that will provide the simulated environment and the simulated network elements. The integration of both tools in order to build the testbed is therefore the central point of the work. Furthermore the testbed is tested and measured in order to determine the framework's performance, scalability and limitations.

In the second part of this thesis, a different project is the base of the work presented. The Daidalos project [1], currently in phase II (therefore onwards refered as Daidalos II), is an EU Framework Programme 6 Integrated Project.

The objective of the project is to have seamless mobility integration between different types of networks and access technologies, such as WLAN, UMTS, DVB and MANET, among others, to provide voice, data, and multimedia services to the users, regardless of the underlying network and technology.

This project is being maintained and developed by several partners spread all over Europe. One of these partners is `Instituto de Telecomunicações - Aveiro`, which is responsible for the development and test of the MANET related architecture, software components and testbed implementation of the project.

In the scope of the Daidalos II project, the relevant work to be presented in this thesis is the development of a real testbed environment that can demonstrate the seamless mobility of mobile nodes between different types of access technologies. Furthermore, it focus in the mobility design between regular WLAN networks and MANETs with supporting infrastructure. This testbed will require the development of MANET specific software components, compliant with the project architecture. A valuable result of this work concerns possible performance measurements that can be therefore obtained from the testbed.

1.3 Structure of the Thesis

The document is structured as follows:

Chapter 2 introduces Mobile Ad-Hoc Networks (MANETs), Vehicular Ad-Hoc Networks (VANETs), and describes architectures, characteristics and applications. Routing in these networks is also introduced, with relevance to Position-Based Routing (PBR). It also makes a reference to mobility proposals for the integration of MANET in architectures that support multiple access technologies.

Chapter 3 addresses the work on the integration of the simulator and emulator tools and the building process of the hybrid testbed. It starts with the motivation for this part of the master thesis and the concepts of simulation and emulation, their advantages and drawbacks. In addition analysis is done in order to select a simulator/emulator tool to use in the testbed. It describes the the main challenge presented by the project in the scope of this master thesis. Also the Network-on-Wheels demonstrator, a routing application for vehicular ad-hoc networks is introduced. Furthermore it presents the design, implementation and performance measurement of the testbed in order to produce some conclusions about the overall implementation.

Chapter 4 presents the real implemented testbed that demonstrates seamless mobility between WLAN and MANET. It describes the Daidalos II MANET architecture, specifying the issues related to MANET integration in the Daidalos architecture. It also introduces the MANET software components implemented and their general flow of messages. Furthermore, the MANET testbed is presented in accordance with the defined test scenarios and finally test results are presented and conclusion are discussed.

Chapter 5 summarizes the work contribution of this thesis and presents some possible future work.

Chapter 2

Ad-Hoc Networks

In Chapter 1 a brief overview of the project and the report has been given. As in any research a proper study of the background technology and environment must be done in order to fully comprehend the matter in hands. In that sense Chapter 2 follows as a deeper insight to the technological environment of vehicular mobility with particular relevance in vehicular networks.

In **Section 2.1**, a deeper insight over the subject of Vehicular Ad Hoc Networks. How they appeared and evolved to what they are today refering it's strong points issues and applications.

In **Section 2.2**, Mobile routing types are briefly analysed in this section following a detailed description about the type of routing used in VANETs.

In **Section 2.3**, mobility is addressed in the scope of MANETs.

2.1 Ad-Hoc Networks

The history of communications has undergone great changes since ancient times. The establishing of communication has been a great challenge. Back then, communications needed to be reliable and effective through great distances, with the purpose of for instance propagating the warning of grave danger, such as a military invasion or a fire.

Methods were simple and vast, but mainly resumed to visual and sound references, such as smoke signs or the ringing of the town bell. Though these methods were relatively good for communication, they could only transmit signals from fixed points. The signals themselves were probably only a handfull of different variants, differing slightly from each other, and could easily be misunderstood. On the other hand, the use of animals, such as horses, camels (with riding men) or doves, was far more flexible in terms of messages and communication points. One good example is in the battlefield, where several riding messagers used to be the base of the communication system, crucial to coordinate offensive and defensive manouvers. The late arrival or death of one messenger could easily change the outcome of a battle.

Technology has evolved over the centuries and some inventions became crucial in today's means of communication. The last century saw good evolutionary steps on communications. Electrical pulses and radio waves made it possible to have reliable and flexible communication. The computer revolution was a major driven force for the success of wireless communications. It allows different types of data to be modulated into radio waves in very complex ways, and it also enhances the use of very complex algorithms to ensure the security and secrecy of the contents.

Some early initiatives were done with the intent of putting these concepts into practice. Some results of intense research were motivated by military purposes, and produced by military organisations. The United States Department of Defense implemented the well known DARPA network [22]. Other initiaves appeared by the simple need of communication as the ALOHA network in the Hawaiian islands [18]. Both of these networks innovated in using digital packet switching over wireless multi-hop connections.

The need for communication became stronger, and research is proved to be the greatest source of inovation pushing forward the development of improved technologies. Mainly driven by military research in the battlefield, new requirements foresee the development of a new type of network for communications were placed. Such requirements were: it should use wireless communication as a means to support mobility; it should not use infrastructured support; it should be simple to deploy; it should be highly robust and flexible; it should be self-organised; it should ensure the security of the communications. The foundation for Mobile Ad-Hoc Networks had been laid.

2.1.1 Mobile Ad-Hoc Networks

Mobile ad-hoc network is a type of mobile network, where devices can exchange information directly between themselves without the need of support from an infrastructure.

These type of networks have the property of being easily deployed and being self-organised, and do not need any type of previous configuration. Usually, they do not use any kind of infrastructure to support communication, thus rely on their wireless capabilities for communicating. In addition to compatibility, cooperation with infrastructured networks is also possible. Being very flexible in terms of topology due to their distributed approach, they are composed of one single type of component, the mobile node. The node itself is composed of a router with one or several interfaces [18]. The mobility of the nodes and their wireless communication abilities are the essential focus of these networks. These networks can be implemented in a single-hop approach, but could also be extended as multi-hop networks.

For a long time this concept was only used in military context due to the lack of comercial interest, until mobility appeared as a strong requirement in comercial communications. The boom of mobile phones has made a great impact on the world, and is paving the way for further development in mobile communications, to satisfy a growing market of mobile users.

Some applications of mobile ad-hoc networks generally include: communications where infrastructure for communications is usually unavailable, providing communication between heterogenous networks and devices, tactical battlefield communication and information sharing in meetings or classes.

Location and movement are implicit concepts in vehicles. A vehicle is a mobile device featuring several good features to provide mobile conectivity. It has a power source and it does not have great constraints on size or weight as opposed to other mobile devices, therefore it provides a better platform to develop on. These advantages are beneficial for the aplication of vehicular ad-hoc networks.

At the beginning, mobile ad-hoc technology was used by vehicles to provide communication between emergency vehicles in emergency scenarios [22]. Altough helpful, this was only one of the many applications of mobile ad-hoc networks. Other vehicular applications focused on avoiding road accidents.

Reducing vehicle collision fatalities and injuries is one of the main goals in vehicular ad-hoc networks research to avoid collisions and other types of road accidents.

2.1.2 Vehicular Ad-Hoc Networks (VANETs)

VANETs share most of the distinct characteristics of mobile ad-hoc networks, and they are considered as a specific type of mobile ad-hoc networks. Albeit similar, they distinguish themselves by the speed of a node's mobility, hence the very dynamic feature of it's topology. This also means a very low link time, and a high probability of losing communication. Also, they generally have patternized movement confined by the road. In some cases, the network topology can be very complex.

VANET Architecture

An overview of the VANET high-level arquitecture is shown in [Fig 2.1] [7]. As it is easily noticed, the architecture is divided into three domains: Infrastructure, Ad-Hoc and In-Vehicle. Also there are three very important elements: the Application Unit (AU), the On-Board Unit (OBU), and the Road Side Unit (RSU).

All the domains are clearly distinct in type of communications and in the physical area of action. Both the Infrastructure domain and the In-Vehicle domain use in majority wired connections between their own elements, but can also have some wireless connections. In comparison, in the Ad-Hoc domain the connections are strictly wireless.

The AUs are generally devices used by applications that use the OBUs as gateways to access other networks such as the Internet. They can be separated from the OBU but can also be co-located with it as an extension rather than part of the core network. The OBUs are devices that act as routers. They are part of the vehicle and assure it's communication with other vehicles's OBUs and road sided RSUs. The RSUs are static devices that are usually placed on the side of the road providing communication betweeen vehicles and the infrastructured network.

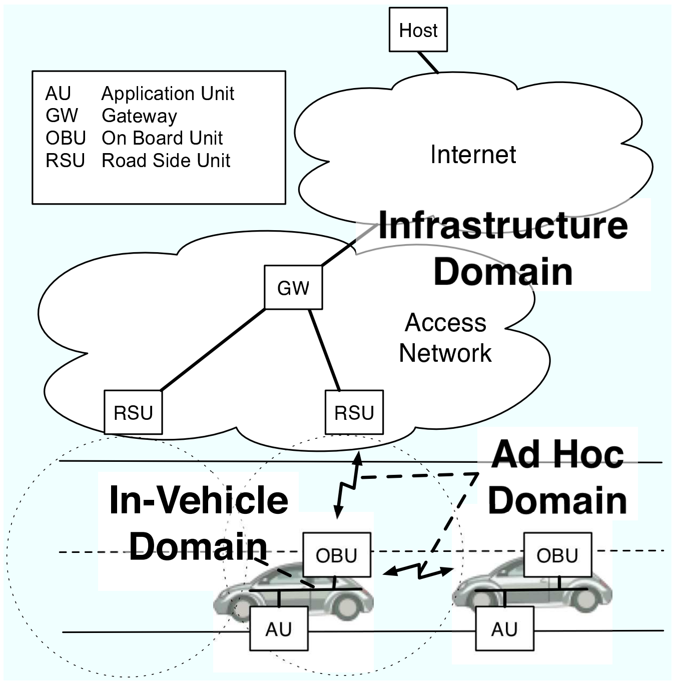

Figure 2.1: VANET Architecture

VANET Characteristics

For reference on this subsection see [18] [22].

- Network Size

  The number of nodes in a network variates accordingly to the density of nodes, in high density areas such as in large cities, the space between the nodes is small, and the number of neighbours is big. On the other hand, in more sparse areas the distance between nodes is large, and the number of nodes is small.

- Direct Connectivity

  The connectivity varies accordingly to the relative speed and direction betweeen nodes. If the relative speed is low and the heading is similar then connectivity should last relatively long (some minutes). In the cases that relative speed is high, or the heading isn't similar, then the connectivity is quite low (some seconds), or there might not even be a connection. Also a crucial factor, is the communication range of mobile nodes. Low ranges implicate less neigbours, and less link conectivity; high ranges implicate more power consumption, more neighbours and more conectivity.

- Network Topology

  Generaly, it's highly dynamic with the large majority of the nodes moving, which can cause a node's joining, leaving, rejoining a network.

- Medium Access Scheme

  The media is shared. The contention of nodes with shared bandwith and radio interference from neighbouring nodes may cause packet loss.

- Routing

  Routing is a challenge. Several algorithms and protocols exist which cope with the very dynamic topology. The constant changing of the topology means that there is a need to delete old routes, calculate new ones and update the ones still existing. All these must be done with low latency and little overhead.

- Self-Organisation

  This is one of the most distinctive features of an ad-hoc network, it resides on the nodes' advertisements, usually through *beacons*.

- Quality of Service

  Due to the distributed routing, shared media, and dynamic topology, it is difficult to maintain any constant or sometimes even temporary quality of service provisioning.

- Security

  With it's shared media access, these type of networks are much more vulnerable to attacks, such as eavesdropping, denial of service and impersonating attacks.

VANET Application Scenarios

For reference on this subsection see [9].

- Safety
  - Cooperative Collision Forwarding

    This is a preventive feature that allows the driver to be informed about a rear-end collision.
  - Pre-Crash Sensing/Warning

    This feature is engaged when the point of avoiding collision is not possible anymore, thus warning the driver to prepare for the crash while vehicle safety measures are taken to minimize occupant injury and vehicle damage.
  - Hazardous Location Vehicle-to-Vehicle Notification

    Notifications are generated when the vehicle crosses locations with dangerous driving conditions.

- Traffic Efficiency
  - Enhanced Route Guidance and Navigation
    The main goal is to propose alternative routes based on traffic, roadworks and other usefull information to help optimize the driver's route.
  - Green Light Optimal speed Advisory
    The vehicle is informed of a red light ahead and notifies the driver of the best speed to minimize fuel comsumption.
  - Vehicle-to-Vehicle Merging Assistance
    This application is focused on minimazing disruptions in the traffic flow, by exchange of information upon entering a road.
- Infotainment and Others
  - Internet Access In Vehicle
    The support of the infrastructured network is used to provide internet access to the vehicles.
  - Point of Interest Notification
    These notifications are used to promote local points of interest.
  - Remote Diagnostics
    This application aims to improve the vehicle's assistance service.

2.2 Position-Based Routing (PBR)

In mobile ad-hoc networks routing is a challenge. Due to it's very dynamical topology different approaches have been used. Many protocols have been implemented, and we can classify them as the following:

- Topology-Based [24]
  This type of approach produces routes using link information to actively route packets [17]. The protocols of this approach can generally be subdivided in [17]:
  - Proactive
    Proactive protocols are also known as *Table-Driven*, because they use updated routing tables through constant exchange of information with neighbouring nodes. Proactive protocols imply more mechanisms to be implemented but offer fresh and reliable information.
  - Reactive
    They are also known as *On-Demand* protocols. These protocols search the network for a route whenever it is required.
  - Hybrid
    As the name implies, these protocols aim to find the balance between the other two types by combining the best features provided by either of the preceding types.

- Position Based [17]
  Position routing relies on a node's knowledge of it's own position and other nodes's positions.

2.2.1 PBR Concept

In mobile ad-hoc networks, particularly in VANETs, movement is a key concept and the very dynamic topology makes it difficult to keep track of the current configuration of the network. While topology-based approaches use link information to maintain connection, position-based routing uses the node's position. A node is aware of it's position by using devices such as GPS. The node advertises it's position to allow neighbours to know where it is. The neighbours's advertisements allow the same role. These advertisements are known usually as network *beacons* and the procedure is called *beaconing.* To find a node that is not directly connected, a *location service* is provided. Routing then is done using algorithms to forward packets based on the nodes's position.

2.2.2 Position

Position Formats

There are various formats that can be used to define a position, but the geodesic and carthesian formats are more important to this project than the others. Due to it's very generic use, the Carthesian three dimensional coordinate format is one of the most used system to represent position. Slightly adapted it can be used to pinpoint any location on Earth, as can be seen on [Fig 2.2] [7].

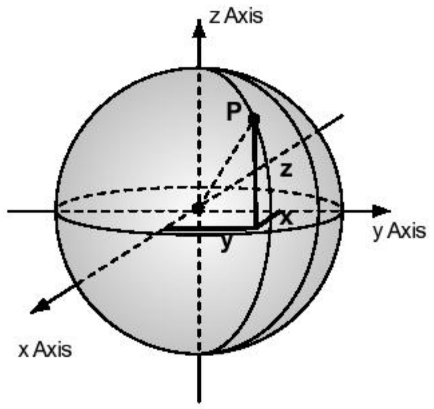

Figure 2.2: Carthesian Coordinate System

Another reference system that became popular in recent years is the Geodesic coordinate system. This system uses a two coordinate set to point a location on Earth's surface. Composed of *latitude* and *longitude*, this type of reference uses angles instead of distances while refering to a point as shown [Fig 2.3] [7].

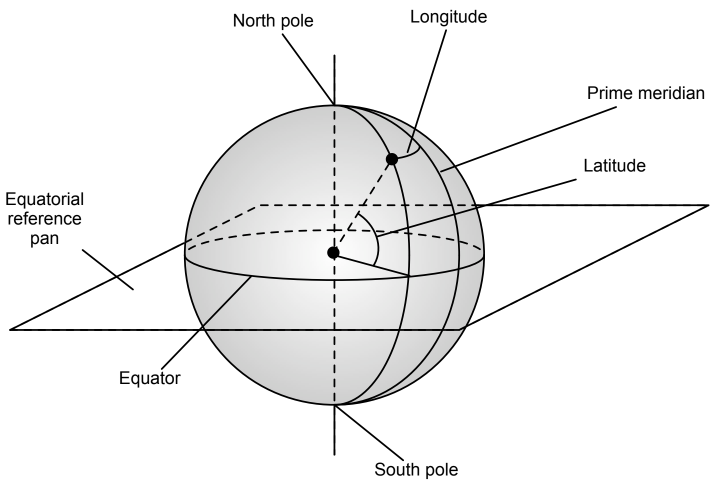

Figure 2.3: Geodesic Coordinate System

Latitude is the angle between the point and the equator line. Possible values are in the range -90 degrees to +90 degrees. *Latitude* is the angle between the point and the prime meridian line, also known as the *Greenwich* meridian. This meridian line goes from the North Pole to the South Pole while passing trough the city of *Greenwich*, England. Possible values are in the range of 0 to 180 degrees west of the prime meridian and 0 to 180 degrees east of the prime meridian.

Position Providing

Providing position to a device has become increasingly easier over the last few years. Due to developments in satelite communication it is now cheaper to obtain Global Positioning System (GPS) devices. The lowered prices of these devices generated a growing popularity making them today a good choice to obtain an accurate and affordable positioning provider. A GPS device could be attached to the mobile node through an appropriate interface periodically transmitting it's position. This type of feeding has already been tested proving it to be accurate and reliable [7].

2.2.3 Position-Based Forwarding Algorithms - Greedy Forwarding

The Greedy Forwarding Algorithm is a very simple algorithm used to forward packets. The Distance-Aware Greedy Forwarding Algorithm [16] goal is to minimize the number of hops. A good example of the Distance-Aware Greedy Forwarding Algorithm is shown[Fig 2.4] [10]:

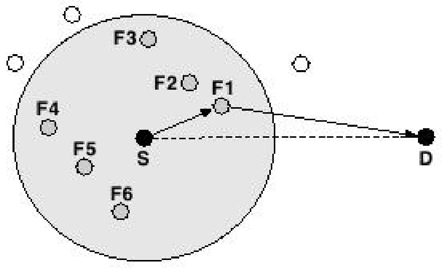

Figure 2.4: Distance-Aware Greedy Forwarding

As depicted the source node *S* is sending a packet to the destination node *D*. Minimizing the number of hops in position based routing is the same as maximizing hop distance. Therefore the algorithm will check for the neighbour node which is closer to the destination in an attempt to make the packet travel a longer distance. After finding a neighbour that complies with this condition the packet will be forwarded to it.

The Greedy Forwarding Algorithm has some drawbacks. When a node is forwarding a packet and there is no neighbour that satisfies the criteria. The packet then is dropped without reaching the destination.

2.2.4 PBR Protocol

Architecture and Component Structure

As specified by the C2C-CC [9], the PBR Communication System [7] supports several types of applications.

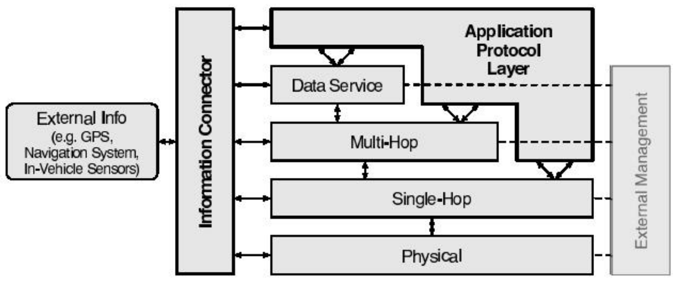
Application Protocol Layer
Data Service
Multi-Hop
Single-Hop
Physical
Information Connector
External Info (e.g. GPS, Navigation System, In-Vehicle Sensors)
External Management

Figure 2.5: PBR protocol Architecture

The communication system supports one-hop and multi-hop communications. The *Information Connector* is a way of allowing inter layer communication. This provides some restricted communication between all the protocol layers. This feature is also a way of providing external information to the protocol. The *Management* component works as a way to configure protocol parameters. At the network layer, several types of services are supported.

Beacon Service

Nodes will periodically advertise information such as the node's position and it's identifier. These periodic advertisements are commonly called *beacons*. This makes a node *aware* of it's neighbours. Upon receiving a *beacon* the information is stored and assigned a lifetime, and the information is discarded when the lifetime is reached.

Location Service

This service aims to complement the *Beacon Service* to obtain information about nodes who are not directly connected. This service is only triggered upon request. When a node requires to know the location of another node which is not directly connected to, a *location request* is sent. If the receiving node is the searched node, then a *location reply* is issued otherwise it passes to the next check.

Forwarding Service

Forwarding refers to all PBR services that rely on the *Beacon Service* and *Location Service*. These can be both single and multi-hop services. Some examples follow [7]:

Unicast This is the typical point-to-point type of communication that is also used by other protocols. It can be single-hop or multi-hop, but in any case is uni-directional.

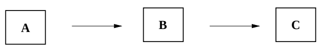

Figure 2.6: PBR Unicast

Topologically Scoped Broadcast This type of broadcast uses the principle of *flooding*. The negative aspects are contained by limitating the number of hops of the broadcast using the hop counter.

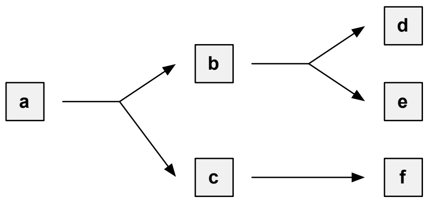

Figure 2.7: PBR Topologically Scoped Broadcast

Geo-Broadcast This feature is intended to generate a broadcast in a specific geographical area. If some node outside the area receives the broadcasted message it will drop the message to assure that the broadcast is confined to it's geographical designated area.

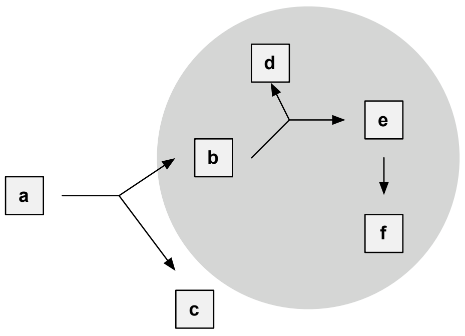

Figure 2.8: PBR Geo-Broadcast

Geo-Anycast Similar to the previous feature, but the message is not broadcasted once it reaches the target area.

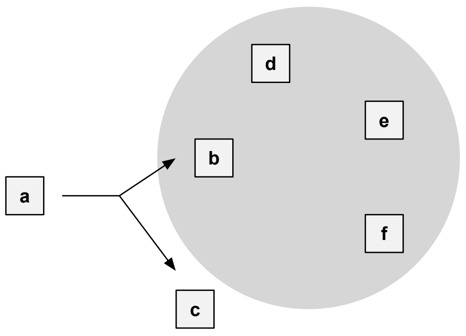

Figure 2.9: PBR Geo-Anycast

2.3 Mobility in MANETs

In Mobile Ad-Hoc Networks, the issue of mobility is an implicit characteristic of the nodes. Factors like speed, heading, the medium and even wheather can affect the node's capability of communication.

Regardless, in this section we will focus in the capability of the nodes to move around freely within a network and even when switching between networks and how it affects the organisation of the network.

In conclusion, small examples of possible scenarios of MANET mobility will be described in a superficial manner. The idea is just to provide a notion of situations that very often happen on these type of networks.

2.3.1 Node Mobility

The most simple and common scenario that happens is the movement of a node within the MANET. In this situation the node by moving itself often causes the link connections with it's neighbours to break. The simple fact that the node moves out of the communication range causes this. However, it's movement may also cause the node to reach into another node's range of communication.

In [Fig.2.10] we can see an example of this scenario. The movement of node `A` causes it to break the neighbouring link with node `B`. But the movement also allows node `A` to be in range with node `C`, wich will trigger the creation of a neighbouring link between them.

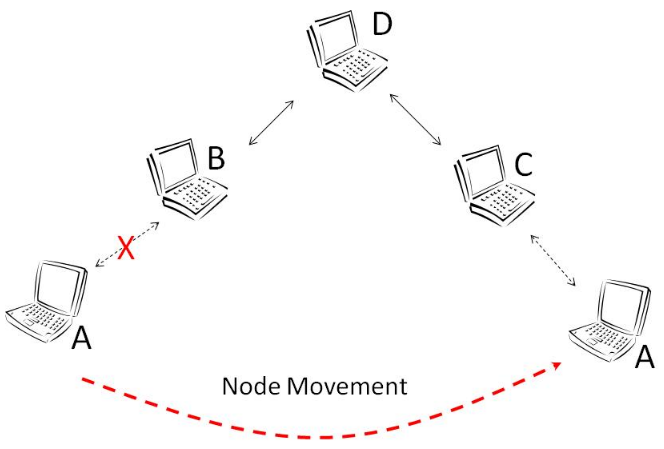

Figure 2.10: MANET node mobility

The overall mobility in this situation can be easily assured by the use of a simple routing protocol. Despite the fact that the movement of node `A` causes a break in communication between nodes `A` and `B` and allows communication between nodes `A` and `C`, the use of a MANET routing protocol will also allow that communication bettween `A` and `B` can continue.

Depending on the type of routing protocol, it will maintain a structured knowledge of the MANET's topology and will update and announce it's information periodically, or it will find the information needed upon request. Regardless, as long as a route (single-hop or multi-hop) is available between two nodes then the routing protocol will provide the necessary information (such as addresses) in order to reach the destination.

In this particular case [Fig.2.10] the existence of node `D` in range of it's neighbours `B` and `C` will allow the formation of a route between nodes `A` and `B` after the movement of node `A`.

Therefore at this level a simple routing protocol can assure that routes are available throughout the MANET, allowing every node to have connectivity to any node.

2.3.2 MANET Mobility with infrastructure support

The support of infra-structure allows many other possibilities to the previous situation, but also adds complexity to the mobility managing.

The first thing to point out is that there are now two levels of mobility. The higher level of mobility is the infrastructured network that can provide connectivity to various other networks such as the Internet, or even other MANETs (as depicted in [Fig.2.11]). The lower level of mobility is the MANET itself where we already established that mobility can be assured by a simple MANET routing protocol.

In this situation [Fig.2.11] node `A` is moving from one MANET to another. The MANETs are both connected by means of an infrastructured network and MANET gateways. The MANET gateways must support both infrastructured and MANET routing as well as support for sharing routing information. From the infrastructure point of view the MANET gateways are just simple routers while from the MANET point of view they are simple nodes that anounce they can reach a great number of networks (and are marked as default gateways when possible).

Once node `A` reaches the destination network (1) the MANET routing protocol will make the information of this event propagate and reach all nodes (including the MANET gateways) (2). The MANET gateways will properly update their infrastructured routing protocol in order to correctly route the information regarding node `A` (3). When the information that node `A` has attached himself in another network reaches the source MANET gateway, this triggers the MANET routing protocol to update itself throughout the MANET (4). Thus, any node on the source MANET can again communicate with node `A` (5).

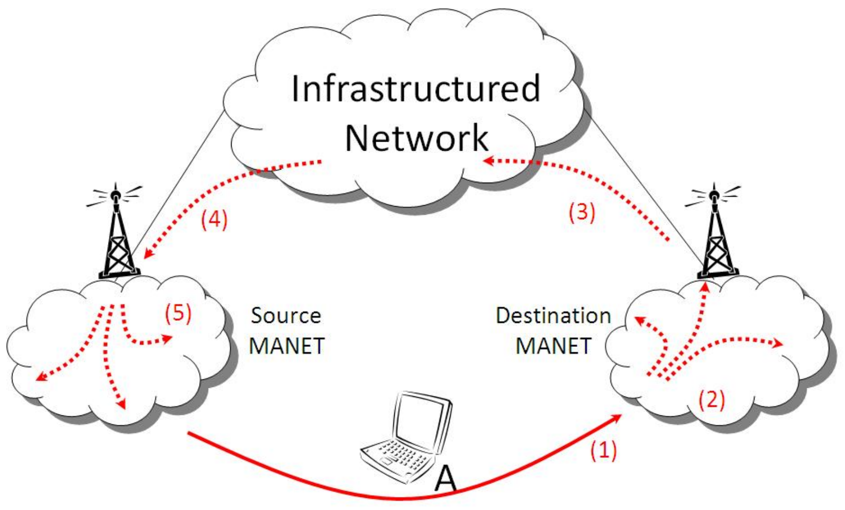

Figure 2.11: MANET mobility with infrastructure support

With the help of infrastructured networks, MANETs can broaden their limits of access without having to compromise their architecture and self-organisation. The possibility of distribution of the mobility management helps to distribute tasks to the responsible network elements in order to simplify the procedure.

Despite of the advantages that can be taken from this approach, there are factors that haven't been taken into account and can make this process more complex such as domains, the use of mobility protocols, support to upper layer features and quality of service.

2.3.3 Mobility in MANETs - Related Work

The purpose of this section is to show other already existing architectural solutions for MANET mobility which have already been implemented and tested. These solutions will be briefly described stating their key concepts and strong points, in order to give a notion of the related work that has been developed on the subject of mobility in MANETs.

IST Daidalos Project - Phase I [26]

The Daidalos project (phase 1) aims at seamlessly integrating heterogeneous network technologies that allow network operators and service providers to offer new and profitable services (voice, data, multimedia). The architecture integrates both wired and wireless technologies, with quality of service (QoS) capabilities under a common authentication, authorization, accounting, auditing and charging (A4C) framework and in a secure communication environment. Mobile Ad-hoc networks (MANET) integration is also one of the main Daidalos achievements.

In terms of mobility, the architecture has support for node movement inside and between MANETs. Inside ad-hoc networks, the mobility is assured by several MANET routing protocols, such as AODV for unicast traffic, SAODV for secure unicast traffic, AOMDV for multipath traffic and MMARP for multicast. The ad-hoc network is connected to the infrastructure network usually through an Access Router (AR). This element is a node (fixed router belonging both to the infrastructure and to the MANET) that routes packets between the external networks and the ad-hoc cloud, and provides the interface to the infrastructure network, in terms of routing, mobility, QoS, security and charging procedures.

As for the mobility between MANETs, there is in place a mobility protocol, which is supported by both the MANET nodes and the gateways (GW). This mobility protocol is MIPv6. When a mobile node moves and starts receiving information messages from multiple GWs, it must select one of them. Changing GW may imply handover. The Daidalos nodes directly connected to the infrastructure (1 hop distance) use a Fast Handover mechanism [26], which provides mobility and very low packet loss probability. The ad-hoc handover process proposed differs from the Fast Handover process: it minimises the handover related signalling messages (since it traverses multiple nodes) at the expense of some packet loss. In both cases, MIPv6 is the protocol used for mobility.

The main difference between phase I and phase II of the Daidalos Project is that the mobility is not limited to the same technology. While in phase I, mobility is only allowed between the same type of networks (in this case MANETs), in phase II this scope is broadened to include other technologies, allowing handovers from other technologies to MANET and vice-versa.

MIP-MANET - Mobile IP for Mobile Ad-Hoc Networks [14]

MIPMANET is a solution for connecting an ad-hoc network, in which on-demand routing is used, to the Internet. MIPMANET provides Internet access by using Mobile IP with foreign agent care-of addresses and reverse tunneling. This allows nodes to enjoy the mobility services of Mobile IP while at the same time the requirements on the ad hoc routing protocol are kept to a minimum.

Simulations of MIPMANET have been performed in Network Simulator 2. The Ad hoc On-demand Distance Vector (AODV) routing protocol has been used for routing within the ad-hoc network. These simulations show that the ability to choose the closest access point to the Internet is worth extra work, as less traffic is generated in the network resulting in lower delays and fewer dropped packets.

Ad hoc networking enables IP mobility within a network whereas Mobile IP enables IP mobility between networks. By combining these two, MIPMANET allows mobile nodes to enjoy extended IP mobility.

Regarding Mobile IP in mobile ad hoc networks, MIP-MANET includes a new movement detection scheme and the use of reverse tunneling to Intemet access points. No major changes have to be made to the foreign agent, all interworking functionality can be put in an interworking unit between the foreign agent and the ad-hoc network. The simulation study has evaluated whether periodic broadcasts of agent advertisements should be unicast or broadcast. Broadcasting agent advertisements periodically is better than unicasting at all times except when there are no visiting nodes in the ad-hoc network. The visiting nodes get more information and can make better choices of where to be registered (and thus which access point to use to reach the Internet). Simulations have shown that the ability to choose the closest access point to the Internet is highly valuable as that lessens the total load on the network.

The advantages of MIPMANET are:

- It separates the tasks of the routing protocol and Mobile IP and provides transparent interaction between the two.
- The changes to Mobile IP only concern the communication between the foreign agent and the mobile node within the ad-hoc network. The behavior of Mobile IP on the Internet side is not affected.
- Only minor modifications have to be made in the on-demand routing protocol for interworking with the Internet.
- Nodes in the ad-hoc network that are not using Mobile IP see the ad hoc network as a stand-alone network, i.e., they have no knowledge about the Internet.
- By the use of a separate module, called the MIPMANET Interworking Unit (MIWU), only minor modifications have to be made in the foreign agents to make the solution work.

The disadvantages of MIPMANET in comparison to this project are:

- It only has a single MANET routing protocol, which is AODV.
- It does not support hieralchical domains, i.e., it does not support the existence of several domain layers, such as, global and local domains.
- It does not support seamless mobility between multiple different technologies and MANETs.
- Concepts like like authentication, authorization, accounting, auditing and charging (A4C) are not taken into account.
- Privacy is another issue that is also not considered.

Micro Mobility and Internet Access in Ad hoc Networks [20]

This platform allows mobile nodes to access the wired Internet and roam from base station to base station. The solution is based on the extension of Mobile IP capabilities to the ad-hoc network while a micro mobility protocol is adapted to support local migration.

This is a solution for Internet access and micro mobility for ad-hoc networks. It relies on the AODV routing protocol for establishing multihop paths between a mobile node and a base station. For micro mobility, the solution is based on HAWAII, a domain based micro mobility scheme.

The transport layer performance of the proposed solution has been evaluated using simulations. The simulations indicate that a fairly high throughput can be achieved, even during very high mobility speeds. However, the characteristics of the wireless environment itself, as well as inefficiencies of the 802.11 MAC layer protocol, lowers the performance when the number of hops increases. By using a less agressive version of TCP such as Vegas, or lowering the maximum window size, the throughput can be somewhat increased. The problem with unfairness needs to be solved before multiple TCP flows can be supported.

The disadvantages of this solution in comparison with this thesis approach are:

- It only has a single MANET routing protocol, which is AODV.
- It does not support seamless mobility between multiple different technologies and MANETs.
- Concepts like like authentication, authorization, accounting, auditing and charging (A4C) are not taken into account.
- Privacy is another issue that is also not considered.

Chapter 3

Network-on-Wheels

After Chapter 2 the VANET environment has been explained. To understand better the focus of the Network-on-Wheels project this chapter starts by explaining the motivation of the project and concepts involved such as simulation and emulation. In addition it is also detailed the process of choosing the tool by comparing it with other tools available. Also the selected tool is analysed. Furthermore, the building proccess of the framework for testing and measuring VANET applications is described. In the end an evaluation of the performance of the simulator/emulator is done, providing a rough idea of the behaviour presented by the integrated tools by means of system resource comsumption.

In **Section 3.1**, the motivation for the project is explained and the concepts of simulation and emulation are introduced. Also presented is the process of choosing the simulation/emulation tool and an overview of the tool selected.

In **Section 3.2**, an introduction to the project problem is stated. A brief overview is taken on the already implemented routing application, and the adopted solution is presented. Also the implementation process is described along with issues found.

Section 3.3, gives details of the methods used for performance evaluation. Also the testbed system itself detailing the system components, configuration and optimizations is introduced. In addition dry-run measurents of the system are presented in order to establish the system's reference in resource comsumption. In the end comparisons are made between results obtained.

3.1 Simulation and Emulation

3.1.1 Motivation for Simulation/Emulation

VANETs are a quite recent subject and there is still plenty of aspects to improve. There are mainly three methods to test protocol performance: field test, simulation and emulation [8] [32].

Usually field tests are only done after extensive reaserch using one of the other two methods. This is because development and implementation of prototypes for real tests are very costly. Another issue is time consumption. Field test takes much time to prepare and to be processed. The advantage of using this method is that testing results are obtained in realistic environments.

These are reflected in some VANET field tests [10].

Simulations have shown that they can be quite flexible intensively repeated and much less time consuming. But simulation also has it's drawbacks such as the flaws of simplifying real world behavior using models. The accuracy of models and their implementations affect the results accuracy.

Emulation uses real world components associated with complex real time modelling to greatly improve the results accuracy. It is however much more time consuming than simulation.

In conclusion, for VANETs real testing is still very expensive to perform. Simulations are widely used but it lacks accuracy. Emulation is a good choice, but much time is required. In response to these facts another solution is created to deliver the best of two methods. A hybrid method of Simulation/Emulation that combines real components to deliver a higher level of accuracy and simulated models to substitute real world parts that are too complex.

Required Functionalities and Desirable Simulator/Emulator Features

The search for an appropriate tool for simulation/emulation was guided by predefined requisites. These requisites were critical to the success of the testbed. Some of them related to the scope of this thesis are shown below.

- Support for Mobility.
- The use of communication protocols to emulate the nodes's behavior.
- The possibility of change of the nodes's protocol stack.
- The ability to simulate physical layer protocols.
- The capability of real traffic generating in the network nodes.
- The possibility of information exchange between the simulator/emulator and other applications.
- The capability to run multiple real VANET routing implementations on a single machine simultaneously.

3.1.2 Simulation

The notion of simulation is closely connected with the notion of system. A system is often regarded as *black box* whose content is not visible. This box produces some sort of result called *output.* This can be the product of the box's reactions to some external information that is *inserted* or self-generated by the box's behaviour.

A simple defenition of the concept is: the computerised representation of the reproduction of behaviours of processes or entities, which exist in the real world. It can also be generalised to the reproduction of any systems's behaviours.

However in order to do such reproductions some understanding of the system's behaviour is required. The knowledge of it's internal components, processes and internal intercommunications is crucial to a successfull computerised reproduction. After the analysis of these items and comprehension of the behaviour's rules and boundaries it is necessary to *translate* them into a format that the computer can correctly process. This format is called a *model* and the process *modelling.* As computers are mainly driven by simple and mathematical logic at the most basic levels it is legitimate that models should be produced in a mathematical format or in a derivative one. For further reference on this section see [15].

Principle of Simulation

As explained above the process of modelling consists of translating the system's behaviour into a mathematical format. Altough this seems quite simple, in reality it can be rather hard. Real-world system's can be very different and very difficult to express in mathematical rules. There are several types of modelling techniques to cope with the diversity and complexity of cases.

One of the characteristics that influenced the development of model is the ability to change system state over time. For instance, a car while moving is constantly changing position. This is easily modelled by a Continuous Simulation Model due to the car's continuous movement. In opposition to this model is the Discrete-Event Simulation Model. A suitable example is the modelling of a clock. A clock only changes it's state in a *discrete-event* way. This means that its state changes instantaneously in very distinct moments in time. These moments are therefore defined as *events.*

In a simulated clock events can be very close only separated by miliseconds or amount of time between events can amount to hours in some cases. In order to improve the efficient use of resources and reduce the overall time consumption of simulation, the *simulation clock* and time-advance mechanisms such as the *next-event time advance* method were developed. The creation of the *simulation clock* allows the current time in the simulation to be run in a way that does not depend on the actual real time.

The *next-event time advance* method as ilustrated in [Fig 3.1] and [Fig 3.2] shows us that the basic action is to suppress the existing time between events making the simulation clock advance from the time of one event to the other thus drastically reducing simulation time and improving the efficient use of the available resources.

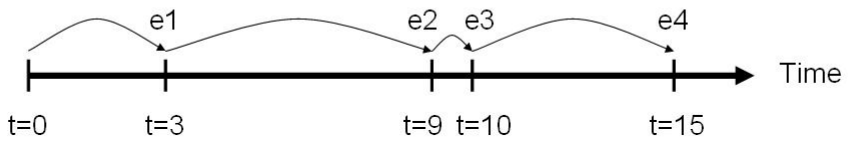

Figure 3.1: Real-Time Events

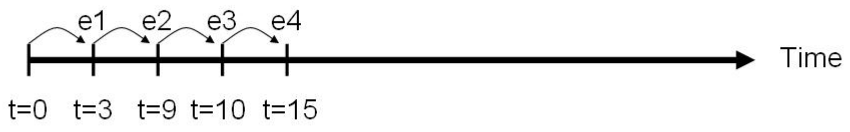

Figure 3.2: Simulated Time Events (using next-time event advance)

Performance and Issues

In the fashion that simulation has been described until now it would seem relatively easy to create one just by following some of the steps described before. Unfortunately it's not that straightforward. It can be a rather complex and prolongued process. One always has to keep in mind certain critical points in order not to produce results that may be completely inaccurate.

A very important point to keep in mind is that the simulation of systems does not always produce *exact* and *precise answers*. If a particular case in study can be solved by analytical methods then simulation only reproduces the analytical result. Otherwise simulation will produce *estimated* results of the system's output.

The level of accuracy can greatly vary and depends on many factors. For instance if a simulation is required to preview the time that a cargo ship takes to cross the Atlantic ocean the level of accuracy can be of hours. A complete different level of accuracy is required by a simulation of a Formula 1 car doing a circuit lap where the precision usually needed is within miliseconds. The difference of the level of required accuracy can greatly ease or harden the process and complexity.

Another crutial factor is the modelling process. Some major aspects are listed here. Firstly, the type of model must reflect aspects of the envisioned data to gather and of the system's behaviour. Secondly, the complexity of the model must be considered taking in account the level of accuracy needed. Neglecting carefull study and decision upon these points, a model may not produce the right results.

The simulation process itself can lead to the production of deceiving results. In addition to the misuse of a good and appropriate simulation model, the scenario set for the simulation can be a point of distortion. The number of replicas and repetitions may also alter the significance of the collected data.

After data has been gathered there is still some room for error. Analysis methods of data can cause a strong effect in the conclusions. Sometimes it happens that the exact same data collected can give very different results when analysed by different methods. Considerating this possibility the data analysis should carefully be done by choosing appropriate methods that suit the context [21].

Advantages

Despite the complexity and drawbacks of simulation, some very good points can be achieved if correct choices and steps are made in the process. It allows comparitive measurement between different systems or the same system with different parameters. It controls the environment and external factors that can affect the simulated system's actions. It significantly reduces the amount of time that is used for simulating systems along great time intervals. Another advantage is that simulations can be applied when pure mathematical analysis is not possible and still obtain useful results with certain accuracy.

3.1.3 Emulation

Imperfect simulation models have difficulty in reproducing real-world behaviour and produce degradation in the results's accuracy as stated before. These results proved in many cases to be very much different (based on the accuracy required) from data collected from real experiments. For these situations this means a waste in time and money doing simulations that would produce inaccurate results.

There is research for alternative methods that would provide more accurate results. One method developed decomposes the simulated systems into smaller interconnected sub-systems allowing to simplify the problem. The introduction of small real-world sub-systems to replace simulated ones helps improving the accuracy of the outcome.

In the computer environment that means the computer (or similar) entities would have their physical components simulated by specific models while interacting with their real software, thus creating the concept of emulation.

Principle of Emulation

Altough several definitions for emulation exist, we prefer to use the following. Emulation is a way to simulate computerised (or similar) entities by modelling the behaviour of physical components and using real implemented software to interact with them in a context scenario [6].

For instance *printing* a document to a *post-script* format is in fact the emulation of a printer. The text editor, by selecting the appropriate printer, will initiate the usual printer communication protocol with the selected printer's *driver*. But in fact the selected printer's *driver* is a piece of software modelled to reproduce the original and real printer behaviour. The following communication will allow the retrieval of the printing information to be redirected to the *post-script* program that will create the desired file.

The whole process is seamless for the text editor, which believes to be communicating with a real printer due to the presented behaviour.

Interaction between elements in such a way can be generalised to other situations. For instance one area in which emulation is becoming quite popular is computer networking. In computer networking, emulation often means one of two cases: the first is a simulated network with emulated nodes while the second is the case of interaction between a simulated network (as the first case) and real-world networks and nodes.

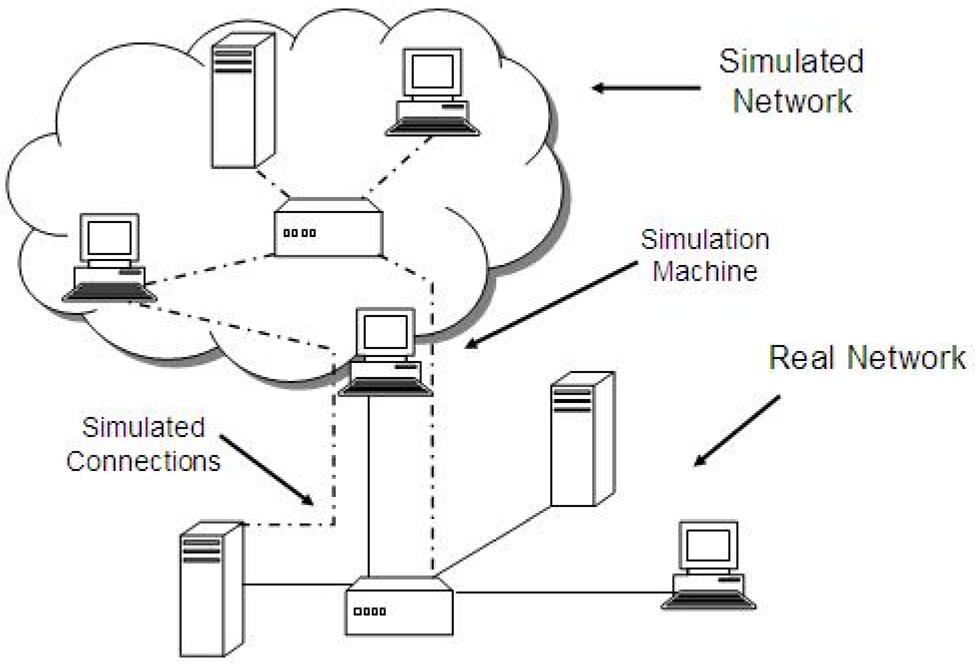

Figure 3.3: Emulation: a real network interacting with a Simulated Network

Applications and usages of these emulation scenarios are quite diverse. For instance one can perform evaluations upon newly developed protocols. Also traffic congestion evaluations can be done. Other usages include the evaluation of protocols over different environmental conditions.

Focusing on the objectives of this thesis, there are two distinct cases of emulation. The first means the emulation of a full scale VANET. The nodes' physical components and mobility as well as the network environment would be simulated. But the protocol software together with other applications would form the real-world component of the network. In the second case it means the interaction of the first case with real VANETs or at least with some fully implemented vehicular nodes. In both cases with the use of emulation valuable reaserch can be done on the particular environment presented by VANETs without some major drawbacks in simulation and also avoids expensive field test.

Performance and Issues

Emulation in network environments has become popular due to it's advantages. However, there are also some points that make this approach less attractive as a means of research.

Time in emulated environments cannot be treated in the same way as in simulations. The mechanisms like the next-event time advance make sense in event-driven simulations but not in emulations. In emulations the time used is or should be real time. A couple of factors contribute to this point. The interaction of simulated components with with real networks or real nodes does not make sense if both are running on different *time scales*. This alters significantly the *natural* behaviour of communication. Another point to bear in mind is that real networks certainly have unpredictable events which cannot be detected *a priori* thus making it impossible to map events into simulated time. Running emulations in real-time has greater time consumption than simulation, thus making it a less attractive choice.

In emulation, using real components means more resources therefore a greater need for processing power. Consequently, computing delays may happen making it difficult to obey real-time constraints and complicate network behaviour may lead to errouneous results [8]. This issue also means problems with scalability both in time and in node numbers emulated.

When compared with simulation results, emulation results are considered to be more close to reality because real components are used.

Advantages

Advantages in emulation can be summarised in a few points:

- Introduction of real implemented software in the environment

  This helps saving time and effort, avoiding to build models of every component, and allowing to use already implemented applications instead.

- Evaluation of Perfomance

  With emulation the real components performance can be evaluated in a controlled and more accurate environment.

- Development of new software

  Emulation provides a very good method to develop testbeds for testing new protocols and applications, thus it gives good indications of it's expected behaviour in real networks.

- Coping with dynamic behaviour and unforeseen events

  As events don't need to be defined *a priori* coping with unforseen network behavior is quite easier. This also helps coping with dynamic network topologies.

- Interaction with real networks

  A very valuable feature is the interaction between simulated and real traffic, which provides a diferent way for projecting and testing complete or complementary networks.

3.1.4 Tools

Having discussed simulation and emulation, different tools and a selection of an appropriate tool is presented here.

Classification

Following is a list of different types of Simulators andEmulators used for VANETs:

- Traffic

  A tool of this type is normally used to incorporate general traffic generating applications. These applications will inject configurable traffic over the designated topology and retrieve statistical data as packet delay, packet loss and network load. Usually very good for analysing heterougenous data flow. They have however constraints on the types of network available for such testing.

- Network

  Here the topology and types of networks are key words. One good point is the possibility of interaction between different types of networks and another one is represented by the great number of represented networks. Typical networks are infrastructured networks, wireless networks, mobile networks, cellular networks, etc.

- Movement

  Patterns of movement are the core of the application, such as configurable speeds, headings, positions and behavior of different vehicles and aircraft types. The purpose is to explore different behaviours in different situations to evaluate what might happen.

- Protocol-Oriented

  These type of tools usually are centered centered in the protocol stacks interaction and protocol traffic. Examples are of routing-specific or transport-specific experiments. They excel at protocol analysis but are too focused.

- Application-Oriented

  In this type of tool the generated traffic in networks is the core of the tool. They are used to include applications that generate a specific type of traffic allowing performance comparisons. They could be specific to for instance VoIP, video streaming, file sharing, etc.

Tool Selection

Taking in account the characteristics required of the tool an application that could combine these required features was researched. The most relevant and interesting applications found are listed below:

- W-NINE [8]

  This is a platform developed for wireless networks emulation. It is based on offline-simulation combined with real networks. Its complex modelling and overall aproach make this framework very accurate and interesting but it also uses a great deal of hardware and a few different simulation/emulation software packages to work. The major drawback is that network topology is not emulated.

- IMUNES [35]

  It provides both network topology and traffic simulation in an integrated and clean way. The graphical interface and quick setup are also a beneficial points. However this tool does not support mobility nor wireless emulation.

- FreeBSD Clonable Network Stack Method [34]

  This method uses virtual images of the network stack to emulate virtual nodes. Albeit having advantages as high performance and very low resource consuming this method also lacks mobility and wireless environment support.

- IKREmuLib [19]

  Although not having graphical interface this emulation tool proves to be quite interesting. Node and environment emulation are present but the absence of the topology emulation requiring much hardware (meaning physical machines) is a rather big obstacle.

- NS-2 [4]

  The most popular tool for simulation. This tool has topology, environment and node emulation. It also has an optional graphical interface. Despite it's strong points it has limited use as an emulator/simulator for our purposes. This is because some severe emulation bugs have been reported but have not been corrected. Also because of the unresponsive behaviour it seems that the emulation aspect of ns-2 is no longer maintained nor supported [29].

- OPNET [12]

  A very complex modeller and complete modeller with simulation tools included. It has very interesting capabilities especially because it now has a wireless version. However it's a very expensive licensed comercial product.

- NCTUns [31] [29] [30] [32]

  A complete and very helpfull tool for simulation/emulation. It emulates network topologies network environment and network nodes. It comes with a graphical and very intuitive interface and satisfies the most of our requisites and criteria for tool selecting.

Conclusion The **N**ational **C**hiao **T**ung **U**niversity **n**etwork **s**imulator (NCTUns) was selected as the software package for simulation/emulation to be used in this project due to the many qualities presented that fit in the project pre-requisites. It is a network/traffic/protocol simulator, and has great versatility and flexibility. Versatility is presented in the numerous elements that can be simulated and emulated. Flexibility is shown by the easy combination of these different type of elements to form various types of homogeneous or heterogeneous networks (thus showing the network simulator aspect). The ability of adding deleting replacing protocols from the nodes's protocol stack allowing to customize the nodes. Also the ability to create and add new protocols is very usefull and contributes to the overall value of the package. In addition almost any traffic genereting application can be executed from the simulated nodes without any modifications.

NCTUns

The initial version of this software the NCTUns [29] [31] [30] [28] [33] [32] was initially designed to reduce or avoid the impact of drawbacks in simulation and it has evolved in many areas since then. In the following NCTUns simulation methodology is firstly introduced. It is followed by emulation mechanisms and the important features.

NCTUns - Kernel Re-entering Methodology

NCTUns uses real pieces of software incorporated in the operating system to process incoming and outgoing packets. This allows to introduce real behavior in the packet processing than rather modelling it. In a more specific case we can see in [Fig. 3.4] [31].

As depicted we can see the processing of IP packets. Emulated devices (i.e., Host 1, Router 1, Host 2) have associated with him a tunnel interface binded to it's own IP address. The outgoing packets are then sent to the kernel TCP/IP protocol stack for processing. Afterwards the packets go to the node's tunnel interface that works similarly to a normal Ethernet interface. Once here the kernel, which was modified to do so, will divert the packets to the simulation engine that will simulate the transmission environment and routing. Similar procedures are done in the receiving side. This procedure allows real world nodes to be added to the simulation making the results more accurate. However this may generate delays in processing the whole scenario if the number of nodes accessing the TCP/IP protocol stack is too high.

For nodes that operate at a level lower than IP (as the depicted switch) the simulation engine is responsible for all the needed procedures thus everything concerning these devices will be treated within the simulation engine.

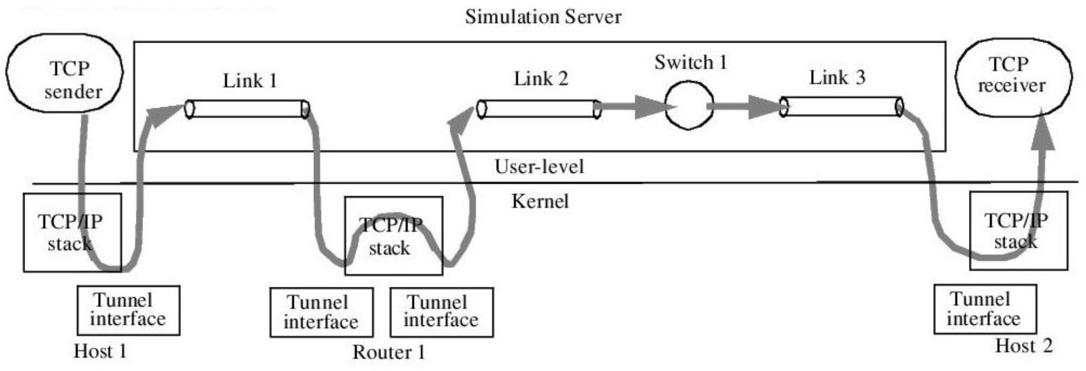

Figure 3.4: Kernel Re-entering Simulation Methodology

NCTUns - Emulation Daemon

The emulation on this tool is mainly implemented by the Emulation Daemon. This piece of the program is in charge of assuring regular communication with real nodes. As pictured in [Fig 3.5] [31] the emulation daemon is inserted in the simulation engine's representations of external nodes. External nodes are used to represent nodes outside of the simulation world and can communicate with the simulated network inside the simulation engine. Usually these external nodes are network elements connected to the simulation machine. This allows the simulation engine to set-up communication properly with these nodes. The Emulation Daemon simply provides a service as a proxy and as an address and port translator/filter between external nodes and simulated nodes.

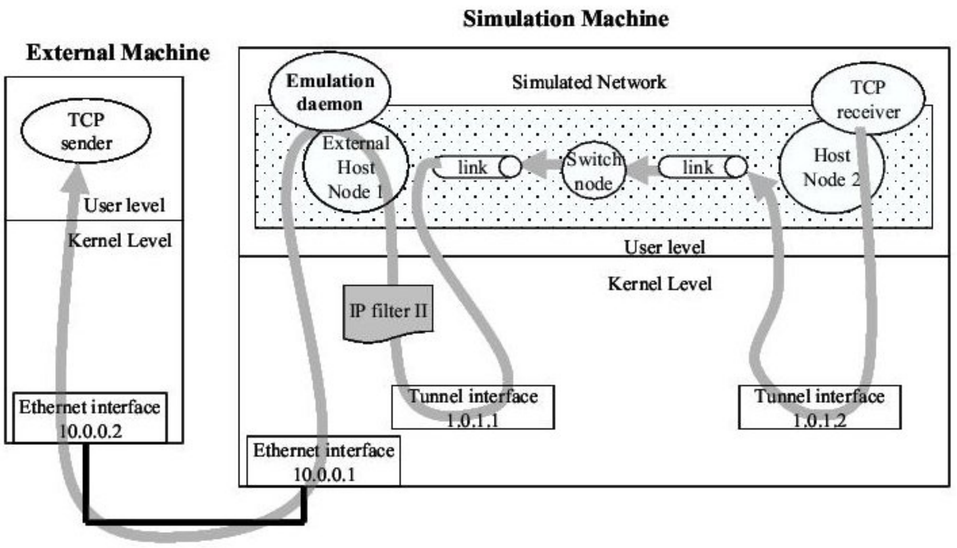

Figure 3.5: Emulation Scenario: Emulation Daemon

It is simple to use the tool as an emulator just by following a couple of steps aided by the graphical interface. In addition, if emulation performance is an issue, the emulation daemon can be converted to a kernel module.

NCTUns - Time Control

The NCTUns has mechanisms to provide time information thus controlling events and overall time of the experiments. This is achieved by the use of a Time API and kernel modification.

For application level programs time is obtained by the simple call of functions that return time values in an approprite structure. These applications requests data are sent to the kernel by the means of *system calls*. The NCTUns kernel however was modified and rebuilt to allow that specific *system calls* can be redirected to the simulation engine in order for it to answer the requests instead.

What happens with the large majority of time-related *system calls* is that the kernel simply acts as an middle-man redirecting the requests to the simulation engine time API. Upon receiving an answer the kernel forwards the data replied to the corresponding process. The Simulation Engine thus controls the system time using it as seen fit.

In simulation this explains how time is manipulated by using an implementation of the next-time event advance mechanism together with a scheduler in order to produce discrete-time events that can easily be detected and processed. As for emulation the real clock is used.

However in both cases if the processing of events cannot be done within the time restrictions imposed time can be *extended* in order to maintain the boundaries of simulation time, i.e., time is *stretched* so that the simulated time is not exceeded.

NCTUns - Features

The features presented by NCTUns have strong arguments that made us choose it as the base to develop this project. These features with respect to our requirements are listed as follows:

- It is a Linux based and open source tool allowing modifications and contributions. It is freely licensed to university and non-comercial use.
- Albeit not supporting specifically the VANET subject it supports Mobile Ad Hoc Networks (future versions will support vehicular networks [32])
- The methodology of the tool contributes to assure more accurate results by using real software to simulate the node's behaviour.
- The modular protocol binding of a node permits the change of the protocol stack of the node by whatever required protocols. Furthermore if the required protocols are not implemented the architecture of the tool allows the addition and building of these lacking protocol modules by the user.

- It can run multiple applications simultaneously on the real protocol stack.
- The tool permits the execution of any UNIX traffic generating application program in the simulated nodes without no modifications [29].
- The failure to provide intercommunication of simulation data between applications and the simulation engine is a definitive drawback.
- The graphical interface has simple and intuitive ways of creating, importing and exporting a network's topology as well as the importing and exporting of a network's traffic commands.
- Statistical data is provided and filtered for every experiment. In addition one can implement new ways of obtaining data logs which are not previously available.
- Measurements and other resource related issues are compared and evaluated in [Section 3.3]

As the items above can show almost all the requirements were met therefore making the tool the strongest choice for the base of the development of the project. However there are a few setbacks that must be worked on to find alternative solutions that fit our needs.

3.2 Building a Hybrid Testbed

3.2.1 Problem Statement

The building of a hybrid emulation and simulation test and measurement environment for VANETs conceptually is the joining of the emulation and simulation methods in an attempt of obtaining more accurate results than pure simulations. With this new approach VANET protocols and applications can be tested and measured providing results closer to the performance data of an eventual real prototype implementation.

Putting in practice the above described goal is not that simple. First it requires a tool that simulates/emulates the basic environment characteristics. Secondly it requires a real application that implements the basic principles of VANET architecture and routing. Both these applications have been presented in prior chapters. The main task and goal of this thesis is then to perform the interconnection between these two tools in such a way that their cooperation provides the desired testbed features and produces more accurate results.

3.2.2 Network on Wheels demonstrator (NoWd)

The *Network on Wheels* Project (NoW) [3] is a research project on VANETs and it creates prototypes of vehicle-to-vehicle and vehicle-to-roadside communications. From this project an important achievement is the elaboration of the Network on Wheels demonstrator (NoWd) [7]. This demonstrator is an evolution from the *Fleetnet* Project

[11] demonstrator. The demonstrator is centered in the VANET OBU nodes since they are the most essential part of a VANET, and it interconnects the In-Vehicle and Ad Hoc domains.

First we give a brief overview over the NoWd's architecture and then we enter some more detailed implementation to give an idea of how the software works and it's particularities.

Component Architecture

In [Fig. 3.6] we can see the reference composition of a VANET OBU.

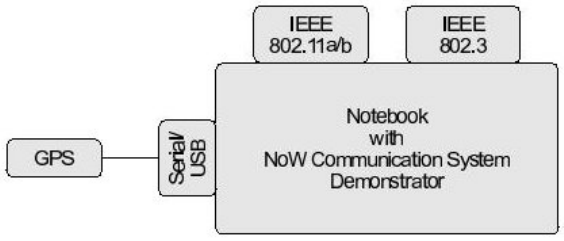

Figure 3.6: NoWd prototype component architecture

There is another composition more detailed that can be seen in [7] but for simplicity purposes we will only use this configuration. There are three distinct interfaces. The wireless interface (IEEE 802.11a/b) is used for communication with other nodes (OBUs) and infrastructures (RSUs). One of the wired interfaces, the Ethernet interface (IEEE 802.3) is used for communication with AUs. And the other wired interface (Serial/USB) is used for obtaining position from a GPS device. The wireless interface is usually connected to an external antenna in order to improve antenna gain.

Protocol Architecture

The protocol architecture as depicted in [Fig.3.7] is a description of the software components and how they are connected and some of the hardware. The PBR Core can control some configuration of the hardware through the Management interface. The PBR Core basically provides PBR routing services. Above the PBR Core there are the PBR-Unaware and PBR-Aware applications. The latter shares infomation with the PBR routing Core by means of the Information Connector.

Compared with the PBR protocol architecture 2.5, the resemblence is evident. Although a rougher and conceptually simpler version, the NoWd protocol architecture contains the core elements of the PBR protocol architecture.

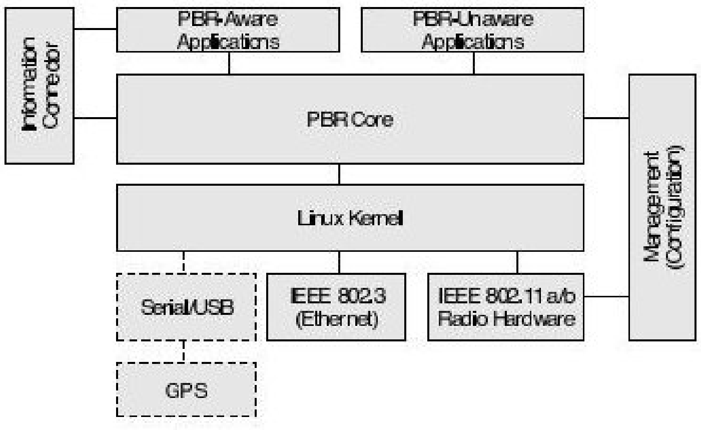

Figure 3.7: NoWd prototype protocol architecture

Features

Adressing The NoWd uses four types of adresses, and each one has it's own purposes. The first one is the MAC address. This address is the identifier associated with the wireless interface. The second one is the NoW address which is used for routing by the PBR algorithm. The third one is an IPv4 address that is used for IPv4 communication. The fourth address is IPv6 address.

DHCP & RADVD The services DHCP and RADVD are generally for AUs self-configuration. The range of addresses generated is controlled by the OBU.

Information Connector The information connector is an optional feature. It can be usefull for sharing of PBR information between the PBR routing core and the applications in an efficient way.

Packet Prioritization & Packet Caching Packets may be prioritized so that packets with a higher priority will be forwarded with priority. Packets may also be cached temporarily to provide a greater degree of reliability.

Software Interfaces

The software interfaces cope with the necessity of communication between the several components of the NoWd protocol architecture as presented in [Fig. 3.8].

The main software interfaces of the NoWd. A more detailed description of each interface's purpose and the implemented solution follows.

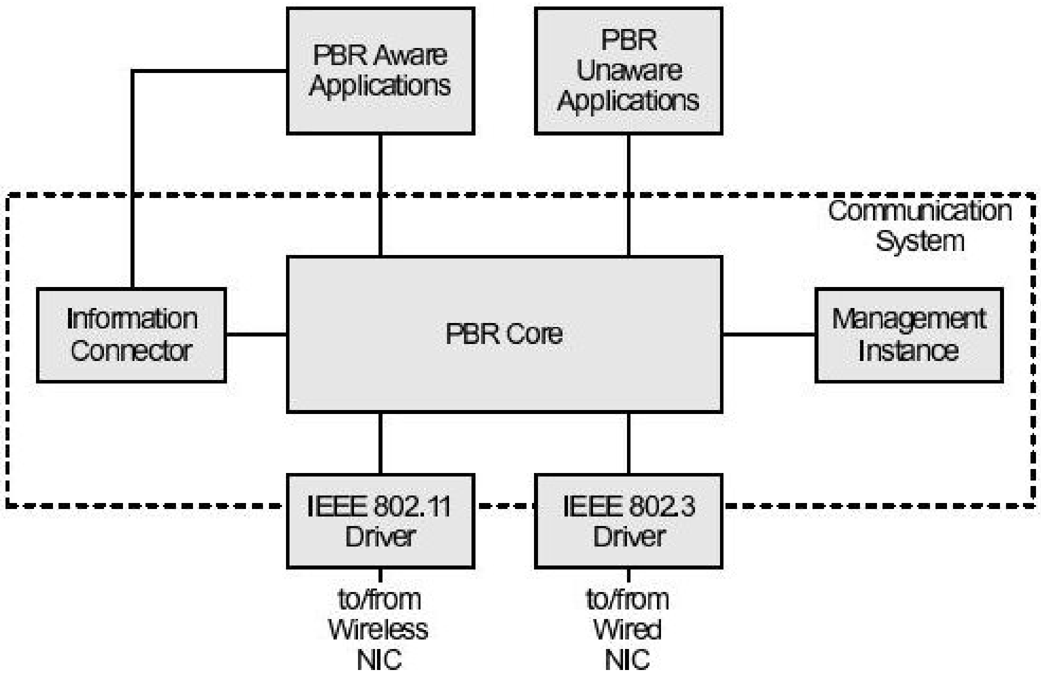

Figure 3.8: NoWd prototype software interfaces

PBR Interface Located between the PBR Core and the wireless driver (IEEE 802.11) the interface's purpose is to receive and transmit PBR packets. It's implementation is as a low level socket (PF_PACKET type) to access the link level information provided by the wireless interface.

PBR-Aware Interface It is located between the PBR Core and the the PBR-Aware applications. Information that flows over this interface comes from applications that specifically use PBR data in their packets. And it is used by the PBR core to send UDP data to the AUs. It is implemented as a UDP socket.

PBR-Unaware Interface & AU Interface The PBR-Unaware interface is located between the PBR Core and the PBR-Unaware Applications, while the AU interface is located between the PBR core and the wired driver (IEEE 802.3). Both interfaces handle the packet flow between the PBR core and the AUs that do not use PBR. The AU interface only receives packets sent by the PBR core and is implemented as a low level socket (SOCK_RAW type). The PBR-Unaware interface is used for the flow of packets between the PBR core and the AUs. Its implementation is rather complex. Simply saying it is basically a netlink socket that assures the right communication with the help of `iptables` [2] rules.

Management Interface Located between the PBR Core and the Management Instance, the management interface usage is usually to configure the PBR core by updating it's information. It's implemented solution is a UDP socket.

Information Connector's Interfaces Located between the Information Connector and both the PBR Core and the PBR-Aware applications, both interfaces are used to exchange information with the Information Connector are implemented as UDP standard sockets.

Frontend Description of NoWd

The input and output of the NoWd. Events are essencially related to packets or timers. In the case of a packet event, i.e. receiving a packet, appropriate functions are called to handle the packet. In case of a timer, the expiring of the timer's time value triggers the specific action associated with the timer.

Event generation is started by an ongoing function while performing simple actions as the exchange of information between interfaces or while executing functions that create and schedule timers. When the timer expires it will trigger an event. Such event can be for example the sending of application data to the wireless interface or a periodical beacon messsage.

Event receiving, i.e. packet receiving, is independently being treated by different handlers depending on the event type. This is achieved with the help of the `select` function [2]. The `select` function will check the appropriate software interfaces for any incoming messages until a timeout value expires. If a message is received the function returns the specified software interface and the correspondent handler is called to process the message and take action accordingly. If however no message is received during the time value of the timeout parameter the function returns with no value. The timeout parameter value is the elapsed time betweeen scheduled timers.

In conclusion the main program (PBR core) flow is seen as an endless loop. At the beggining of the loop there is a check for the existence of the next timer (the next scheduled event). If there is none, then next event timer is created and scheduled. Following this check the `select` function is called with a list of the interfaces to be checked and the time remaining to the next scheduled event timer. Upon returning one or several interfaces the correspondent handlers are called to deal with the received events. After the return of the `select` function a last check is made to the timer seeing if it already triggered himself. This is followed by some timer management clean-up and the returning to the beginning of the loop.

3.2.3 Adopted Solution for the Testbed

On the one hand we have a tool (NCTUns) that comprises protocol emulation/simulation. On the other hand we have another tool (NoWd) which implements VANET routing. The joining of both tools is difficult due to restrictions on both sides.

The NCTUns conceptual architecture [Fig. 3.4] only allows user-space traffic generating programs to run on virtual nodes that are above IP layer.

In addition there is a far more strict restriction that any protocol layer of the virtual nodes protocol stack (between IP layer and physical layer inclusively) that needs to be created or inserted in order to replace an existing one must be implemented as a NCTUns protocol module. This is a grave issue since NoWd is already a fully implemented tool that covers link, routing and transport layers. Also the NoWd program uses some link layer information.

The process of completely or partially implement the NoWd as NCTUns module has two major undesired consequences: firstly it would need quite some time to model and adapt such a big program across all the involved layers; and secondly it would imply that later changes and added features or modules needed also to be implemented in the developed NCTUns protocol module for obtaining new resuls.

The solution resides rather in small adaptations towards their coexistence in the emulated/simulated environment. The interfaces of the NoWd application [Fig. 3.8] become the key to resolve this problem. Interfaces are responsible for receiving and sending packets. In a practical point of view they control all communication from and to the NoWd PBR Core. Furthermore they define at what type of layer the communication is executed.

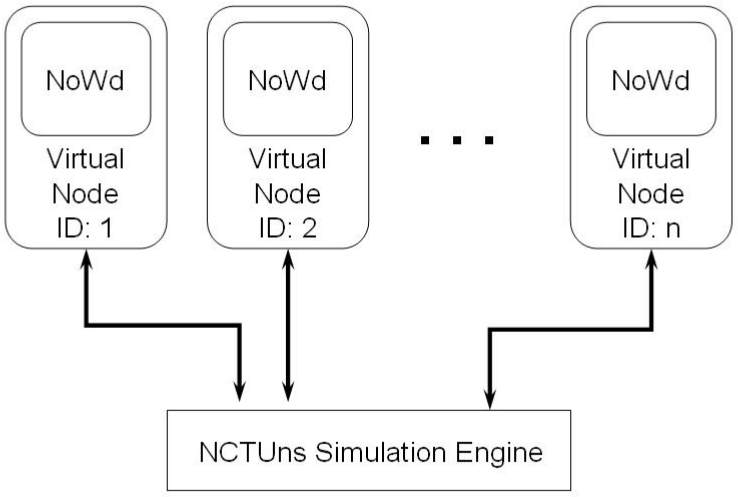

Figure 3.9: Testbed architecture

After some investigation a solution was elaborated, the allocation of a NoWd program running in each virtual node 3.9. By changing the interfaces to work in the transport layer all the NoWd packets could be encapsulated into NCTUns TCP/UDP packets. The NCTUns would treat the NoWd program as a traffic generating application without needing to make major changes. This is solution although not ideal, it proved feasible.

In 3.10 we can see the detailed resulting protocol stack of the testbed, with NCTUns being responsible for lower layers and the NoWd as in charge of upper protocol layers.

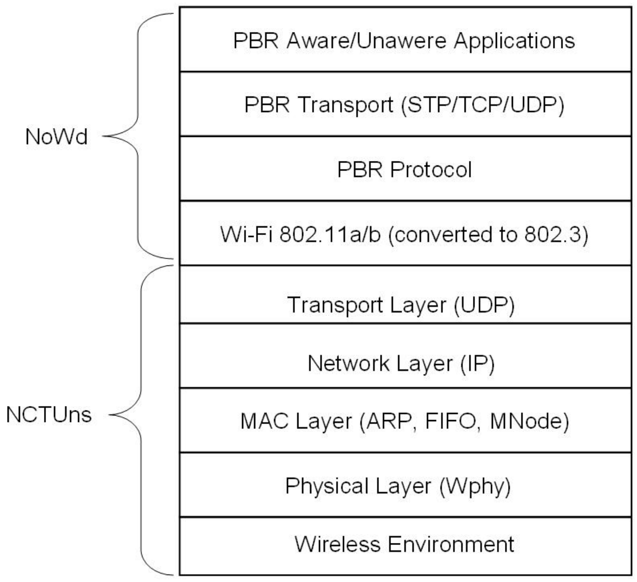
NoWd
PBR Aware/Unawere Applications
PBR Transport (STP/TCP/UDP)
PBR Protocol
Wi-Fi 802.11a/b (converted to 802.3)
NCTUns
Transport Layer (UDP)
Network Layer (IP)
MAC Layer (ARP, FIFO, MNode)
Physical Layer (Wphy)
Wireless Environment

Figure 3.10: Testbed Protocol Stack

3.2.4 Implementation

The purpose of this section is not to dwell on details of the integration of both tools but rather to give a clear idea of what problems were found and the approach used to resolve them. Therefore after a listing of the contributions made to this project a simple list of these events was gathered and is shown.

Contributions

The contributions consist in the designing and integrating the NCTUns and NoWd tools in order to produce a testbed for VANETs.

- The integration of both applications.
  - Modification of NoWd with respect to requirements of the testbed, using compilation options so that compatibility with real node is assured.
    - Removed incompatible features.
    - Addition of new interfaces.
    - Changing interfaces management and defenition.
    - Changing of NoWd's behaviour on sending and receiving packets.
    - Adding new funcionalities for NoWd's internal use.
    - Changing of support programs to support the testbed constraints.
    - Changing of the execution script `nowe` to support the testbed constraints.
  - Modifications in the NCTUns to specificaly support the testbed.
    - Implementation of a mechanism to communicate the NCTUns's node position to the NoWd's correspondent node.
    - Implementation of a conversion algorithm from carthesian to geodesic coordinate format.
    - Implementation of a mechanism to properly calculate and display the execution time.
- The implementation of mechanisms to simplify the usage of the testbed.

  Creation of the `nowe_install` script to:

  - Easily Install multiple nodes.
  - Easily Uninstall multiple nodes.
  - Minimize disk space consumption by allowing nodes to share certain files.
  - Define nodes's start time and end time of NoWd's execution.
  - Define warmup and warmdown periods.
  - Create the traffic application file.
- The realization of performance tests on the testbed providing some data about the behaviour of the testbed.

Selected Features

Obstacle When executing the NoWd program in virtual nodes continous problems emerged blocking the normal execution of the application.

Reason This problem had a slow and time consuming resolution process. The reasons of blocking the regular program flow were the use of functionalities, interfaces and mechanisms that the NCTUns does not support in their virtual nodes. Some examples include the IPv6 protocol for addressing and the creation of a dummy interface in the virtual node.

Solution The solution found was simply to remove any access or reference of those features from the NoWd starting file thus creating a new version of NoWd for simulation/emulation.

Socket Adaptation

Obstacle While running the NoWd application on a multi-node scenario it was detected that the nodes could send beacon packets but could not receive them. This behaviour became similar for all other types of packets.

Reason In a single-host scenario UDP clients bind their sending sockets to the local IP address and a port number in order to send packets, while UDP server sockets only need to be binded to a port number. In the case of a UDP socket being binded to a specific address the ability of that socket receiving broadcast packets is eliminated. In a simulated wireless multi-host environment the ability to receive broadcasts is vital thus creating a sort of paradox: by binding the wireless PBR communication socket to an address the ability to receive broadcast packets would be compromised; by binding the wireless PBR communication socket to no specific address would result in an impossibility to send packets.

Solution Though the problem seemed confusing the solution adopted was quite simple: the substitution of the only one wireless PBR communication socket by a pair of wireless PBR communication sockets one for receiving and another for sending. Measures were taken considering port numbers chosing in order that all nodes have the same receiving socket port number (for broadcast purposes) and assuring that the sending of packets is also to this port number.

Address Adaptation

Obstacle A problem was detected while trying to send Unicast packets over the PBR protocol in a multi-hop route. The packets were tranmitted but they would not reach the destination being dropped by the all listening nodes in range of the first hop.

Reason The reason behind this problem was an addressing fluke. Packets were being sent through the wireless PBR socket associated to the broadcast address regardless of packet type and destination.

Solution The solution is to sort the type of packet before sending and in case of not being a broadcastable packet then the destination's node ID should be converted to correspondent IP address for posterior association on sending the packet.

Node Identity

Obstacle The two applications have different procedures to obtain or calculate the node identifier which in both cases is used for addressing. The assurance of node ID's consistency between the two programs proved to be an obstacle.

Reason NCTUns uses a method of self-generation of the nodes's IDs and it is impossible to change a node's ID after the creation of the node. The NoWd however gets the nodes's IDs through configuration files. It is then clearly easier to modify the data received by the NoWd than to change the NCTUns's way of atributing the node's ID.

Solution Resolving this problem implicated a good deal of small changes in order to automaticaly have the correct node ID. Firstly the total number of simulated nodes must be known in order to install that exact number. Secondly during installation (using the `nowe_install` script) nodes's folders and files are created sequentially and it's configuration files are changed in order to reflect the current node ID. This allows NoWd to know it's own node ID by reading the contents of it's own configuration files. Remaining only the point of how does the NCTUns know which NoWd node number to run in which of it's own nodes. This last issue is solved by the importing of a traffic application file by NCTUns that contains commands specifying in which NCTUns node to pass a parameter (current node ID) to the execution script of NoWd (`nowe` script) that has the responsability of using the right files and execution parameters for the correspondent node.

Position Feeding

Obstacle A very tough problem that was found is related to the consistency of the node's provided position for the two programs in the same format regarding that both applications have different ways of obtaining position and the position itself is in different formats.

Reason The source of the problem here is the inability of NCTUns to provide real-time position information to virtual nodes's applications. The NoWd needs to be provided with data about his own position. It also has mechanisms in place for receiving such position data from for example a GPS device or from the management interface through the use of Location Update packets. Another fact regarding position is the different formats used by each program. NCTUns uses carthesian coordinates and NoWd uses Geodesic coordinates.

Solution The implemented resolution uses a tricky method that is not the most efficient of approaches but it's feasible. A change was done to the wireless physical module of NCTUns in order to perform five simple steps every time a packet is sent through the virtual wireless interface: first (1) open a socket bound to the node's IP address; second (2) get the position of the node by using the NCTUns's API; third (3) convert the position coordinates from carthesian to geodesic using the Borkowski's Non-Iterative algorithm; fourth (4) build a simple Position Update packet with the returned geodesic coordinates by the conversion algorithm; fifth (5) send the packet to the local node's IP address and port of the management interface and close the socket. With this method position is updated in the worst case once every time a beacon is sent. But in cases of high load networks this means heavy resource comsumption.

Position Update

Obstacle During testing the observation of logs led to another problem. Only a specific number of nodes was successfully self-updating it's own position thus the issue of assuring consistent positioning between the two applications was still unsolved.

Reason Behind this problem is a bug in the translation of the nodes's ID to the nodes's IP address. The characters read from the configuration files were converted directly without suppression of zeros. When converting from host byte order to network byte order the zero character on the left of a number in the IP address induced an error producing the wrong address and the packets would not arrive the own node's management interface.

Solution To solve this problem a different method of converting the characters read from the configuration files was applied that suppresses any zeros present to the left of a number thus producing a correct IP address and correrctly delivering the location updates to the rightfull nodes.

Installation

Inconvenience Much time was wasted in configurations setups and file managing every time that a test experiment needed to be done.

Solution The `nowe_install` script was elaborated to help on the easy installation and removal of the supporting files needed for each node and the main execution file for NoWd the `nowe` script. In addition this file does the necessary changes to the correspondent configuration files of every node. This script also creates the important traffic application file that is to be imported by NCTUns in order to know when and in which node to execute which command. Also in this script it's the calculation of when each node is started and ended therefore defining the warmup and wardown periods. Finally the last function performed by the script is to add a traffic generating command used during the project for testing.

Warmup and Warmdown Periods

Osbtacle Random nodes would not start or finnish the NoWd program at the previously pre-defined moment. The nodes which did not start in time would not run the

NoWd causing experiments to be executed with only a limited number of nodes. For the nodes who would not finnish in time their processes would not be correctly terminated.

Reason Applications like the NoWd are not light on execution thus nodes nodes should not start their executions of the application in the same instant for some obvious reasons: first they would heavily overload the system in the beggining of the experiment; second they would be probably competing for the same resources which would delay the start of all executions; third they would take more time until they stabilize and properly run.

Solution The solution found is the use of warmup and warmdown periods. Warmup and warmdown are time intervals correspondently at the beggining and end of the experiment time space. Their purpose is to avoid certain factors as stated above and help to reduce instability and high peaks of processing during the experiments. These type of time periods have been implemented in the project. The responsability of calculating how much time is needed and the spacing between the executing events is taken by the `nowe_install` script. The choosing of this script as the place to do such actions is obvious because it is the script that creates the traffic application file which contains the commands for executing in the virtual nodes and also it has information about how many nodes are being used in the experiment.

3.2.5 Emulation Mode

The purpose of this part of the report is to describe the emulation scenario. Despite the complete integration between the NoWd and the NCTUns only one type of interaction has been seen so far, the simulation/emulation of a VANET in the simulation machine. Another type of interaction possible is pure emulation. This section then describes the building of a new scenario in pure emulation.

Scenario Description

This developed type of scenario consists of the interaction of a simulated/emulated network of mobile nodes with a real world host as depicted in [Fig. 3.11].

As it is clear in [Fig. 3.11] the real world host is represented with a special node (marked with an `e`) in the simulated/emulated environment with the purpose of being able to communicate with simulated/emulated nodes. This special node is used to connect the virtual node's protocol stack with the emulation daemon [Fig.3.5]. In this way the simulation machine knows that there are real nodes connected to the simulated/emulated network and can route the traffic as the machine will work similarly to a proxy or gateway between the simulated/emulated network and the real node.

The great advantage is that real implemented prototypes can be able to communicate and exchange real packets with a simulated/emulated network. Also a whole new type of testing can be done with this testbed.

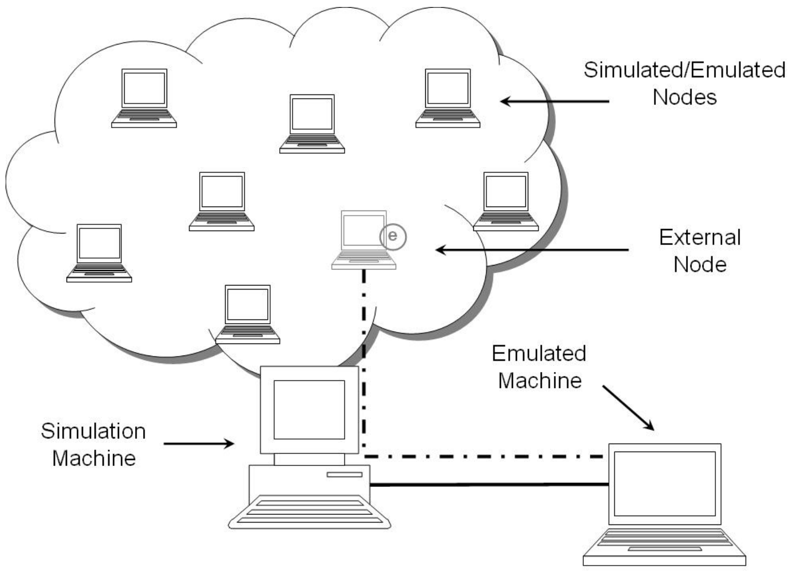

Figure 3.11: Description of Emulation Scenario

Implementation

The status of the testbed at the end of the previous section prepared the software in good part to work in this scenario. The NoWd would work above IP level by the use of UDP sockets but the node ID posed again as an issue. Solutions used before were tryed but could not be applied. Therefore a new solution was needed.

A simple method was used to cope with this problem altough being a little inconvenient. A few small modifications in the program together with the manual configuration of the node ID solved the problem.

In conclusion for this scenario some beforehand extra procedures need to be taken. These procedures are described in three steps: first the add of the routing rules as advised by the NCTUns; second the manual change of the node ID in the configuration files of NoWd; third execute the NoWd with the correct node ID as it's argument.

Restrictions and Open Issues

A restriction of this scenario is the need of the real host to be inside the same network subnet of the simulation machine.

An incovenience in emulation of real hosts is the extra manual procedures that need to be done. Also as emulations run in real-time this means that real hosts need to operate in real time.

Open issues that arised from the emulation and emulation results mainly are due to in the fact that broadcast does not work well in these type of experiments. For instance a real host can broadcast a message into the simulated/emulated network but broadcasted messages from within the simulated/emulated network never reach the real host. This happens due to the fact that NCTUns assumes that application traffic in emulation is typically made of point-to-point unicast causing a serious flaw in communication.

But this flaw becomes even graver for VANETs because of the PBR protocol behaviour. The fact that broadcasted beacons by the simulated/emulated nodes cannot be listened by the real host makes the real host not aware of any neighbours. Aside this flaw unicast traffic works fine allowing other types packets to be regularly exchanged between the nodes.

3.3 Performance Evaluation

3.3.1 Objectives and Evaluation Methods

Objectives

Mainly the purpose behind the evaluation is to provide an idea of the performance demonstrated by the testbed. This performance testing however is neither to be exhaustively analysed nor to be extensively repeated.

The main objectives are structured as follows:

- Generally give a rough idea of the overall performance of the testbed.
- Analyse the testbed scalability in terms of number of simulated nodes.
- Measure System Time Comsumption
- Measure System Resource Comsumption
  - CPU Usage
  - Memory Usage

Methods

Methods are data sampling, data processing and execution time measuring. Regarding the main objectives of the performance analysis, the methods should be easy to use and time inexpensive.

Data Sampling

The sampling of data is to obtain instant system resource status. In addition another way was needed in order to allow the instant system information fetching. That way also needed to repeatedly fetch the information in regular intervals, since the begining of the experiment until it's end.

A way found to collect data is the `sar` command [2]. This command regularly takes samples from the system status regarding several aspects that can be preselected thus allowing us to filter the information needed. Also the spacing of the sampling and the limit number of samples can be predefined.

For defining the parameters taken by the sampler, it's assumed that the execution time of the experiment is not previously know. This is assumed because the real execution time varies as results show. In this sense sample collecting must be taken until the forced stoppage of the sampler execution. In addition the spacing between samples was set to the smallest possible (1 second). Also parameters for filtering data were considered resulting in the commands shown below.

For sampling data: `$ sar -P ALL -q -r -u -o <filename> 1 0`

For reading sampled data: `$ sar -P ALL -q -r -u -f <filename> 1 0`

To integrate the data sampler command with the experiment process, for all experiments the following procedure was taken. First the sampler is manually started immediatly followed by the experiment execution. The when the experiment is finished it is immediatly followed by the manual forced stop of the sampler. The process has a small margin of error that is resumed to generally one sample per experiment. This error however is quite acceptable regarding the nature of the evaluation measurement and the average number of samples per experiment.

Data Processing

For simplicity the method used for data processing is the mathematical mean of sampled data over a test processing. In addition the data sampler already provides the mean of the samples which greatly helps in the calculation of results.

Execution Time Sampling

Execution time and simulated/emulated time are not the same. The latter refers to the time used in the experiment meaning the time space that is supposed to be simulated/emulated while the first one is the time that the experiment spent on a computer.

In our experiments in order to obtain a means for comparison between both simulated and execution times, it was needed to calculate execution time. This however is not provided by the NCTUns therefore it was needed to be implemented.

Before implementing the excution time calculation three questions needed to be answered, i.e. when to start counting time and when to stop counting time and how to calculate the elapsed time. The first two can be obtained by looking at the console output of the NCTUns where some indications are on when the simulation starts and ends. The method used to calculate the time is very simple. Two time points are taken. One is taken right before the invocation of the simulation starting method. The other is taken right after the simulation ending method returns. After taking the two time points and allocating them in appropriate structures the calculation of the time space between points is done and printed on the console.

3.3.2 Testbed Configuration

Hardware

- CPU

  AMD Athlon 64 3000+ (2 GHz)

- Memory

  1 GB PC2700 (2x512MB, DDR333 MHz)

- Hard Drives

  Maxtor DiamondMax Plus 9 120 GB

- Motherboard

  Asus K8N-E Deluxe

- Graphics Card

  ATI Radeon 9200 SE

- Network Card

  3com 3c905c-TX/TX-M

Software

- Operating System

  Linux Fedora Core 4 with full installation. Also with a rebuilt kernel for NCTUns, version 2.6.11 of 03/05/2006.

- Swap Space

  Swap space is configured for using 2 GB.

- NCTUns

  NCTUns is completely installed not with the kernel patch, but with a rebuilt kernel instead.

- NoWd

  NoWd source code and it's binary installation files.

Dry-Run Reference Data

In order to establish the behaviour of the system at startup, and also when running only some of the NCTUns components and a single NoWd execution some measurements were taken, describing the reference behaviour of the system.

In total five scenarios were considered. Firstly when the system is running the startup programs only, with no other applications running. This reference situation is called *Basic Running*. Secondly it is the situation when the NCTUns's main programs are also running. Thirdly it is the situation where the execution of the NCTUns's Graphical User Interface is added to the previous measurement situation. The last two situations comprise the execution of a small experiment of a single node running the NoWd application in emulation or in and simulation mode.

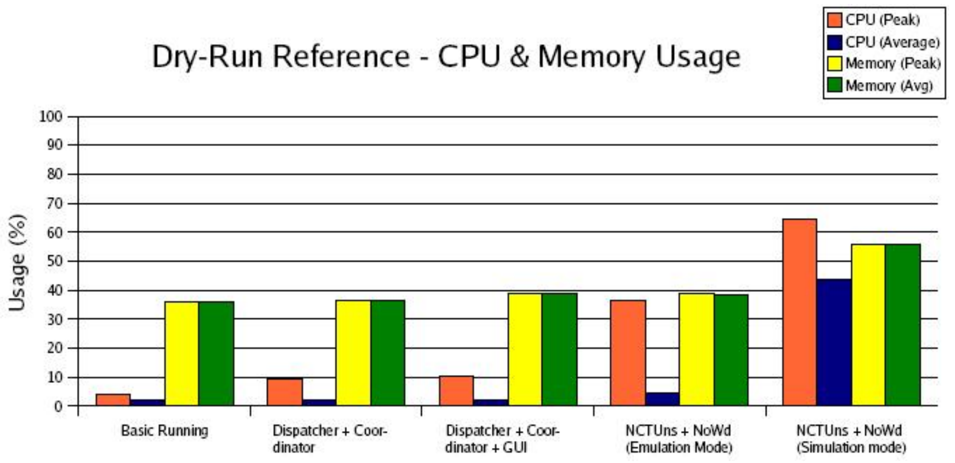

Figure 3.12: Dry-Run System Reference

As seen in [Fig. 3.12] the resource consumption is quite similar until the the execution of NoWd, with a small CPU usage and medium memory usage. When NoWd is executed we can see two different reactions: in the emulation mode the only observed difference to prior measurements is a boost in the CPU peak performance; in simulation we clearly see a good increase in all the measurements taken this is due to the very small execution time. The time execution differs from the first four scenarios to the last. While in the first four it takes 60 seconds in the last it takes only 2 seconds thus resulting in higher measured values.

3.3.3 Experiment Details and Results

The general scenario used for this case consists in the simulation and emulation of a network with a variable number of vehicular nodes, during a simulated period of time while using a single machine. Using both emulation mode and simulation mode allows the later comparison between them. In addition all nodes should be running the NoWd application and some extra traffic is to be generated.

Experiment Details

To obtain correct and proper results in acordance to the previously stated objectives some parameters and configurations must carefully be defined before the execution of the experiments.

The most important experiment parameters to define are listed below followed by a short description.

Node Number This is one of the most important parameters because it is directly related to the testbed's scalability performance. In that sense, the selected values for this parameter were: 5, 10, 20, 30, 40, 50, 60, 70, 80 and 90 nodes.

Node's Position and Movement These parameters define the spacial distribution of nodes during the experiment. The initial positions of the nodes are set to random, and the movement is characterized by an average speed of 36Km/h with a random heading. All nodes' movement is confined within an area of 1.5 Km by 1.5 Km.

Range of Communication This parameter uses the default value of 250mts.

Simulated Time This parameter is the time the experiment going to be simulated or emulated. This parameter was set to 60 seconds for all experiments.

Network Traffic With the purpose of producing more realistic network behaviour some extra traffic is to be generated by the nodes, i.e. traffic besides the regular *beaconing*. This is achieved by the generation of periodical topologically-scoped broadcast packets, configured for 5 hops.

Experiment Repetitions This a very important factor for the results credibility and accuracy. Because of time restrictions a low number of repetitions was chosen: 5 repetitions. Considering the number of repetitions the total number of experiments amounts to 100.
(1 scenario x (emulation mode + simulation mode) x 10 node numbers x 5 repetitions = 100 experiments)

Simulation Results

All data presented is the average of the results produced over 5 repetitions.

CPU Usage

Table 3.1 demonstrates the use of processing power of a machine. Peak performance is mostly at maximum while the average performance begins in a quick ascension to afterwards maintain high values.

Number of Nodes	CPU Peak Usage [%]	CPU Average Usage [%]
5	85.37	34.46
10	99.96	41.91
20	100	72.87
30	100	80.57
40	100	87.56
50	100	83.65
60	100	90.41
70	100	93.75
80	100	94.2
90	100	91.22

Table 3.1: CPU Usage results for Simulation

Memory Usage

Number of Nodes	Memory Peak Usage [%]	Memory Average Usage [%]
5	40.05	39.72
10	40.88	40.28
20	42.33	41.22
30	43.13	41.91
40	44.75	43.54
50	44.04	42.35
60	46.32	44.58
70	47.97	45.93
80	49.87	48.04
90	50.65	48.33

Table 3.2: Memory Usage results for Simulation

A quick overview of table 3.2 shows that memory usage does not variate much regarding the similarity between average and peak values. In addition the gap between the memory used by 5 nodes and by 90 nodes is not very big (only about 10%).

Execution Time

As it is evident in table 3.3 there is an increase in time consumption while the number of nodes also increases. This is logically due to the necessity of spending more time to execute a larger number of NoWd programs and network events.

Number of Nodes	Execution Time [sec]
5	1.9
10	2.46
20	11.85
30	19.48
40	29.99
50	38.84
60	51.27
70	63.48
80	75.3
90	87.06

Table 3.3: Execution Time results for Simulation

Emulation Results

CPU Usage

Number of Nodes	CPU Peak Usage [%]	CPU Average Usage [%]
5	82.91	8.51
10	85.19	10.13
20	100	21.1
30	100	31.02
40	100	42.22
50	100	64.01
60	100	81.25
70	100	94.46
80	100	93.56
90	100	95.58

Table 3.4: CPU Usage results for Emulation

By looking at table 3.4 we can conclude that despite constant high peaks, the overall CPU consumption demonstrates a gradual increase until reaching values near the maximum.

Memory Usage

The Memory usage seen in table 3.5 seems very stable due to similar values presented by peak and average. Also there is a small increase both in peak and average memory consumption everytime more nodes are added to the network.

Number of Nodes	Memory Peak Usage [%]	Memory Average Usage [%]
5	40.75	40.57
10	41.18	40.92
20	42.21	41.86
30	43.29	42.8
40	44.8	43.85
50	45.51	44.35
60	47.18	45.68
70	48.43	46.6
80	49.52	47.57
90	51.41	49.25

Table 3.5: Memory Usage results for Emulation

Execution Time

Number of Nodes	Execution Time [sec]
5	60.22
10	60.3
20	60.54
30	60.29
40	60.35
50	61.11
60	61.23
70	64.54
80	77.08
90	87.19

Table 3.6: Execution Time results for Emulation

An interesting fact presented by table 3.6 is a constant time consumption until the network reaches a number of nodes close to 70. from thereon the time consumption starts to greatly increase. It is apparent that near this number of nodes the machine's capabilities are being fully used. Therefore from that point on time constraints are very difficult to maintain.

3.3.4 Conclusions and Result Comparisons

As a final and more general analysis of the results presented in the previous section, the results of simulation and emulation will be compared on a per item basis followed by a final conclusion on overall performance of the testbed.

Comparison of Results

CPU Usage

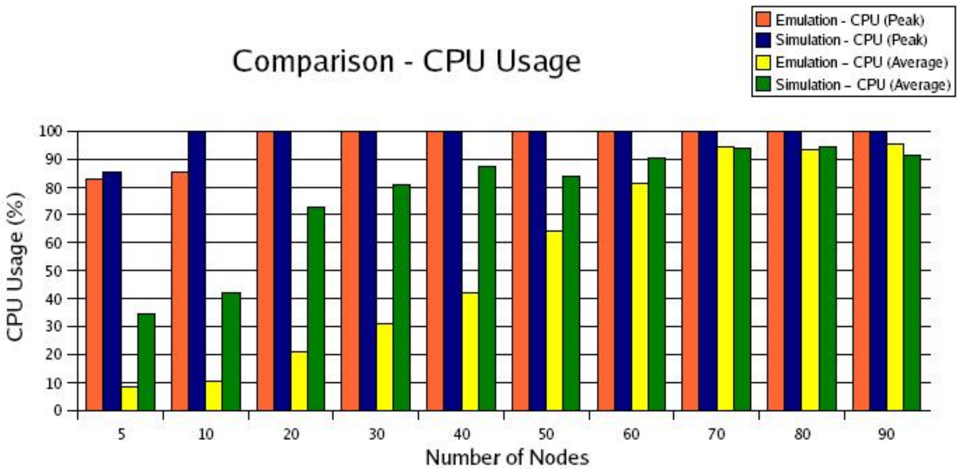

Figure 3.13: Comparison of CPU Usage

As depicted in [Fig 3.13] there is a clear difference between Emulation and Simulation resource comsumption. For the majority of experiments, especially the ones with a smaller number of nodes, emulation is lighter for the system as proved by the evolution of average values of CPU usage. As for peak CPU usage the overall performance shows a very similar behaviour presented by both methods. This happens mainly because simulation is executed in less time forcing to a less extensive but more intensive processing.

In an overall analysis we can conclude that the testbed requires a very high processing power, especially for simulation and large emulated networks.

Memory Usage

Simulation and emulation are quite similar in memory usage. Both methods use some but not much of this system resource reaching maximum values of about 50% as shown by [Fig. 3.14]. In addition the memory usage of the testbed has an overall variation of about 10% meaning that there is not much increase in this resource's comsumption. Also the similarity of peak and average values indicates a stable and small increase in the evolution of memory consumption.

In a final remark emulation proves to be slightly more costfull than simulation and the overall behaviour does not seem to be very heavy for the system.

Execution Time

The evolution of the execution time is rather special as seen in [Fig. 3.15]. Both simulation and emulation maintain a regularly expected behaviour, with simulation

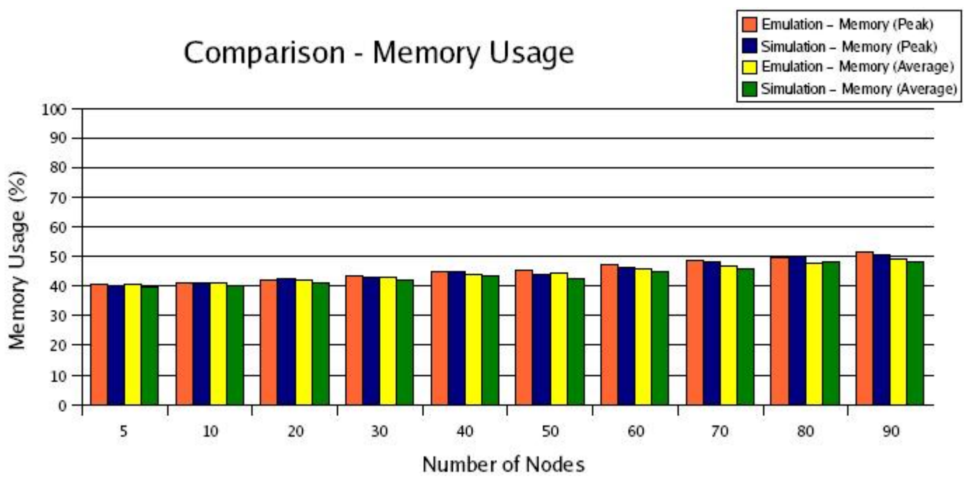

Figure 3.14: Comparison of Memory Usage

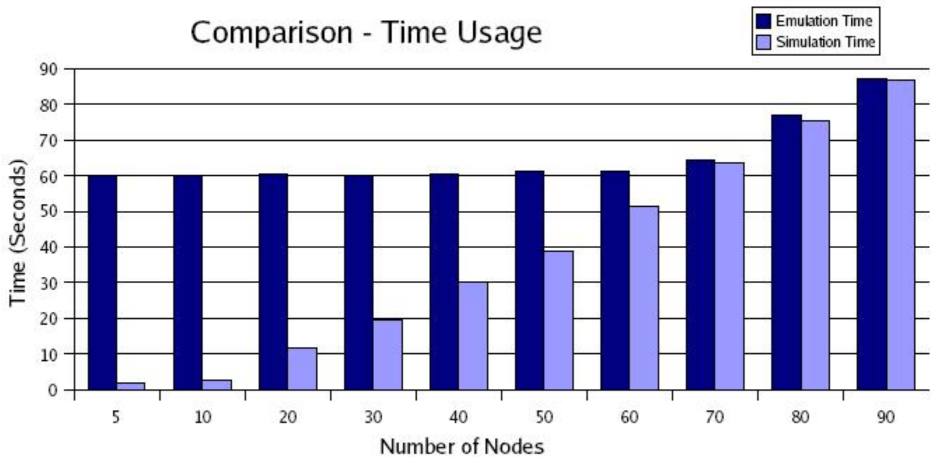

Figure 3.15: Comparison of Execution Time

proving much more time saving than emulation, until the network reaches about 70 nodes. This is a turning point in the time behaviour of simulation and emulation, because after this point execution time greatly increases in a similar way for both.

In the time point of view we can state that simulation is far better until the referred turning point. From this point onward execution time greatly exceeds the pre-defined simulation time for both simulation and emulation.

This behaviour of execution time is explained by the fact that near the turning point's number of nodes the testbed cannot respect anymore the time constraints thus causing a problem for emulation but not for simulation.

Conclusion

The analysis of the collected data allows to conclude the work on the testbed which gives us a general idea of it's performance.

The testbed is very demanding in terms of processing power which is proved by the CPU usage analysis. Furthermore with the actual system configuration the node number scalability testing has shown the system reaching it's processing limits at a number of nodes around 70, marking this as turning point in the system's performance. The more visible consequences of exceeding the turning point are: firstly to have the CPU subjected to a constant heavy load; and secondly the great increase of execution time thus causing experiments to need greater amounts of time to simulate and emulate smaller time periods.

As for memory comsumption the overall performance during the scalability tests does not affect the results in a relevant way, showing similar behabiour both for simulation and emulation, consisting mainly in small increases of usage levels.

Simulation and emulation performance analysis comparison until the turning point of 70 nodes in this testbed results in the observation that simulation is heavier for the system but lighter for execution time while the reverse is observed for emulation.

As a more general conclusion, the testbed is usefull for VANET research and development. However, scalability is an issue as only about 50 to 60 nodes can be run without any problems. In addition, time is an issue in emulation, because the execution time surpasses the simulated time demonstrating difficulties in maintaining real time restrictions.

Chapter 4

Mobility in Ad-Hoc Networks

Chapter 2 introduced concepts for Mobile Ad-Hoc Networks and Vehicular Ad-Hoc Networks in order to better understand the environment area of this thesis and the challenges involved. Chapter 3 described in detail one of the main points of this thesis, the building of a hybrid testbed for VANETs. Chapter 4 will introduce the proposed solution for the seamless mobility of ad-hoc nodes and it's deployment in a real testbed. It will also introduce the implementation of software components for the testbed integrating the Daidalos II architecture. Firstly, the Daidalos architecture will be introduced, with special emphasis on MANET components. Afterwards the implementation work will be described, followed by the description of the proposed test scenarios. Finally the evaluation of results is performed and conclusions are drawn.

The organization of this chapter is as follows:

In **Section 4.1** an introduction is made to the Daidalos II project.

In **Section 4.2** the Daidalos II project architecture is described, with special relevance to mobility and MANETs.

In **Section 4.3** is detailed the implementation of the required modules of the testbed.

Section 4.4 presents the test scenarios and the results obtained.

Section 4.5 shows some of the conclusions that can be drawn from the results.

4.1 Introduction

Nowadays, several different types of network access technologies, such as Wi-Fi, WiMAX, or UMTS, are available to the common user. This opens the possibility of a user to use different technologies in order to access the same provided services.

Although the issues of integrating different technologies are diverse, the growing need in ubiquitous access enabled the necessity of greater research for the integration of heterogenous networks.

The Daidalos II project aims to provide an architectural solution in the field of ubiquitous access [27]. This architecture would enable the users to seamlessly move between networks and different technologies, while accessing the same provided services.

Other goals of the project [27] are:

- To provide mobility management (divided in Local and Gobal domains).
- To have an identity based mobility management solution, through the independent and general management of identities.
- The integration of MANETs and NEMOs in the mobility architecture.
- To enable host multihoming, where the host owns multiple physical network interfaces and concurrently gets access through them.
- To perform integration of ubiquitous and pervasiveness concepts for customized services to the users.

4.2 Architecture

In this section a brief overview of the general Daidalos II architecture will be shown, followed by essential architectural features and a description of the MANET architecture and physical components.

4.2.1 Global Architecture

In this project, one of the main goals is to provide seamless behaviour when switching between different types of access networks, the greatest issue that comes to mind is Mobility.

How can the regular data exchange continue while switching between technologies, thus producing seamless access?

The approach to this issue is based on the simple principle of *"Divide & Conquer"*.

Therefore, in order to simplify the management of mobility, the mobility management was splitted into two levels: Local Mobility Management (LMM), and Global Mobility Management (GMM). Another feature that helps coping with mobility issues is the use of the IEEE 802.21 Media Independent Handover Functions.

4.2.2 Mobility

By dividing the mobility management into Local Mobility Management (LMM) and Global Mobility Management (GMM), it also implicitly divided the architecture into Global Mobility Domain (GMD) and Local Mobility Domain (LMD). The Global Mobility Domain is considered to be the core network, which inter-connects the several Local Mobility Domains (LMDs), thus enabling inter-LMD mobility.

This fracture in mobility management also allows that each LMD can use it's own Local Mobility Protocol (LMP), and for inter-LMD mobility a Global Mobility Protocol (GMP) assumes the command. The use of Local Mobility Domains can be even used to incorporate a single access technology that may require special mechanisms of mobility at a Local Domain level, in order to facilitate the integration of those technologies in the architecture, as shown in [Fig.4.1].

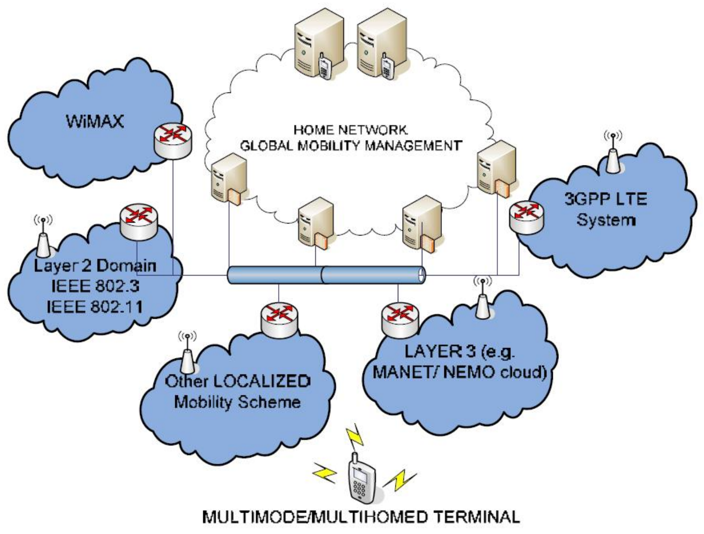
HOME NETWORK
GLOBAL MOBILITY MANAGEMENT
WiMAX
3GPP LTE System
Layer 2 Domain IEEE 802.3 IEEE 802.11
Other LOCALIZED Mobility Scheme
LAYER 3 (e.g. MANET/ NEMO cloud)
MULTIMODE/MULTIHOMED TERMINAL

Figure 4.1: Daidalos II Mobility Architecture [27]

The fact that current terminals are able to support different types of technologies emphasizes the role of the terminal in this architecture. The management of all interfaces, each with it's own specific features, becomes a challenge, especially when considering mobility between interfaces.

Therefore, the mobile terminal acquires more control over it's own process of mobility. For instance, it can solely take the decisions of moving from network and on to which network it shall move. This requires that the node is aware of where it can go according to available networks and access technologies present. Also the process of updating it's location must be triggered by himself.

4.2.3 Network-Based Local Mobility Management - NetLMM

The purpose of Network-Based Local Mobility Management (NetLMM) is to manage the node's mobility from the infrastructured network point of view. It gathers lower layer information to allow the network to keep track of a node, and reconfigure itself so that the node's mobility can be assured by the infrastructure. The node can report its location or it can be done in a passive discovery mode by the access routers.

At a Local Mobility Domain level mobility is assured by the network, while at a Global Mobility Domain level the Mobility Anchor Point must be informed of the node's movement.

The basic elements composing the structure of NetLMM are the Mobility Anchor Gateways (MAG) which are basically Access Routers that are aware of the node's movement and the Local Mobility Anchor (LMA), located between the Local and Global Mobility Domains. The LMA gathers the location of the nodes provided by the MAGs, saving a record of the location of every node in the Local Domain.

4.2.4 Media Independent Handover - IEEE 802.21

The concept behind the Media Independent Handover is to minimize the downfalls of inter-techonology handovers, thus providing the users with better mobility behaviour.

The fundamental idea is to produce an abstraction layer, that gathers link and network layer information to report it to upper layers in order to improve handover performance between heterougenous networks. This improvement's goal is to make the handover seamless to the user, by using the mobile node's capabilities to decide if, when and where to make a handover between different technologies.

The resulting abstraction layer produced is named Media Independent Handover Function (MIHF). As shown in [Fig. 4.2] the MIHF is a 2.5 layer approach, that works as a translator between the upper management layers and the several different and specific drivers. There are some features that are provided by the MIHF:

- Event Service

  Translates driver related events into meaningfull management triggers.

- Command Service

  Translates generic management triggers into driver specific commands.

- Information Service

  Enables the exchange of information so that decisions like handovers can be taken more efficiently.

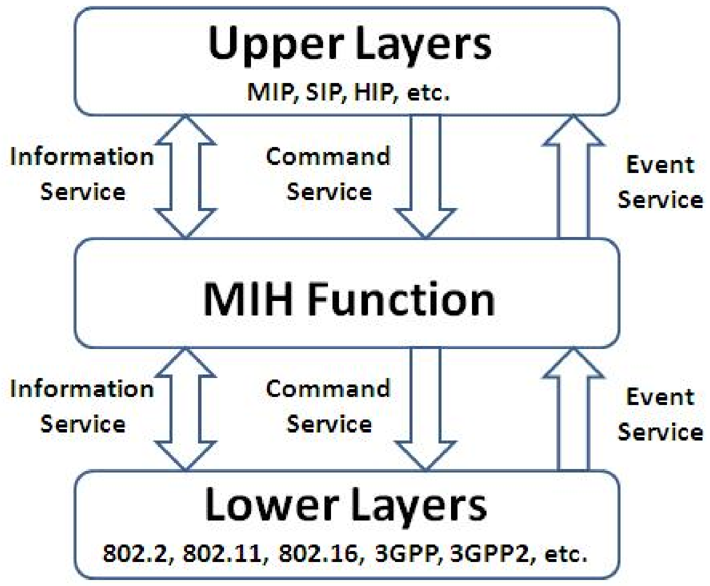

Figure 4.2: IEEE 802.21 - Media Independent Handover Function [27]

4.2.5 Additional Features

Besides the core features of the project, there are other features that emerge from it. Some are requirements for specific technologies while others are just implicit by the nature of the project. All these features help to enrich and enlarge the number of possible applications.

- Multihoming

  Multihoming is an inherent concept when the Mobile Terminal is equipped with different access technology interfaces. As mentioned before the architecture supports multihoming, in both Global Mobility and Local Mobility. The Mobile Terminal's Correspondent Node will however only be aware of Global level multihoming, since Local Mobility is seamless.

- Bi-directional load balancing.

  Closely related to multihoming, load balancing is also a very interesting feature. Usually the mobile terminal is using multihoming with different technologies and therefore the load balancing must be done at a Global Level (assuming that the Local mobility Domains are technology-driven).

- Mobility for quality of service sessions.

  Quality-of-Service (QoS) is a crucial factor for several technologies. Thus the mobility of bandwith reserves must be assured.

- Mobility for multicast sessions

  Many of the services provided by upper layers use multicast . In a mobility architecture such as this one, these sessions must be maintained in order not to loose vital information associated with those services.

4.2.6 Mobile Ad-hoc Network Architecture

Mobile Ad-hoc Networks (MANETs), in the Daidalos II project are usually composed by multiple nodes connected in multi-hop fashion and supported by Access Routers (MAGs) that provide comunication with the infrastructured network. The infrastructured network and the ad-hoc multi-hop network constitute the Local Mobility Domain and the infrastructured network assures conectivity with the Global Mobility Domain.

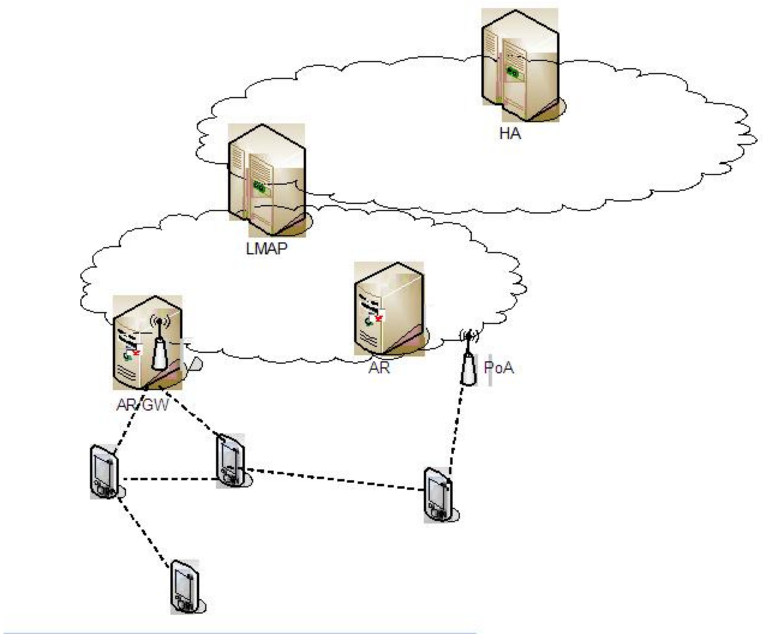

Figure 4.3: Example of MANET Architecture in Daidalos II [27]

Architectural Challenges

- *Network Layer and Local Mobility*

  In regular Wi-Fi networks the node is always at one-hop distance from the MAG. This means that the detection of nodes arriving or nodes that left is quite easily done by the MIPv6 protocol discovery in the MAG. However, MANETs are multi-hop wireless networks, which implies that nodes attaching to the network at more than one-hop distance, cannot be detected by the MAG.

  The approach taken in order to solve this issue consists in a simple adaptation of the MANET work flow. Every node arriving to a MANET is now required to report it's location by sending a message to the MAG, thus triggering the LMP, and consequently the node's bootstrap procedure.

- *IEEE 802.21 and Network Layer*

  The IEEE 802.21 was designed to work in networks with direct infrastructure support, i.e., networks in which the node is at distance of a single-hop from the MAG/AR, and under these circumstances the MIHF message transportation scheme can work, since it is based in MAC addresses. For MANETs this poses an issue because their multi-hop architecture requires a messaging scheme based on network layer and IP addresses.

  To overcome this issue the MANET is declared as a different technology that cannot use MAC addresses but rather uses only IP addresses, thus maintaining 802.21 communication.

- *IEEE 802.21 Events*

  With the previous issue solved a new issue surfaced in it's place. The fact that the MANETs work only on Network Layer based addresses, caused an issue in the Event Service of 802.21. The MIHF receives Link Layer events directly from the technology drivers, events like the loss of a link that now are meaningless. Meaningfull information is now generated in the Network Layer, like the loss of routes.

  The solution for this issue is the creation of a module for the Mobile Terminal that gathers Link and Network Layer information from the routing and auto-configuration protocols generating meaningfull 802.21 events. This module, named MANET Wrapper will produce an abstraction layer that will act as if the node is at a one-hop distance of the MAG.

Mobile Node

Here is a simplified representation of the mobility-related modules in the Mobile Terminal and below in [Fig. 4.4] is grahically demonstrated the architectural organization of the modules in the Mobile Terminal.

- MANET WRAPPER

  The MANET WRAPPER is the core and most important component for enabling MANET node mobility. In the Daidalos architecture the MANET technology works in resemblance to the WLAN technology, because they both share the same lower layer resources (such as driver, wireless interface, etc...) and consequently have (in part) the same behaviour. Despite their similarities, the MANET technology has to deal with a more wider range of scenarios and issues than WLAN, namely multi-hop situations. Thus, the Manet Wrapper is introduced in the architecture, to cope with ad-hoc requirements that go beyond the capacity of WLAN. The presence of this module avoids deep changes to the mobility architecture by acting as a direct link emulator between the point of access and the mobile node, therefore making the MANET technology behave as if it was WLAN technology and look seamless from the architecture point of view.

- RPSC (Routing Protocol Selector and Controller)

  The routing protocol selector and controller works as a manager of ad-hoc routing protocols, and the interfaces where they can be run. Decisions on the concurring access of the protocols to interfaces are made based on the interface's attached point of access information.

- MAGINFO/AUTOCONF

  the Mobile Anchor Gateway Information module provides support for the advertisement of available Ad-Hoc gateways and networks to the mobile node, allowing the node to discover which Ad-Hoc gateways are available in which network. It also implements the Ad-Hoc auto-configuration protocol, which is vital for starting the node's communication with the gateways.

- Multihoming

  This module provides multihoming support for MIPv6.

- PANA Client (PAC)

  This module manages the terminal's authentication by means of an authentication protocol which is crutial to allow access beyond the gateways and into other networks such as the Internet.

- MTC

  This module controls mobility in the terminal, including a VID (Virtual Identity) Authenticator to authenticate the virtual user, and the IIS (Intelligent Interface Selection) to choose the interface to connect to.

- MIHF

  This module provides an abstraction layer for the underlying access technologies by using the IEEE 802.21 protocol to communicate with them (see section 4.2.4).

- VIP (Virtual Interface Proxy)

  The virtual interface proxy is a module that aims to provide privacy to the node's communication. Its job is to map real interfaces into especially created virtual interfaces, based on privacy information, such as the VIDs (Virtual Identities). Virtual interfaces are used as if they were real interfaces allowing users to use a different interface based on privacy criteria. The network will then have a perception of users, instead of nodes, even if several users are located in the same node.

- WLAN-RAL (Radio Abstraction Layer)

  The WLAN Radio Abstraction Layer, aims to provide an abstracted interface of the WLAN driver for communication with 802.21 messages. This module then controles directly the WLAN driver and consequently manages the wireless interface. The RAL-WLAN interface acts basically as a translator of 802.21 commands in low-level interface procedures and 802.11 events in meaningfull 802.21 events.

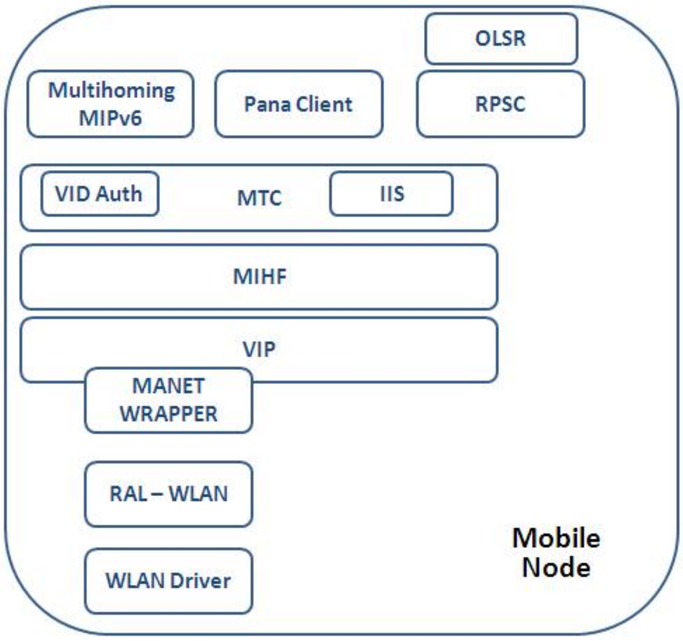

Figure 4.4: Mobile Node Architecture [27]

Mobility Anchor Gateway

The architectural structure of the MAG in the Access Routers (AR) is similar to terminal. Although similar still some important differences exist. For instance, the MAG needs Local Mobility Protocol support and does not require any modules related with the management of VIDs, such as, the Virtual Interface Proxy and the Virtual Identity Authenticator.

4.2.7 Ad-Hoc Bootstrapping

The process of bootstrap consists in following the required steps that culminate in granting the mobile node with global connectivity, i.e., obtaining access beyond the MAG and more important, beyond the LMA.

Altough with different goals, the bootstrap and the following handover are in part very similar to each other.

Getting connectivity with Point-of-Access (PoA)

The first step is to obtain conectivity with the Point of Access and the MAG. In WLAN this is implicit, since any node is at one-hop distance from a Point of Access. But in a MANET a node might be at several hops from the Access Point, thus the use of MAC and IPv6 Link-Local addresses like in WLAN is useless, and the node does not yet possess a global IPv6 adress. Also regular routing cannot be used at this early stage of bootstrap, because the node is not authenticated with the network and therefore its traffic will not be forwarded. In fact, the Access Routers drop any packet that is not originated from an authenticated node.

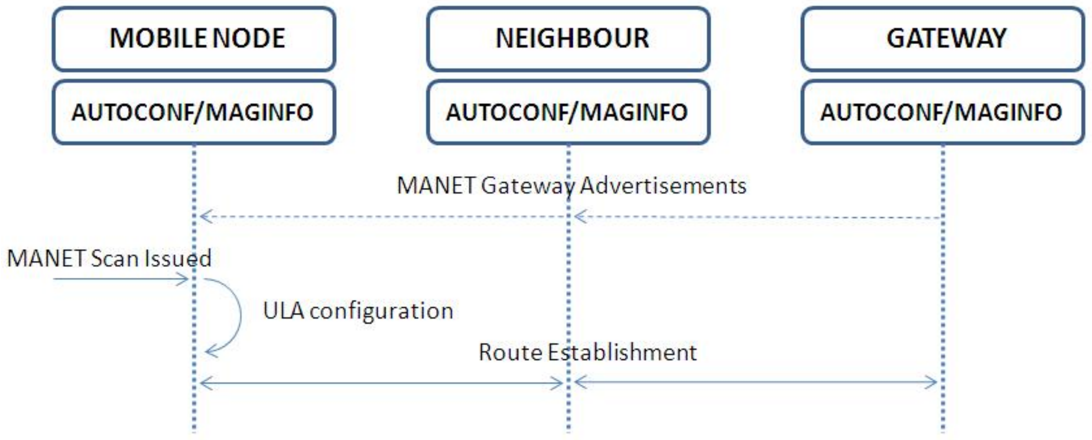

Figure 4.5: Getting access to the Gateway.

The solution to this obstacle is provided by the Auto-Configuration module (MAGINFO). When a scan for ad-hoc networks is issued, this module will create a temporary (during bootstrap) MANET-scoped address (only works inside the MANET), the ULA (Unique Local IPv6 Unicast Address [27]), as depicted in [Fig.4.5]. Now with this address, and the help of a simple routing algorithm being executed by the autoconfiguration module in each node, a search will be executed to locate any ad-hoc supporting MAGs. For this search the node will be listening for MAGINFO GW advertisements and if there are any MAGs supporting ad-hoc technology MAGINFO will establish a bi-directional route between the node and the MAG.

Authentication & Address Configuration

Once this route is set the node can start the authentication procedure by sending authentication data to the MAG, as shown in [Fig.4.6]. If the data is valid then the MAG will first register the node's location at the LMA and afterwards reply the node with a MIPv6 Router Advertisement that upon reception will trigger the MIPv6 auto-configuration and generate a valid Care-of-Adress (CoA) from the prefix advertised.

Upon configuring a valid CoA the node can remove the ULA and start the MANET routing protocol, that will ensure connectivity troughout the MANET and reach the MAGs that will now forward the traffic to other networks when required.

If the node is not in it's home network, the node's MIPv6 will carry out a binding update notification towards the Home Agent. And to finalize the bootstrap the node's MIHF will register itselt in the MAG in order to receive future information on events, such as networks to where handovers are possible.

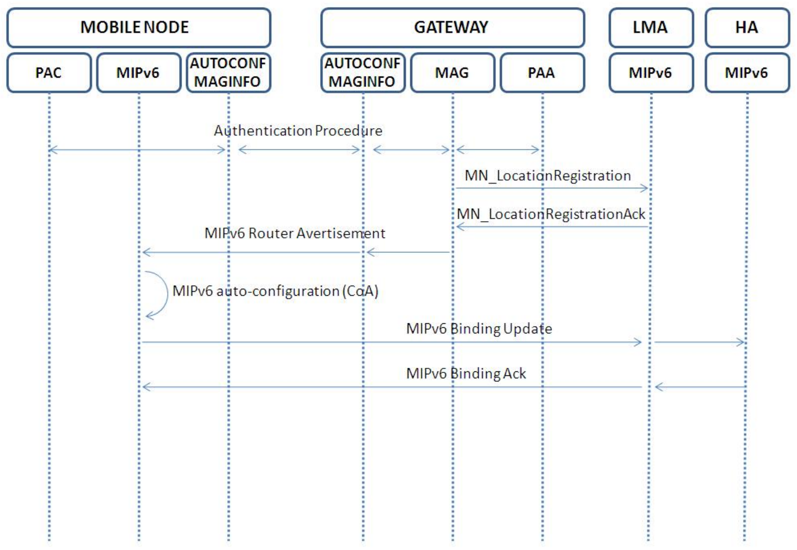

Figure 4.6: Authentication & address configuration procedures.

4.2.8 Handover

By introducing the wrapper (see section 4.2.6), the ad-hoc handover operations are managed in a seamless way as in the infrastructure. If a new 802.11 PoA is selected, three different handover scenarios are possible. On the first one, the node is connected to infrastructure and requests a handover to an ad-hoc PoA. On the second one, the MN is part of a MANET and connects to infrastructure. The third scenario is the one where the node is requesting a handover from one ad-hoc PoA to another.

In the scope of this thesis we will focus on the first scenario, but some procedures concerning other scenarios will also be addressed.

Scanning for Candidate Networks

Before performing any handover, it is needed to know where to perform it. Traditionally, this information is obtained by performing a network scan, which retrieves the list of surrounding APs (wireless networks) that the mobile node (MN) can reach. For the normal infrastructure mode, this list is enough; however, in an ad-hoc environment, the list of available networks does not guarantee that the networks have a reachable gateway (GW) that can provide access to the operator network. To provide the desired information, the obtained network list needs to be validated, making the candidate discovery process a two phase one.

The first phase is the scan for surrounding networks, with a normal 802.11 scan, as shown by [Fig.4.7]. In the second phase, the results corresponding to ad-hoc networks will be validated. To perform validation, the mobile node has to associate to each ad-hoc network found and receive an auto-configuration message. Receiving one of these messages will ensure that the MANET is in fact connected to one gateway. In addition, the auto- configuration message will also have information about the gateway, such as its address, which is needed later to perform handover. Since the handover candidate discovery is also a part of the 802.21 protocol, this task will also be performed by the MANET Wrapper. Upon reception of a Scan Request command sent by the MIHF, the MANET Wrapper will take control of the process, first performing the network scan, and then validating the results. Once the validation of all ad-hoc networks found is done, the MANET Wrapper will issue an 802.21 event towards the MIHF, one for each network, containing the MANET relevant information.

Candidate discovery issues: This candidate discovery process has a disadvantage. The scan, and subsequent association to each network for performing its validation, will interfere with the ongoing communications of the node, since the WLAN card will be busy validating the network. One way to mitigate this problem is the use of two wireless cards, one for normal communications, and other for performing scans. If two wireless interfaces are not available, then some performance drawbacks have to be expected when the candidate discovery is in progress.

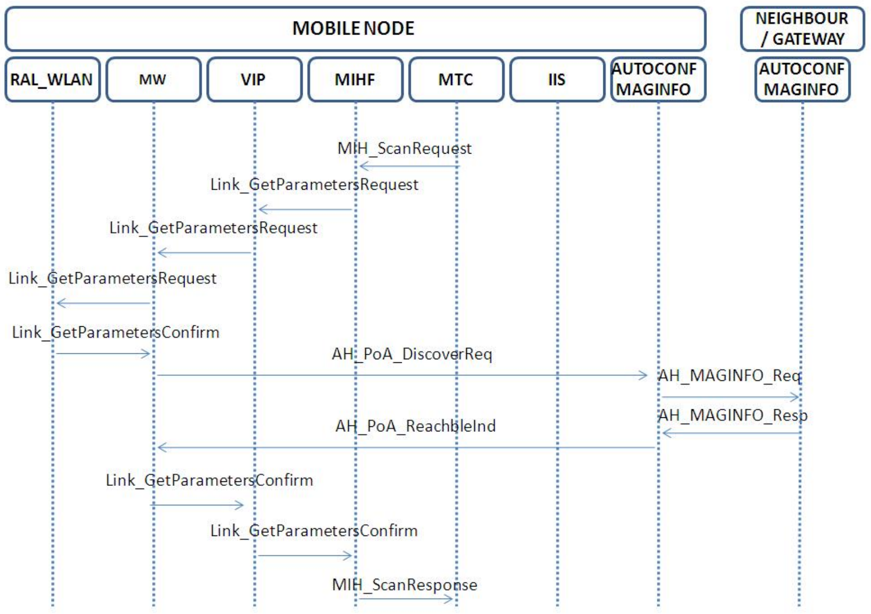

Figure 4.7: Network Scan and Validation Scan [27]

Triggering the mobility process

The usage of the IEEE 802.21 protocol and its support by the MANET will bring the necessary elements to support the local mobility protocol, namely the mobile node's presence detection. In addition, 802.21 can be used to provide the signalling needed to maintain quality of service information during handovers. These 802.21 messages can then be used by the access routers and gateways in order to detect new nodes in the network, and trigger the local mobility protocol.

This mobility process also supports make-before-break handover, where the handover destination AR is informed about a new node wishing to attach to this network, before its movement. This type of handover is supported in this local mobility architecture due to the usage of the IEEE 802.21 protocol, which will provide messages for this process.

Handover Process - MANET to WLAN

The MIHF in the mobile node forwards the MIH_Handover_InitiateRequest to the current gateway's (cGW) MIHF, which forwards it to a QoS-Broker (a resources manager in the access network) to get permission to perform the switch. An answer is sent back to the MN using the same path as the request message, as seen in [Fig.4.8]. The MN then issues a MIH_Handover_CommitRequest signalling that the resources in the new network should be allocated, and instructs the new access router (nAR) to activate the already allocated resources. mobile node then switches to the new SSID and channel, and sends a

MIH_Link_Up_Indication message to the nAR. This event reaches the LMP Engine in the nAR which uses the LocationRegistration message to register the new terminal's location with the LMA. The answer triggers a RouterAdvertisment (RA) from the LMP Engine towards the mobile node. The mobile node then triggers the MIPv6 in order to register the new address with the Home Agent. Then, a MIH_Handover_CompleteRequest is sent to free the resources in the ad-hoc network. The last step in the process is the de-registration of the old Care-of Address (CoA) issued by the LMA Engine to the ad-hoc gateway which confirms it.

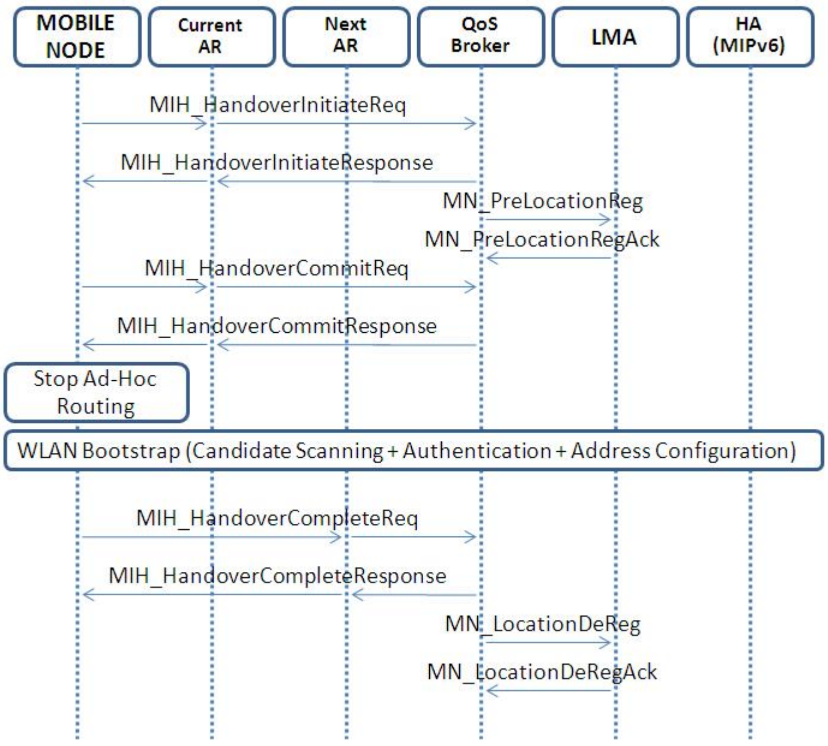

Figure 4.8: Handover from MANET to WLAN.

Handover Process - WLAN to MANET

This scenario uses a similar MIH signalling between the mobile node and the network entities, but differs on the way the MIH_Link_Switch messages get processed. Besides connecting to the link layer (L2) ad-hoc network, MANET Wrapper also signals the auto-configuration module to configure an ULA address and activates the bootstrap routing protocol, so that mobile can reach the gateway to get authenticated, send the Link- Up message and receive the RA message. The router advertisement message will be addressed to ULA, instead of the link-local address.

The MANET Wrapper acts as a Radio Access Layer (RAL) for both ad-hoc and 802.11 infrastructure PoAs that fully implements MIH-LINK- SAP interface. The

MIH_CommitRequest/Response phase precedes the MIH_Link_Switch to the new point-of-access (see [Fig.4.9]), and after handover is complete, the old link is switched off. The mobile node is aware of the type of handover (inter or intra-technology), and generates the correct message sequence accordingly. Notice that ULA is only required in handover process if the mobile node cannot know its new address previously to handover. However, with pre-authentication schemes, the mobile node can be previously authenticated in the new network and get the new address before moving.

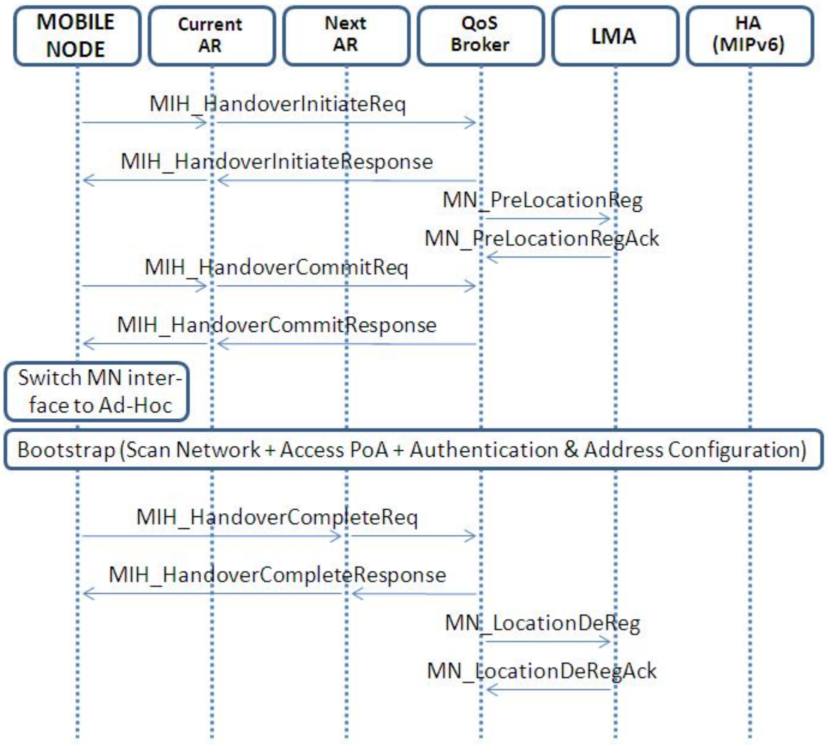

Figure 4.9: Handover from WLAN to MANET.

Handover Process - Signaling

Inside the mobile node, MTC controls the handover operation by interacting with MIHF, IIS and VID authenticator. A handover can be triggered to fullfill user preferences and/or in response to a link degradation. In a typical handover operation, MTC starts the process by issuing a ScanRequest to MIHF, to get the available PoAs (see [Fig.4.7]). MIHF generates a LinkGetParametersRequest message and passes it to MANET Wrapper (MW) via the Virtual Interface Proxy(VIP). The Manet Wrapper sends the message to Radio Access Layer (RAL-WLAN in this case), that returns a LinkGetParametersResponse with the information about both ad-hoc and infrastructure link layer (L2) 802.11 networks in range.

Given that not all the ad-hoc link layer (L2) networks have a PoA attached, and it is also impossible to know the hop-count just by looking at link layer information, the Manet Wrapper must then gather more information with the help of MAGINFO/AUTOCONF. To do so, the latter will issue a AH_PoADiscovery message and wait for a AH_PoAReachableIndication response for each available ad-hoc PoA (see [Fig.4.10]). The message also carries information about the gateway ID and the link quality and metrics. When all the ad-hoc L2 networks are scanned, the Manet Wrapper will then answer to the LinkGetParameters request with a LinkGetParametersReply, that will inform MTC about both infrastructure and ad-hoc PoAs. The MTC then forwards that information to IIS via a IIS_Init_Trigger message, and receives the new PoA assignments on a IIS_Report_Indication message, as seen in [Fig.4.10].

If a new PoA is assigned to an interface by the IIS, the MTC manages this process by sending MIH commands to MIHF via MIH-SAP.

MTC starts by sending a HandoverInitiateRequest to MIHF, that will forward the message to the current AR and receives a HandoverInitiateResponse from it. MTC then triggers authentication by sending a AuthorizationResquest message to the VID authenticator. VID authenticator will interact with PAC. If the node authorization request on the new PoA is accepted, PAC will return the mobile node's care-of-address (CoA), that is then forwarded to VID Authenticator and then to the MTC, on the AuthorizationResponse message.

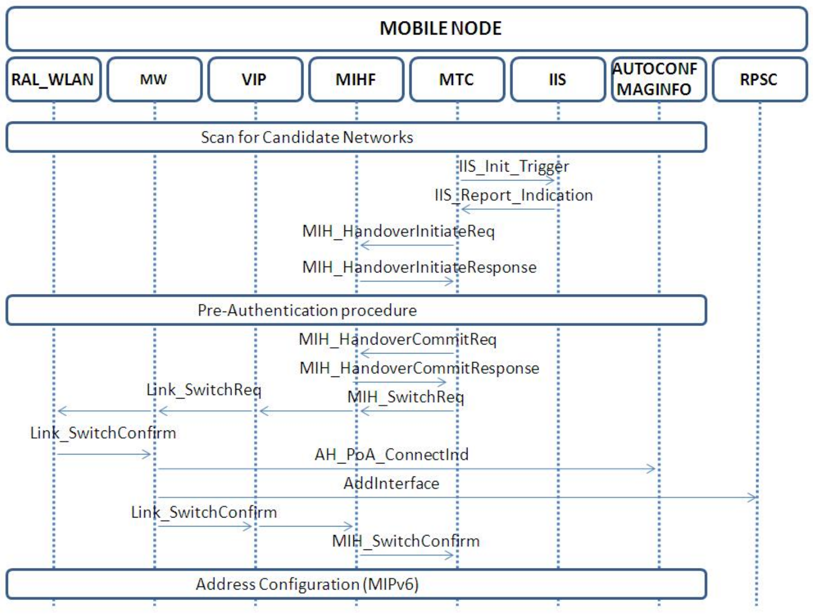

Figure 4.10: Handover from WLAN to MANET - Mobile None's Internal Signalling.

The MTC will then commit the handover by issuing a HandoverCommitRequest to MIHF that will send the message to the current AR, that in turn will send a Handover-CommitResponse. If all this message exchange was successful, MTC will then command the MIHF to switch to the new network, with a SwitchRequest message. MIHF will then send a LinkSwitchRequest that is forwarded by VIP to the correct RAL. In our case, the Manet Wrapper will be acting as a proxy of the real RAL. If the handover is to ad- hoc, Manet Wrapper will request a switch to the corresponding ad-hoc layer 2 network by sending LinkSwitchRequest to the RAL- WLAN that will answer with a LinkSwitchConfirm message, indicating the result of the operation (see [Fig.4.10]).

On success, Manet Wrapper will then start the ad-hoc routing protocol on the selected interface and inform MAGINFO/AUTOCONF about the connection to a new PoA. MAGINFO/AUTOCONF will then start sending Gateway Route Reply (GRREP) messages to the new gateway, signalling the connection of a new mobile node to it. Internally to the gateway, MAGINFO/AUTOCONF sends a LinkUpIndication to the local MIHF that will trigger location registration on LMA. LMP engine on the gateway will send the prefix to the node's ULA via a RouterAdvertisement message that is processed by the node's kernel and MIPv6 daemon.

MTC finally sends a HandoverCompleteRequest to the new access router via the node's MIHF. When MIHF receives the HandoverCompleteResponse, the handover phase is over and the mobile node is fully connected and authenticated on the new PoA.

4.3 Implementation

In the scope of this thesis the most relevant project capability is the seamless support of nodes moving between infrastructured networks and ad-hoc networks, maintaining access to the Internet with similar quality.

The main goal of this part of the thesis, resides in the MANET architecture, especially in the possibility of bootstrapping a Mobile Terminal in an Ad-hoc domain, and also the handover from infrastructured WLAN to Ad-Hoc.

In implementation terms, this means the development of the required MANET software components, in order to assemble a simple testbed where the previously mentioned goals in the thesis' scope can be tested and measured.

In fact, the software components to be implemented in the mobile terminal (MT) and access router(AR) / gateway(GW) are the following:

- `A24_MT_OLSR` (including also the OLSR routing protocol deamon), for the Mobile Terminal.

  The `A24_AR_OLSR` module is identical to `A24_MT_OLSR`, however it is used in the access router/gateway.

- `A24_MT_RPSC` (Routing Protocol Selector and Controller), for the Mobile Terminal.

  The `A24_AR_RPSC` module is identical to `A24_MT_RPSC`, however it is used in the access router/gateway.

- `A24_MT_MW` (MANET Wrapper), for the Mobile Terminal.

  The `A24_AR_MW` module is a much more lighter and smaller version of `A24_MT_MW`, designed to be used in the access router/gateway.

Note that other modules such as the MAGINFO/AUTOCONF will not be described here due to being responsability of other partners of the project. These partner modules will however be integrated with the previously presented modules in this testbed.

4.3.1 OLSR

The Optimized Link State Routing Protocol [13] is a very popular routing protocol for MANETs. However there was not nearly enough time to produce a dedicated version of the protocol. Thus a small research was done in order to find a suitable open source application.

Requirements: the application should be easy and simple to use and configure. Also it should work in IPv6 and cope with dinamic addressing and it should have some QoS mechanism.

`OLSRDeamon`

After some research the `OLSRDeamon` [5] (version 0.5.3) was the choice made. It complies with all requisites, it is quite flexible and easy to use. However, in order to integrate it with the Daidalos architecture some modifications had to be done. But instead of modifying a very complex code, it was decided that it was easier to just develop a simple controler module for the `OLSRDeamon`.

`A24_MT_OLSR / A24_AR_OLSR`

This module is just a way to control the regular execution of the `OLSRDeamon`. It receives commands as packets through a Unix socket to Activate and Deactivate the protocol on a specific interface, thus keeping track if an interface is already running the protocol or not. Basically, it works as an interface manager for `olsrd`, making routine checks, ensuring that the way `olsrd` is executed complies with the Daidalos requirements.

Work Flow

Upon reception of a request to `Activate` an `olsrd` instance, some routine checks are done. For instance, if the interface index is valid and if the interface is already running an instance of `olsrd`. If all checks are ok, then there is a query for the interface name because `olsrd` needs the interface name not the index.

Afterwards, a fork is done for executing `olsrd` in IPv6 mode to a specific interface in order to not abort the execution of `A24_MT_MW`. The process id of the child process is saved for later tracking for termination.

Upon reception of a request to `Deactivate`, an `olsrd` instance does some similar routine checks to the ones of enabling. Then the process id of the child is retrieved and a kill command is issued with that process id.

Tracking the proccess id was the solution found to cope with the necessity of a way to link a specific instance of `olsrd` to an interface, for later termination.

Upon reception of a request to `SetInterface`, the interface index is validated. Afterwards there is a query for the interface name which is stored.

Interface with `A24_MT_RPSC / A24_AR_RPSC`

There are three main primitives, all started by the RPSC module:

- `Activate (interface_index)`

  This primitive allows to initiate an `olsrd` instance in the selected interface index, if free.

- `Deactivate (interface_index)`

  This primitive allows to terminate an `olsrd` instance in the selected interface index, if existent.

- `SetInterface (interface_index)`

  This primitive allows to save the interface name for later use.

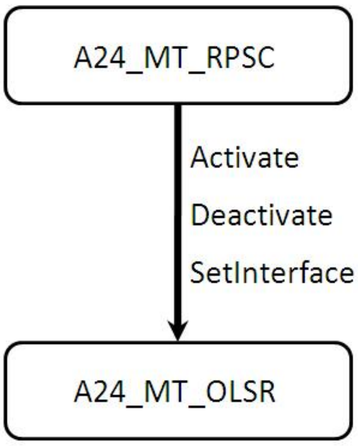

Figure 4.11: A24_MT_OLSR / A24_AR_OLSR module interface.

4.3.2 RPSC

The Routing Protocol Selector and Controller is a protocol and interface manager that controls concurrent access from the protocols to the interfaces. It allows the use of multiple routing protocols in MANETs, such as OLSR [13], MMARP [25], AODV [23], etc. Its job is to keep track of which protocol is running in which interface and ensure that no incompatible routing protocols are running in the same interface.

`A24_MT_RPSC / A24_AR_RPSC`

It receives commands as packets through a Unix socket to add an interface, delete an interface, to change the routing protocol that runs in an interface and to change the IPv6 address of the interface if required.

Work Flow

In the essence this module is a list of structures associated to real interfaces (indexed by the interface number), since each interface index works as a clear identifier. These structures store information about the interface, namely if there is a routing protocol running in the interface, and the IP address if it exists.

The behaviour of this module is very similar to the previous one (OLSR). Actions are taken, taking into account the list of interfaces and their internal state. When a message is received, the state of the interface is always checked in order to decide if the request should be executed or if an error should be reported.

When the module receives an `AddInterface` message, the state of the interface is checked, and if it already has a routing protocol running, an error is reported; otherwise, an `Activate` message is sent to the appropriate protocol module and the interface's state is changed. The primitive `DelInterface` has a similar procedure, the difference resides in the chosen action, the error is reported in case no protocol is running in the interface, while otherwise a `Deactivate` message is sent to the appropriate protocol module and the interface's state is changed.

If a `ChangeRP` message is received, there are different procedures to take depending on the state of the interface. If the interface is clear, then it will behave as if it received an `AddInterface`, but if the interface is not clear, then it will firstly start the same procedure as that produced by `DelInterface`, and afterwards it will run the procedure as if it received an `AddInterface` message.

The procedure to be executed when a `ChangeIPAddr` primitive is activated is simple. Its starts by checking the interface for IPv6 addresses, then a match is made between the addresses in the interface and the one on the message. If a match is found, then the situation is reported; otherwise, the interface is cleared of addresses (exception made to the link local address), and the new address is added to the interface.

Interface with OLSR

There are three main primitives, all started by the RPSC module, and described earlier, in the implementation of OLSR (see section 4.3.1).

Interface with MANET WRAPPER

There are four main primitives, all started by the MANET Wrapper module:

- `AddInterface (interface_index, interface_address, routing_protocol)`

  This primitive registers the routing protocol running in the interface, and takes necessary measures to start the requested routing protocol in the designated interface.

- `DelInterface (interface_index)`

  This primitive allows to terminate an existing routing protocol in the designated interface.

- `ChangeIPAddr (interface_index, interface_address)`

  This primitive changes the IP address (IPv6), or adds it if none is present in the designated interface.

- `ChangeRP (interface_index, routing_protocol)`

  This primitive will terminate any running routing protocols on that interface and then will start the received routing protocol in the same interface.

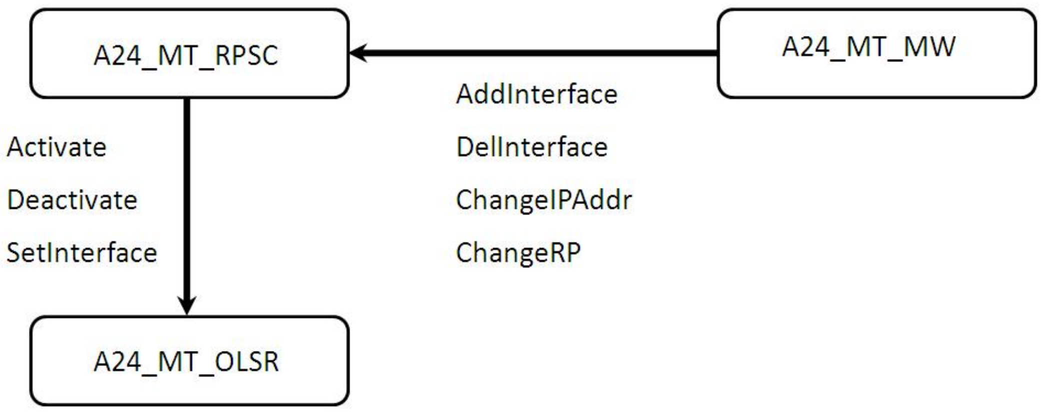

Figure 4.12: A24_MT_RPSC / A24_AR_RPSC module interface.

4.3.3 MANET WRAPPER

As discussed before (see section 4.2.6) the MANET Wrapper is the core component of the ad-hoc architecture in the project. The MANET Wrapper uses not only information and operations provided by ad-hoc auto-configuration and routing protocols, but also from the 802.11 link layer (L2) events provided by the driver. Because the same card can be used for infrastructure and ad-hoc connectivity, the MANET module presented to MIHF (see section 4.2.4) must keep the 802.11 functionalities of the original 802.11 module (WLAN-RAL, see section 4.2.6), and extend them with the ad-hoc information.

The MANET Wrapper enables the transparent use of WLAN-RAL in ad-hoc and infrastructure mode; it communicates with both 802.11 RAL-WLAN and ad-hoc modules (namely the MAGINFO/AUTOCONF and RPSC), and generates meaningful messages to the MIHF [Fig.4.13]. The events are processed by the Wrapper in a way that they have relevance in the context of the MANET. Commands from the MIHF can trigger operations on the MANET modules and on the 802.11 RAL as well. Thus, MANET is supported at a lower level inside the architecture, and the upper layers will interact with the MANET using the abstract interface (MIHF) they use for the other technologies.

The MANET wrapper introduces the ad-hoc PoAs to the framework. The metrics that characterize the quality of the PoA will relate to the link between the mobile node and the gateway. Information about the link is provided by an ad-hoc gateway advertisement and auto- configuration protocol (in the MAGINFO/AUTOCONF) that informs the mobile node about the connectivity towards the gateway, and provides a metric that characterizes the path to it, that is updated hop- by-hop.

`A24_MT_MW`

As the previous modules, the communication with this module is also done through the use of Unix sockets. But unlike the other implemented modules, the Manet Wrapper has four of these sockets. Three of these are permanently open to receive and send messages, one to communicate with the VIP (which communicates with MIHF), another to communicate with the RAL-WLAN, and the last one to communicate with the AUTOCONF/MAGINFO module. The third socket is only temporarily open when required to send messages to the RPSC module.

The interfaces with MIHF and RAL-WLAN use communication that in its essence is 802.21 messaging, and consequently have a large set of primitives that will not be thoroughly explained here, the exception is made for some of them that will be mentioned and are crutial for understanding the work flow of the module. Other interfaces have their own set of primitives.

The core idea of emulating a virtual single-hopped link between the mobile node and the PoA while filtering possible received information that might be relevant, transformed the Manet Wrapper module in a ad-hoc filter associated with a PoA Manager and an ad-hoc routing controller in between of MIHF (via VIP) and the RAL-WLAN.

`A24_AR_MW`

This is a much lighter version of the mobile node's version, however is very alike with the mobile node's version in terms of interfacing, apart the fact that there is no connection with the AUTOCONF/MAGINFO module here.

While the mobile node version must deal with and analyse 802.21 events and also manage the PoAs and ad-hoc routing, the responsabilities of this module in the access router/gateway are quite less, mainly because there is no necessity of managing PoAs here, as well as no need to filter the events. In fact, the only requirement for this module is to start-up and control the ad-hoc routing.

Work Flow

Being the core module of the ad-hoc architecture, the module comprises a certain degree of complexity, and instead of detailing the process based on the messages received, the behaviour of the module will be divided in main procedures and then explained separately.

The main processes that compose the module are:

- Advertising
- Scanning
- Switching

Advertising is the first procedure to be executed because it is triggered by the message `Link_AdvertisementIndication` from the RAL-WLAN, which is usually the first message to be received by the MANET Wrapper. The received message is copied and forwarded to the VIP module. The copy of the message is altered in order to create an announcement of the MANET technology. This transformation consists in just two small changes: the first one is a change on the interface identifier (which is the MAC address extended with zeros until eight bytes - XX:XX:XX:XX:XX:XX:00:00), the substitution of the last byte from 00 to 01 (resulting XX:XX:XX:XX:XX:XX:00:01); the second change is in the technology identifier, wich is usually 19 (WLAN technology) and is replaced with 20 (MANET Technology). After producing these changes, the copied message is forwarded to the VIP module.

The reason for these changes is that, regardless of depending on WLAN, the MANET is recognized by upper layers as an independent technology and therefore needs to be announced like any other technology: with a `Link_AdvertisementIndication`. Also, in order to not confuse upper layers with the same interface identifier (which has eight bytes by default and is largely used for searches and matching), a new one is fabricated to avoid any conflict.

Scanning is the most important procedure in the work flow of this module. It starts when a `Link_GetParametersRequest` message arrives. At this point there are four possible scenarios: the message requests for a general WLAN scan (1); the message requests a general MANET scan (2); the message orders a specific scan for a WLAN PoA (3); the message asks for a specific scan of a MANET PoA (4).

In case (1) the only matter to be attended is to clean any MANET PoAs that may appear in the `Link_GetParametersConfirm` message afterwards produced by the RAL-WLAN. This is required to be done because the RAL-WLAN scan doesn't make a distinction regarding the technology asked for in the scan and just returns results regardless of technology. Thus, the MANET Wrapper needs to clean these results for upper layers.

Regarding case (3) the PoA requested for scanning is copied in order to be matched with the results presented in the `Link_GetParametersConfirm` sent by RAL-WLAN. If there is a PoA match, then all remaining PoAs except the match are cleaned, otherwise all PoAs in the message are cleaned and the message is forwarded to the VIP as if no PoAs were found.

As for cases (2) and (4), they are much more complex and involve other modules. For instance, case (2) requires that upon reception of the `Link_GetParametersConfirm` message, besides cleaning the message of any WLAN PoAs, it also requires the validation scan to be performed, and only then the message is passed to the VIP module.

The **validation scan** is performed after cleaning the PoAs received in the `Link_Get ParametersConfirm`, when there are still MANET PoAs in the message. These remaining PoAs need to be validated, i.e., we have to find out if these MANET PoAs are attached to a MANET that has at least one gateway. Thus, an `AH_POA_DiscoverReq` is sent to the MAGINFO/AUTOCONF module, which will answer with an `AH_POA_Reach` `ableInd` for every gateway found. The MANET Wrapper will then make an *essid* match in order to assure it is the correct network and will also keep track of the best gateway *per* network. Once returned, the best gateway of the network is stored along with the PoA returned by RAL-WLAN and matched by *essid* in the MANET Wrapper. This assures that the association between PoA (which is in fact the MANET neighbour node) and MANET gateway is stored for later reference. Finally, the matching PoA identifiers in the `Link_GetParametersConfirm` are replaced by the gateway identifier.

Case (4) is only slightly different from (2). The first difference resides in temporarily storing the PoA requested for scan in the `Link_GetParametersRequest` and replace the specific PoA scan for a general MANET scan, because it makes no sense to search for a PoA (in this case a gateway) that may be at a multi-hop distance and consequently will never be detected by a simple WLAN (single-hop) scan. After receiving the `Link_GetParametersConfirm`, the message is cleaned of any WLAN PoAs and the validation scan is performed. Afterwards the remaining PoAs are matched with the temporarily stored PoA and, if they match, the message is forwarded to the VIP; otherwise, the message is cleaned of all PoAs and is as if no PoAs were found and then forwarded.

Switching depends on the other two previous procedures. It can only be performed after the previous procedures have been run at least once. This procedure is triggered by the reception of a `Link_SwitchRequest` message. On one hand, if the switch requested is of WLAN technology, then both messages (`Link_SwitchRequest` and `Link_Switch Confirm`) are forwarded unchanged.

On the other hand, if the `Link_SwitchRequest` is issued to a MANET PoA, then the PoA is temporarily stored and the identifier switched by the one associated to it, i.e., the gateway address returned by the upper layer will be swapped by the neighbour PoA, because the RAL-WLAN will not really associate with the gateway since it may not be in single-hop range, but will rather associate with the neighbour which was proven to be somehow associated with the requested gateway. Upon receiving a `Link_SwitchConfirm`, the PoA will firstly be matched in order to check if this is the tracked PoA and, if true, then the neighbour PoA identifier will again be swapped with the gateway identifier and the message forwarded to the VIP module.

Interface with RPSC

There are four main primitives, all started by the Manet Wrapper module, and described earlier, in the implementation of RPSC (see section 4.3.2).

Interface with VIP and RAL_WLAN

These two interfaces use 802.21 messaging and consequently have a large number of primitives. Therefore, they will not be addressed here, except a small reference to the most crutial messages for the module work flow, as follows:

- *Flowing Downward* (MIHF → VIP → **MW** → RAL-WLAN)
  - `Link_GetParametersRequest`

    This message is the request for available PoAs from the designated interface and technology.
  - `Link_SwitchRequest`

    This message is the request for the designated interface to switch from the current PoA and technology to the requested PoA and technology.
- *Flowing Upward* (MIHF ← VIP ← **MW** ← RAL-WLAN)
  - `Link_AdvertisementIndication`

    This message is the advertisement of the interface to MIHF informing it exactly what the interface has and supports (MAC address, technology supported, a vailable set of messages and commands, etc...)
  - `Link_GetParametersConfirm`

    This message is the reply to `Link_GetParametersRequest` and contains the information about available PoAs (their MAC addresses, channels, SSID's and technologies).
  - `Link_SwitchConfirm`

    This message is the reply to `Link_SwitchRequest` and contains usually the result of the switch (unsuccessfull, successfull or rejected), and information to where did it switch to (PoA MAC address, channels, SSID's and technology).

Interface with MAGINFO/AUTOCONF

There are four main primitives, the first three are sent by the Manet Wrapper module, while the last one is sent by MAGINFO/AUTOCONF.

- `Connect(interface_name)`

  This primitive establishes a connection between the modules and also informs MAGINFO/AUTOCONF about the name of the ad-hoc scanning interface.

- `AH_POA_DiscoverReq(interface_ID)`

  This primitive is the request for any available ad-hoc PoAs that can be reached by scanning the interface with `interface_ID`.

- `Close()`

  This primitive closes the connection established by `Connect`.

- `AH_POA_ReachableInd(GW_IPv6_addr, hop_count, Vlink_state)`

  This primitive is the response to `AH_POA_DiscoverReq` and returns information about the reachable PoAs (one message is sent by each PoA found).

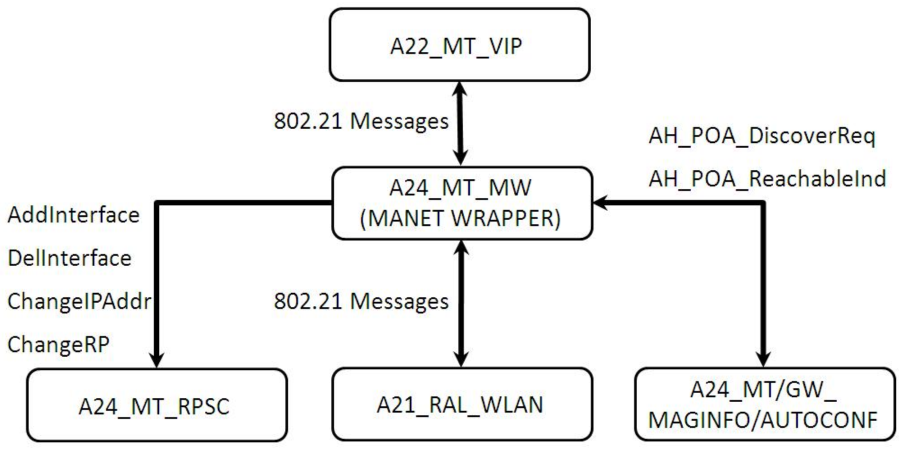

Figure 4.13: A24_MT_MW module interface.

4.4 Conclusions

Even though no results were obtained regarding the performance of the testbed, some conclusions can be drawn about the work produced in this part of the thesis.

The testbed is ready to perform bootstrap and handover tests, being therefore ready to be measured in terms of its performance. Unfortunately, these measures were not taken due to lack of time to perform them, and produce significant results.

Nonetheless, the main objective of this part of the thesis, was achieved and the implemented software is working properly. The testbed to test and measure mobility performance of MANETs, was properly set-up and integration of this software with other partners software was very successfull. Integration tests with other components not only helped to perfect the expected behaviour from the modules implemented during this thesis but also to better understand the working flow of the complete Daidalos mobility framework.

Although the testbed is set-up and working there are still some limitations. Some project design issues resulted in the simultaneous coexistence of MANET and WLAN technologies becoming an impossibility. This means that a mobile node can only support the announcement of either MANET or WLAN technology, rather than both. Consequently, this issue brings up a drastic cut on scenario possibilities, a great conditioning on what type of technology to support, and even graver is the reduced network possibilities for mobility due to absence of one of the technologies.

Also the concept of privacy cannot currently be applied in the MANET environment. Basically privacy is enforced by Virtual Identities (VIDs) which are supported by virtual interfaces (implemented by the Virtual Interface Proxy - VIP). However, currently used MANET routing instances are not able to run in virtual interfaces. Even if modifications ensued to provide support for the use of multiple routing instances, thus multiple VIDs it would still be required a mechanism that could properly sort the packets by the correct virtual interfaces and correspondent routing instance, under penalty of the wrong routing instance process packets which are not destined to it.

The possibility of multihoming and multiple gateways is considered, but is not yet supported, due to flaws presented at both the global mobility domain (GMD) level and local mobility domain (LMD) level. One of these flaws is the fact that the global mobility protocol MIPv6 does not support the registration of multiple Care of Addresses to the same node. Another flaw is that the LMD is not multihome-aware, thus cannot properly forward the traffic.

Chapter 5

Conclusion and Future Work

5.1 Conclusions

5.1.1 Network-on-Wheels

VANETs are a type of network that have undergone a number of tecnhological advances in recent times. Some traditional research methods applied in VANETs have some drawbacks. Field testing for instance is very expensive, while simulation may not provide accurate results, and emulation is too time consuming.

In this thesis a different approach is adopted combining the use of emulation and simulation. The result is an implemented hybrid framework.

Especially developed for VANETs, this testbed consists in the integration of two tools: a simulator/emulator named NCTUns and a PBR routing application called NoWd. The integration of both tools is done in such a way that it allows the simulator/emulator to provide the simulation of the wireless environment as well as the simulation of lower protocol layers in several emulated nodes, and also the emulation of the network protocol stack. This leaves the PBR routing application in charge of any upper layer thus controlling the network routing and application data flow.

This implementation was adapted in order to meet the testbed requirements and still have a good level of integration between the tools. Routing and application related communication supported by the NoWd program is seamless to the NCTUns thus apermitting the execution of multiple real programs to run in simulated nodes using a real protocol stack.

In addition an emulation scenario regarding communication with NoWd prototypes has been implemented. The implementation involved some adaptations on the prototype in order to succesfully allow communication between the prototype and the simulation machine. Not all functions in the PBR routing protocol are supported due to a flaw in the NCTUns application. Testing and measurement of the testbed performance was done to analyse the testbed scalability in terms of node number, these tests consisted in taking system resource measures in order to demonstrate the testbed influence in resource comsumption by variating the number of nodes in the simulated network.

Data collected from these tests pointed out a few important characteristics of the testbed's performance. For instance, the testbed does not require much memory, but it does require a heavy processing power. Regarding execution time the result analysis shows a normal behaviour for networks up to about 70 nodes. However for bigger networks time constraints become difficult to respect due to processing power limitations. This results in a great increase of the execution time both in simulation and emulation for a network size beyond that number of nodes.

In Conclusion a fully working implementation of a hybrid testbed has been seen in this thesis. New, more accurate and less costfull research for VANETs can now be executed, paving the way to new test-proved developments, improvements and enhancements in VANETs.

5.1.2 Daidalos II

Mobility in Mobile Ad-Hoc Networks is an implicit concept and an advantage but also a great challenge. The peculiar aspects that make this type of networks differ from others, are both their strong points and weaknesses.

The high level of node movement can degrade communications, but can also provide connectivity with otherwise unreachable nodes and networks. The freedom of movement and ability to easily connect to any available node is a point in favor. However, the connected network might be isolated and not have access to any other networks.

To cope with mobility caused issues in MANETs The presented framework of Daidalos II aims to develop a solution that integrates several access technologies, including MANET. Integration with other technologies will not only help to cope with MANET mobility issues but will also enable ubiquitous access by providing seamless handovers between technologies.

In this thesis, profit is taken from the Daidalos II mobility architecture and its features in order to set up a testbed to test and measure the performance of MANET nodes regarding mobility events. Such events comprise for instance mobility bootstraps and handovers.

To achieve the described goal, some software components needed to be implemented to complete the MANET related mobility architecture. These required MANET components were then developed, and tested on the testbed. Tests proven to be successfull and the integration of the MANET components with the rest of the mobility framework was fully achieved.

Therefore a testbed for test and measurement of MANETs integrating an architectural mobility solution was produced. However, time constraints did not allow to produce relevant results concerning performance measurements. Regardless, the testbed is ready to perform measurable mobility related procedures such as bootstraps and handovers.

There are also some limitations regarding the testbed, namely some architecture design issues that prevent the use of MANET and WLAN technologies simultaneously, thus limiting the number of possible scenarios to be produced in the testbed.

Also privacy in the MANET environment cannot currently be applied. Privacy which is enforced by Virtual Identities (VIDs), supported by virtual interfaces (implemented by the Virtual Interface Proxy - VIP), isnot available due to restrictions of the used MANET routing protocol.

The possibility of multihoming and use of multiple gateways is considered, but its not yet implemented due to simple flaws in the local and global mobility protocols. These flaws include: MIPv6 inhability to register multiple care of addresses associated to the same node; the need of the local mobility protocol to be multihome-aware.

5.2 Future Work

5.2.1 Network-on-Wheels

In addition to the work in this thesis there is still room for improvement.

The first and foremost aspect to point out is the Emulation Mode broadcast critical flaw that denies a real node to receive broadcast packets from the simulated network. This is impeditive to the establishing of full and correct PBR protocol communication with real hosts. Solving this problem opens a whole new section of Emulation Mode tests to the testbed.

Another important issue to be improved is the manner in which the position is transmitted to the node's NoWd applications. Calculating a new position format and also opening and closing of a communication channel are procedures executed every time that a packet is sent by the node thus the method is resource consuming and not eficient affecting the overall performance of the testbed. This arises greater issues if the network load is high. A more efficient and less resource consuming solution could be devised.

The realness of node movement, i.e., the node's ability of producing more realistic behaviour can be enhanced by the simple introduction of real traffic and movement patterns in order to produce the node's path. Resulting in a more realistic dynamic topology and consequently more truthfull outcomes.

An aspect that could also be explored in order to produce better results is the use of an advanced MAC module to improve the network routing and the wireless environment accuracy.

Other more obvious enhancements of the testbed include the fine tuning of the testbed performance by means of extensive software testing and the addition of future developments on both the NoWd application and the NCTUns simulator/emulator.

5.2.2 Daidalos II

On this part of the thesis a good amount of work can still be done in order to improve the overall framework.

Starting by the implementation of a `ConnectionLost` primitive which is in the project specification. This primitive is to be implemented in both `A24_MT_OLSR` and `A24_MT_RPSC`. Its purpose is to inform `A24_MT_MW` of the loss of connectivity with a PoA.

This would mean that the `OLSRDeamon` would also have to be changed in order to support the notification of other modules.

As for the module `A24_MT_RPSC` there are another couple of improvements that can be done: the first one is the extension to dynamically allow the use of several routing protocols; and the second one is to have a cleaner way to implement the `ChangeIPAddr` primitive, since it is assumed that all routing protocols can cope with dynamic addressing and also multiple addresses in an interface.

In the `A24_MT_MW` module some modifications can be done to improve the overall performance of the module. The communication between `A24_MT_MW` and `MAGINFO/AUTOCONF` can be refined and made more efficient. Also the wrapper is only ready to receive messages from the MIHF (via VIP) and from the RAL-WLAN. This means that also needs changes applied to support the reception of the `ConnectionLost` primitive. The mechanism that manages the knowledge of PoAs can be refined to provide a cleaner and easier way of operating with PoA data.

In a more general view over the framework, there is an addition of great importance that can be inserted in the testbed. The solving of the design issues that prevent the coexistence of WLAN and MANET technologies, will remove some of the major limitations of this testbed. However this can only be achieved by envolving other concerning partners in the discussion.

The support of multiple Virtual Identities (VIDs) is also a point for improvement. The design and implementation of a solution to enable the use of multiple VIDs in a MANET environment is paramount in order to fullfill one of the main goals of the Daidalos II project, which is privacy.

Another matter that can be addressed here is Multihoming and the use of multiple gateways. There is already some work done on this subject, presented in [27], and that can be fitted in the current testbed.

Bibliography

[1] Daidalos Project. http://www.ist-daidalos.org.

[2] Linux Programmer's Manual Pages.

[3] NoW - Network on Wheels. http://www.network-on-wheels.de.

[4] NS-2 Homepage. http://www.isi.edu/nsnam/ns/.

[5] OLSR Deamon Homepage. http://www.olsr.org.

[6] Wikipedia, the on-line encyclopedia. http://www.wikipedia.org.

[7] R. Baldessari, A. Festag, R. Schmitz, H. Füßler, S. Schnaufer, and M. Torrent-Moreno. NoW - Network on Wheels: Description of the Communication System Demonstrator. Technical Report NLE-CR-2006-59, Network Laboratories, NEC Deutschland GmbH, June 2006. 84 pages.

[8] E. Conchon, J. Garcia, T. Perennou, and M. Diaz. Improved IP-Level Emulation for Mobile and Wireless Systems. *IEEE WCNC*, 2007.

[9] Car2Car Communication Consortium. Car2Car Communication Consortium Manifesto - Overview of the C2C-CC System, May 2007.

[10] A. Festag, R. Baldessari, and H. Wang. On Power-Aware Greedy Forwarding in Highway Scenarios. WIT, 2007.

[11] W. Franz, H. Hartenstein, and M. Mauve, editors. *Inter-Vehicle-Communications Based on Ad Hoc Networking Principles – The Fleetnet Project.* Universitätsverlag Karlsruhe, http://www.uvka.de/univerlag/volltexte/2005/89/, November 2005. ISBN 3-937300-88-0.

[12] OPNET Technologies Inc. http://www.opnet.com/.

[13] P. Jacquet and T. Clausen. Optimized link state routing protocol. *RFC 3626*, 2004.

[14] U. Jonsson, F. Alriksson, T. Lasson, P. Johansson, and G.Q. Maguire Jr. MIP-MANET - Mobile IP for Mobile Ad Hoc Networks. *Workshop on Mobile Ad Hoc Network and Computing (MobiHOC'00)*, August 2000. Boston.

[15] A.M. Law. *Simulation Modeling & Analysis*. McGraw-Hill, 4th edition, 2007. ISBN 0-07-298843-6.

[16] C. Maihofer and R. Eberhardt. Geocast in Vehicular Environments: Caching and Transmition Range Control for Improved Efficiency. *IEEE Intelligent Vehicle Symposium*, pages 951–956, June 14-17 2004.

[17] M. Mauve, J. Widmer, and H. Hartenstein. A Survey on Position-Based Routing in Mobile Ad-Hoc Networks. *IEEE Network*, 15(6):30–39, November/December 2001.

[18] C.Siva Ram Murthy and B.S. Manoj. *Ad Hoc Wireless Networks: Architectures and Protocols*. Prentice-Hall PTR, 2nd edition, 2004. ISBN 0-13-147023-X.

[19] M.C. Necker, C.M. Gauger, S. Kiesel, and U. Reiser. IKREmuLib: A Library for Seamless Integration of Simulation and Emulation.

[20] A. Nilsson, A. Hamidian, and U. Korner. Micro Mobility and Internet Access Performance for TCP connections in Ad hoc Networks. *Nordic Teletraffic Seminar 17*, 2004. Oslo - Norway.

[21] K. Pawlikowski, H.-D. Joshua Jeong, and J.-S. Ruth Lee. On Credibility of Simulation Studies Of Telecommunication Networks. January 2001.

[22] C.E. Perkins. *Ad Hoc Networking*. Addison Wesley, 2001. ISBN 0-201-30976-9.

[23] C.E. Perkins and E.M. Royer. Ad-hoc on-demand distance vector routing. In *Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications*, pages 90–100, 1999.

[24] E.M. Royer and C.-K. Toh. A Review of Current Routing Protocols for Ad Hoc Mobile Wireless Networks. *IEEE Personal Communications*, pages 46–55, April 1999.

[25] P.M. Ruiz and A.F.G. Skarmeta. Integrated IP Multicast in Mobile Ad-Hoc Networks with Multiple Attachments to Wired IP Networks. *15th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, PIMRC 2004*, (Vol.1):579–583, 2004.

[26] S. Sargento, T. Calçada, J.P. Barraca, S. Crisóstomo, J. Girão, M. Natkaniec, N. Vicari, F.J. Galera, M. Ricardo, and A. Glowacz. Mobile Ad-Hoc Networks Integration in the Daidalos Architecture. In *Proceedings Mobile Summit Communications*, June 2005. Dresden - Germany.

[27] S. Sargento, R. Sarrô, R. Duarte, and P. Stupar. Seamless Mobility Architecture Supporting Ad-hoc Environments.

[28] S.-Y. Wang. The GUI User Manual for the NCTUns 3.0 Network Simulator and Emulator. Distributed as NCTUns 3.0 Software Documentation, March 2006.

[29] S.-Y. Wang and K.-C. Liao. *Computer Networking and Networks, Chapter 7.* Nova Science Publishers, 2006. ISBN 1-59454-830-7.

[30] S.-Y. Wang and Y.-B. Lin. NCTUns network simulation and emulation for wireless resource management. *Wireless Communications and Mobile Computing*, (5):899–916, 2005.

[31] S.Y. Wang, C.L. Chou, C.H. Huang, C.C. Hwang, Z.M. Yang, C.C. Chiou, and C.C. Lin. The Design and Implementation of the NCTUns 1.0 Network Simulator. Distributed as NCTUns 3.0 Software Documentation.

[32] S.Y. Wang, C.L. Chou, Y.S. Tseng, M.S. Hsu, Y.W. Cheng, W.L. Liu, and T.W. Ho. NCTUns 4.0: An Integrated Simulation Platform for Vehicular Traffic, Communication, and Network Researches. WiVeC, 2007.

[33] S.Y. Wang, C.H. Huang, C.C. Lin, C.L. Chou, and K.C. Liao. The Protocol Developer Manual for the NCTUns 3.0 Network Emulator and Simulator. Distributed as NCTUns 3.0 Software Documentation, March 2006.

[34] M. Zec. Implementing a Clonable Network Stack in the FreeBSD Kernel. In *Proceedings of the 2003 USENIX Annual Technical Conference*, 2003.

[35] M. Zec and M. Mikuc. Operating System Support for Integrated Network Emulation in IMUNES.