Development of a Hybrid Simulation and Emulation Testbed For VANETs

R. Costa, S. Sargento, R. Aguiar, Instituto de Telecomunicações, Aveiro (Portugal)
and W.Zhang, NEC Network Laboratories, Heidelberg (Germany)

Abstract – 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 and simulation lacks accuracy in wireless environments.

This article aims to introduce 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.

I. INTRODUCTION

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

Vehicular Ad-hoc Networks (VANETs) as a specific type of Mobile Networks also have inherited many of these challenging aspects. 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 is used for research and development in this diploma thesis.

This article is related to the Network on Wheels (NoW) project [1]. 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 the network and transport protocols, their implementation in an experimental platform and performance evaluation in realistic environments. An application part of this project is the NoW demonstrator which supports the network layer.

This article introduces an integrated testbed for test and measurement in a VANET environment.

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 issue. Furthermore the testbed is tested and measured in order to determine the framework's performance, scalability and limitations.

II. AD-HOC NETWORKS

A. 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 [2]. 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.

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.

B. Vehicular Ad-Hoc Networks

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.

In the beginning, mobile ad-hoc technology was used by vehicles to provide communication between emergency vehicles in emergency scenarios [3]. Altough helpful, this was only one of the many applications of mobile ad-hoc networks. Other vehicular applications focused on avoiding road accidents.

Vehicular Ad-Hoc Networks (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.

B.1 VANET Architecture

An overview of the VANET high-level arquitecture is shown in [Fig 1] [4]. 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).
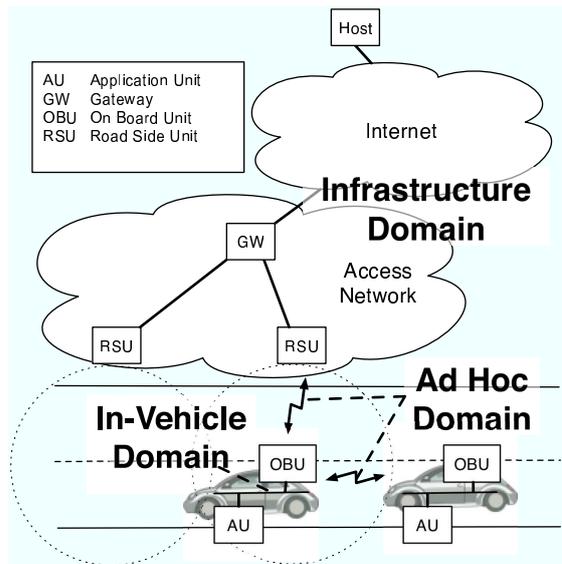

Figure 1 - VANET Architecture

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.

B.2 VANET Characteristics

Beow are described some of the most important characteristics of VANETs [2] [3]:

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.

B.3 VANET Application Scenarios

Safety [5]: Cooperative Collision Forwarding, Pre-Crash Sensing/Warning, Hazardous Location Vehicle-to-Vehicle Notification.

Traffic Efficiency [5]: Enhanced Route Guidance and Navigation, Green Light Optimal speed Advisory, Vehicle-to-Vehicle Merging Assistance.

Infotainment and Others [5]: Internet Access In Vehicle, Point of Interest Notification, Remote Diagnostics

III. 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 [6]: This type of approach produces routes using link information to actively route packets [7]. The protocols of this approach can generally be subdivided in [7]:

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 [7]: Position routing relies on a node's knowledge of it's own p[osition and other nodes's positions.

A. 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, such as the Distance-Aware Greedy Forwarding Algorithm [8] [9].

B. PBR Protocol

As specified by the C2C-CC [5], the PBR Communication System [4] supports several types of applications.
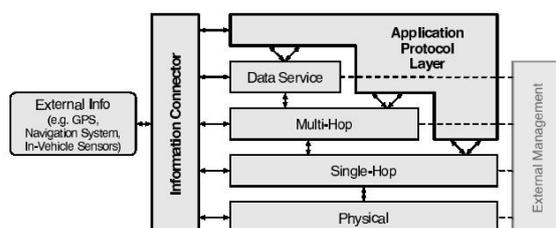

Figure 2 - PBR protocol Architecture

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.

There is a *Beacon Service* which will allow nodes to periodically advertise information such as the node's position and it's identifier. This service makes a node *aware* of it's neighbours.

There is also a *Location Service* that aims to complement the *Beacon Service* to obtain information about nodes who are not directly connected. This service is only triggered upon request.

The *Forwarding Service* 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 [4] are the Unicast, Topologically Scoped Broadcast, Geo-Broadcast and Geo-Anycast.

IV. Simulation and Emulation

A. Motivation

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 [10] [11].

Usually field tests are only done after extensive reasearch 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 [9]. 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.

B. Required Functionalities

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: the 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 and the capability to run multiple real VANET routing implementations on a single machine simultaneously.

V. A HYBRID TESTBED

A. 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.

B. Simulator/Emulator Selection

There are several different types of Simulators and Emulators, each one focusing on a feature or characteristic of the scenario. Some of those types of Simulators and Emulators are: Traffic, Network, Movement, Protocol-Oriented and Application-Oriented.

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 were: W-NINE [10], IMUNES [12], FreeBSD Clonable Network Stack Method [13], IKREmuLib [14], NS-2 [15], OPNET [16] and NCTUns [17] [18] [19] [11].

The National Chiao Tung University network simulator (NCTUns v3.0) 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 prerequisites. 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.

B.1 NCTUns

The initial version of this software the NCTUns [18] [17] [19] [20] [21] [11] was initially designed to reduce or avoid the impact of drawbacks in simulation and it has evolved in many areas since then.

The features presented by NCTUns v3.0 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 (the recently released version v4.0 [11] already supports vehicular networks, but was not available at the beginning of the project)
- 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 [18].
- 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

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.

C. Network on Wheels demonstrator (NoWd)

The *Network on Wheels* Project (NoW) [1] 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) [4]. This demonstrator is an evolution from the *Fleetnet* Project [22] 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.

C.1 Component Architecture

In [Fig. 3] we can see the reference composition of a VANET OBU.

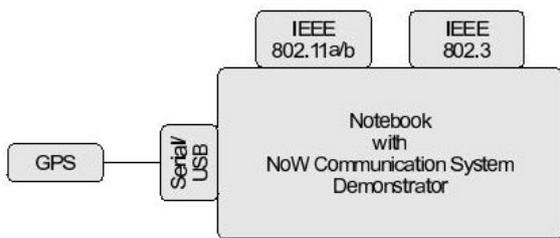There are three distinct interfaces. The wireless interface

Figure 3 - NoWd prototype component architecture

(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.

C.2 Protocol Architecture

The protocol architecture as depicted in [Fig.4] 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.
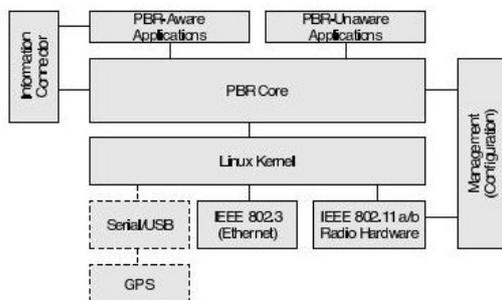

Figure 4 - NoWd prototype protocol architecture

Compared with the PBR protocol architecture [Fig.2], the resemblence is evident. Although a rougher and conceptually simpler version, the NoWd protocol architecture contains the core elements of the PBR protocol architecture.

C.3 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 ad-

dresses 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.

VI. 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 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 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.
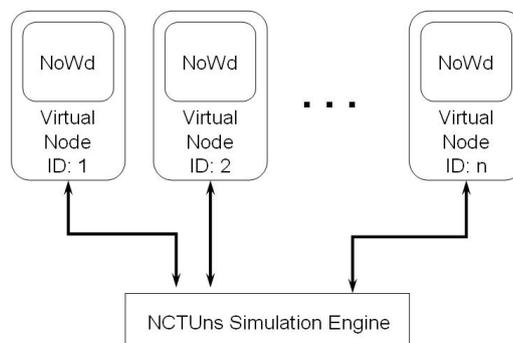

Figure 5 - Testbed architecture

After some investigation a solution was elaborated, the al-

location of a NoWd program running in each virtual node [Fig.5]. 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, proved feasible.

In [Fig.6] 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.
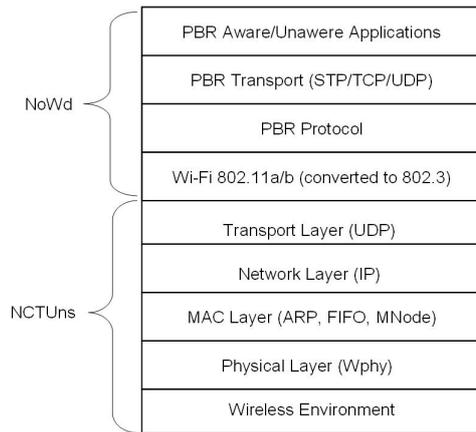

Figure 6 - Testbed Protocol Stack

VII. IMPLEMENTATION

The implementation mainly consisted on two aspects [23]:

- Integration of both applications in order to build the testbed, which included changes on both parts to allow the flow of essential information between them.
- The creation of mechanisms that could simplify the usability of the testbed.

VIII. EMULATION MODE

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. 7].
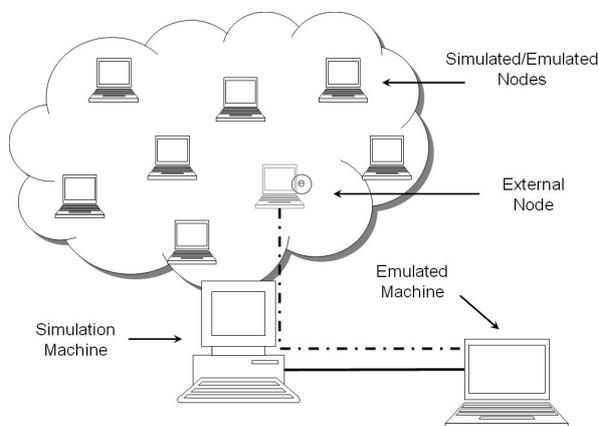

Figure 7 - Description of Emulation Scenario

As it is clear in [Fig. 7] 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 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.

A. 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.

IX. PERFORMANCE EVALUATION

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 & Memory Usage

A. 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.

A.1 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 [24]. 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.

A.2 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.

A.3 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.

B. 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.

For further details ont the system software & hardware settings see [23].

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.
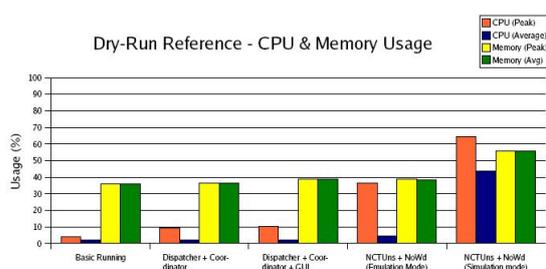

Figure 8 - Dry-Run System Reference

As seen in [Fig. 8] 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.

X. 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.

A. Experiment Details

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.

XI. 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.

A. Comparison of Results

A.1 CPU Usage

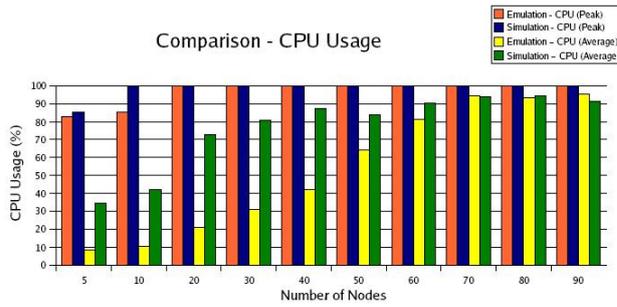As depicted in [Fig 9] there is a clear difference between Emulation and Simulation resource comsumption. For the

Figure 9 - Comparison of CPU Usage

Figure 11 - Comparison of Execution Time

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.
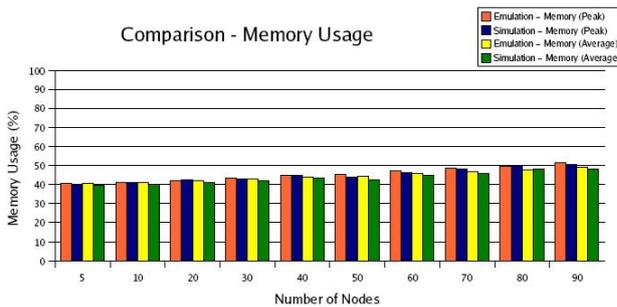
A.2 Memory Usage

Figure 10 - Comparison of 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. 10]. 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 cost-full than simulation and the overall behaviour does not seem to be very heavy for the system.

A.3 Execution Time

The evolution of the execution time is rather special as seen in [Fig. 11]. Both simulation and emulation maintain a regularly expected behaviour, with simulation 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.

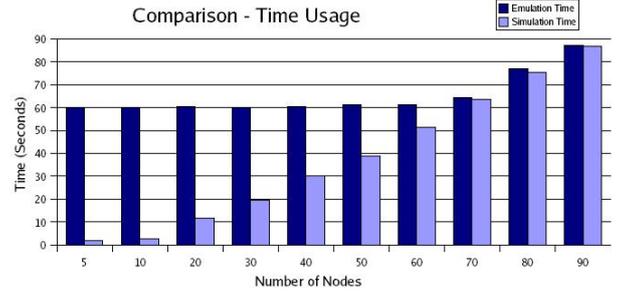In the time point of view we can state that simulation is far better until the referred turning point. From this point onward execution time greatly exceeds the pre-defined simulation time for both simulation and emulation.

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.

XII. CONCLUSIONS

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 article 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.

XIII. Future Work

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.

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.

References

[1] "NoW - Network on Wheels", http://www.network-on-wheels.de.

[2] 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.

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

[4] 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.

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

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

[7] M. Mauve, J. Widmer, and H. Hartenstein, "A Survey on Position-Based Routing in Mobile Ad-Hoc Networks", *IEEE Network*, vol. 15, no. 6, pp. 30–39, November/December 2001.

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

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

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

[11] 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.

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

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

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

[15] "NS-2 Homepage", http://www.isi.edu/nsnam/ns/.

[16] OPNET Technologies Inc, ", http://www.opnet.com/.

[17] 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.

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

[19] S.-Y. Wang and Y.-B. Lin, "NCTUns network simulation and emulation for wireless resource management", *Wireless Communications and Mobile Computing*, , no. 5, pp. 899–916, 2005.

[20] 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.

[21] 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.

[22] W. Franz, H. Hartenstein, and M. Mauve, Eds., *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.

[23] R. Costa, "Development of a Hybrid Simulation and Emulation Testbed For VANETs", Master's thesis, Universidade de Aveiro, 2007.

[24] "Linux Programmer's Manual Pages".