Detail Comparison of Network Simulators

Mohammed Humayun Kabir, Syful Islam, Md. Javed Hossain, Sazzad Hossain

Abstract: Simulation software is an important platform of finding results to be expected from a practical hardware setup which is costly and tedious to modify frequently. There are a lot of Network Simulators in the communication world. Some of them are dedicated to wireless network, some of them are dedicated to wired network or both type of networks. Because of wide variations in operating systems, hardware requirements, programming software requirements, output features and scalability, it is very difficult to choose a suitable simulator for a specific job. Our aim is to subjugate the barrier. We have studied and compared the detail features of the Network Simulators for the sake of choosing the absolutely necessary one from the pool of simulators.

Index Terms—License Type, Simulation Event Type, Available Module, Scalability, Parallelism, GUI Support, Ease of Use.

1. INTRODUCTION

THE simulators are crying need of researchers in all over the world. Finding a suitable simulator from the pool of them is not so easy. From the most important necessary point of view, we have studied about computer network communication simulators on the basis of availability, easiness, scalability and other important properties (user interface, data manipulation, graphical display etc.) of them. We have studied so far the discrete event simulators (the system behavior can be simulated by modeling the events in a system as per order of the scenarios the user has setup), for example: NS2, NS3, QualNet, GloMoSim, NetSim, OMNeT++, OPNET, TOSSIM, J-SIM, NCTUns, DRMSim, SSFNet, GrooveNet, TraNS etc. After detail discussion of each simulator, we have summarized the properties in Table 1 and Table 2. As a result, any researcher will be able to choose the necessary simulator quickly.

2. SIMULATORS& THEIR PROPERTIES

The simulators studied so far have been described in detail as follows.

2.1 NS2

NS2 (Network Simulator Version-2) is an open source discrete event simulator designed especially for network research[1]. In 1996-'97, NS2 was initiated and licensed for use under General Public License (GNU). It provides support for both wired and wireless simulation of functions and protocols such as TCP, UDP etc. NS2 was developed from the ideology of REAL simulator[19]. NS2 is one of the popular simulators due to its flexibility and modular behavior. It is written in two key languages: C++ and Object-Oriented Tool Command Language (OTcl)[15]. C++ defines the internal mechanism of simulation objects. OTcl is used for users to control the simulation scenario and schedule the events. The C++ and the OTcl are linked together using TclCL. NS2 provides a large number of built-in C++ objects. It is advisable to use these C++ objects to setup a simulation using a Tcl simulation script. However, advance users may find these objects insufficient. They need to develop their own C++ objects, and to use a OTcl configuration interface to put together these objects.NS2 performs simulation to explore different issues like protocol interaction, congestion control, effect of network dynamics, scalability etc. It runs on the right kind of scenario which includes but is not limited to the topology size, density distribution, traffic generation, membership distribution, real-time variance of membership, network dynamics etc. NS2 outputs either text-based or animation-based simulation results [32]. To interpret these results graphically and interactively, tools such as NAM (Network AniMator) [16] XGraph [17] and Gnuplot [18] are used. To analyze a particular behavior of the network, users can extract a relevant subset of text-based data and transform it to a more flexible presentation.

Type: Open source.

Languages: C++ and Object-Oriented Tool Command Language (OTcl).

Supported Operating System: GNU/Linux,FreeBSD, Mac OS X, Windows XP, WindowsVista and Windows 7.

Hardware Requirement: Free disk space of 5GB required, minimum 256MB RAM suggested

Features:

- *Dr. Mohammed Humayun Kabir has been serving the department of Computer Science and Telecommunication Engineering as an Associate Professor at Noakhali Science and Technology University, Sonapur, Noakhali-3814, Bangladesh, E-mail: hkabir269@gmail.com*
- *Syful Islam is an undergraduate student of the department of Computer Science and Telecommunication Engineering at Noakhali Science and Technology University, Sonapur, Noakhali-3814, Bangladesh. E-mail: syfulcste@gmail.com*
- *Md. Javed Hossain has been serving the department of Computer Science and Telecommunication Engineering as an Associate Professor at Noakhali Science and Technology University, Sonapur, Noakhali-3814, Bangladesh, E-mail: javed.nstu@gmail.com*
- *Sazzad Hossain. is an undergraduate student of the department of Computer Science and Telecommunication Engineering at Noakhali Science and Technology University, Sonapur, Noakhali-3814, Bangladesh. E-mail: sazzadhossain@live.com*

- NS2 supports modeling of traffic distribution.
- It provides simulation of a variety of protocols HTTP, TCP, UDP, SRM, RTP and some routing algorithms.
- It covers almost all variants of TCP, several forms of multicast, wired networking, several ad-hoc routing protocols and propagation models (but not cellular phones), data diffusion, satellite and other stuff.
- NS2 uses different **scenario generators** that consists of a **topology generator,** an **agent generator** and a **routing generator[20]**.
- The **topology generator** can generate topology using any standard graph generator (GT-ITM, Tiers etc). Currently it supports GT-ITM generator only and converts the topology graph into NS format.
- The **agent generator** can be used to define transport protocol agents like TCP or SRM.
- The **routing generator** defines the routing protocols to be used in the simulation. The options include unicast routing, multicast routing and an option to turn on/off the expand address.
- In NS2, the event scheduler keeps track of simulation time and release all the events in the event queue by invoking appropriate network components.
- It provides NAM and X-Graph to animate and plot the simulation result respectively.

Advantages:

1. NS2 has large number of available models, realistic mobility models, powerful and flexible scripting and simulation setup, large user community and ongoing development.
2. NS2 provides an easy traffic and movement pattern by including an efficient energy model.
3. It provides a set of randomized mobility model[29] and there are several projects to bring advanced mobility models to the simulators.
4. Complex scenarios can be easily tested.
5. Popular for its modularity.

Limitations:

1. NS2 needs to be recompilation every time if there is a change in the user code.
2. Real system is too complex to model i.e. complicated infrastructure.
3. The mixture of compilation and interpretation made it difficult to analyze and to understand the code.
4. Including a lot of nodes may slow down the simulation process.

2.2 NS3

The *NS3[2]* simulator is a discrete-event network simulator targeted primarily for research and educational use. It is licensed under the GNU GPLv2 license, and is available for research and development-3. It defines a model of working procedure of packet data networks, and provides an engine for simulation. Several users are using*NS3* to model non-Internet-based systems.NS3 uses two key languages like its predecessor NS2. The simulator is entirely written in C++ with optional python bindings. Simulation scripts can therefore be written in C++ or in Python. Animators are used to visually display the results. Both languages work nicely on it.NS3 provides a strong library which is useful for the user to do their work by editing NS-3 itself. For wired technology, the*NS-3* provides device model of a simple network of Ethernet which uses CSMA/CD as its protocol scheme with exponentially increasing back-off to contend for the shared transmission medium. It also provides a set of 802.11 models that attempt to provide an accurate MAC-level implementation of the 802.11 specification and a PHY-level model of the 802.11a specification.

Type: Open source.

Languages: C++ and Optional Python Bindings.

Supported Operating System: GNU/Linux, FreeBSD, Mac OS X, Windows XP, Win-Vista and Windows 7.

Hardware Requirement: Free disk space of 5GB required, minimum 256MB RAM suggested

Features:

- It reduces the memory footprint of the simulation and allocate no memory for virtual zero bytes. Mobility model is not needed, as the wired devices do not need to know its node position.
- The core of NS3 is written in C++ and with Python scripting interface (compared with OTcl in NS2). Several advanced C++ design patterns are also used[2].
- It generates PCAP packet trace files, other utilities can be used to analyze traces as well. Users can write program for designing a simulation either in C++ or in Python Programming Language.
- The protocol entities are designed to be closer to real computers.

- It supports the incorporation of more open-source networking software and reduce the need to rewrite models for simulation
- Alight weight virtual machines are used.

Advantages:

1. High modularity than its ancestor NS2.
2. Support simulation for TCP, UDP, ICMP, IPv4, multicast routing, P2P and CSMA protocols.
3. Support for ported code should make model validation easier and more credible.
4. Much more flexible than any other simulators.
5. Wide range of use in both optimization and expansion of the existing networks.

Limitations:

1. NS3 still suffers from lack of credibility.
2. NS3 is intended to replicate the successful mode of NS2 in which a lot of different organizations contributed to the models and components based on the framework of NS2.
3. NS3 needs a lot of specialized maintainers in order to avail the merits of NS3 as the commercial OPNET network simulators.
4. Active maintainers are required to respond to the user questions and bug reports, and to help test and validate the system.

2.3 QualNet

QualNet(Quality Networking)is a commercial version of GloMoSim used by Scalable Network Technologies (SNT), written purely in C++. It is a network evaluation software and is entirely modeled as a finite state machine[23]. QualNet is engineered on a layered architecture comprising of Application, Transport, MAC and Physical layers. It can simulate a mixture of both wired and wireless networks. Unlike the NS simulator, QualNet can simulate multiple wireless technologies including 802.11s draft. It supports plenty of mobility models including the Group mobility model, Random waypoint model and Trace-based models. QualNet is equipped with an extensive range of libraries for simulating a variety of networks like WiFi, Sensor networks, MANET and WiMAX[23]. Finally, QualNet is unique for providing a comprehensive environment for designing protocols, creating and animating network scenarios, and analyzing their performance.

Type: Commercial. But they have separate licenses for Academicians and others.

Language: Object Oriented Programming Language (C++)

Supported Operating Systems[3]: UNIX, Windows (Windows 7 Home Premium and Professional 32-bit and 64-bit editions, Windows 8 and Windows 8 Pro 32-bit editions), MAC and Linux (CentOS -5.9, Red Hat Enterprise Linux 5.9, Ubuntu-12.04 LTS).

Hardware requirements: Processor[3]: 32-bit (x86 compatible) or 64-bit (x86-64 compatible),**Memory:** 4 GB free, **Disk:** 5 GB disk space, **Video:** 1680 x 1050 or better screen resolution Discrete graphics card with at least 2 GB memory supporting hardware 3D acceleration

Features:

- QualNet can support real-time speed to enable developers and network designers to run multiple analyses by varying model, network, and traffic parameters in a short time.
- Itcan model thousands of nodes by taking advantage of the latest hardware and parallel computing techniques. It is a very powerful simulation tool that can support simulation of 500 to 20,000 nodes.
- QualNet can run on cluster, multi-core, and multi-processor systems to model large networks with high fidelity.
- It provides high fidelity commercial protocol and device models to enable more accurate modeling of real-world networks.
- It enables the designer to design large wired and wireless networks using pre-configured or user-designed models.
- It also facilitates to design new protocol models and to optimize new and existing models.
- QualNet can connect to other hardware and software applications, such as OTB, real networks, and third party visualization software, to greatly enhance the value of the network model.

Advantages:

1. QualNet supports multiprocessor systems and distributed computing.
2. It can simulate a mixture of both wired and wireless networks.
3. It provides GUI that is convenient and easy-to-use.
4. It facilitates sophisticated animation capabilities.
5. It can run on cluster, multi-core, and multi-processor systems.

Limitations:

1. The simulation tool QualNet is an extension of GloMoSim which is being commercialized.

2. Installation of QualNet on Linux is difficult.
3. The java based user interface provided by this simulation software is slow.

2.4 GloMoSim

Global Mobile Information System Simulator (GloMoSim) is a parallel discrete event simulation software[4] that simulates wireless and wired network systems. It is designed as a set of library modules, each of which simulates a specific wireless communication protocol in the protocol stack. It assumes that the network is decomposed into a number of partitions and a single entity is defined to simulate a single layer of the complete protocol stack for all the network nodes that belong to the partition. The parallel implementation of GloMoSim can be executed using variety of conservative synchronization protocols, which include the null message and conditional event algorithm. The library has been developed using PARSEC[4], a C based parallel simulation language. It uses the Parsec compiler to compile the simulation protocols. It has been designed to be extensible and comprehensible. GloMoSim aims to develop a modular simulation environment for protocol stack that is capable of scaling up networks with thousands of heterogeneous nodes. GloMoSim currently supports protocols for a purely wireless network[4].

Type: Open Source

Language: Basic knowledge on PARSEC, C programming.

Supported Operating Systems: Windows(Windows NT or 2000 with Visual C++ ver. 6.0) and Linux(Red Hat 6.0 or higher), Sun SPARC Solaris 2.5.1 or higher with gcc/g++, SGI IRIX 6.4 or higher with gcc/g++.

Hardware requirements: Processor: 32-bit (x86 compatible) or 64-bit (x86-64 compatible),**Memory:** 4 GB free, **Disk:** 5 GB disk space.

Features:

- GloMoSim is a library-based sequential and parallel simulator for wireless networks.
- GloMoSim facilitates the ability to use in a parallel environment which distinguishes it from most other wireless network simulators.
- It allows the simulation Scalability to simulate networks with a hundred and thousand of nodes[20].
- It supports various layers like Mobility, Radio Propagation, Radio Model, Packet reception models, Data Link, Network (Routing), Transport and Application. i.e. (It supports almost all the OSI layers with limited benefits).
- GloMoSim supports direct satellite communication, multi-hop wireless communication and most of the traditional internet protocols.
- It facilitates to build a library of parallelized models[26] that can be used for the evaluation of a variety of wireless network protocols.

Advantages:

1. GloMoSim Provides modular simulation for protocol stack.
2. It is capable of scaling up to networks with thousands of heterogeneous nodes.
3. Parallel model execution is provided to users in transparent manner[21].
4. It is free for education and research.

Limitations:

1. The documentation of GloMoSimis quite poor. No specific routing protocols for sensor network, no energy consumption models for transport layer and IP address support.
2. It provides the Random Waypoint mobility model, which may not be suitable for all types of simulations.
3. Update of this simulator is not regular since the developer team is targeting mainly on QualNet which is the commercial VersionofGloMoSim
4. PARSEC support for Redhat 7.2 which is outdated[19].

2.5 NetSim

NetSim is a stochastic discrete event network simulationtool[5] used for network lab experimentation and research. It's a leading network simulation software for protocol modeling and simulation, allowing us to analyze computer networks with unmatched depth, power and flexibility. NetSim comes with an in-built development environment, which serves as the interface between User's code and NetSim's protocol libraries and simulation kernel[22]. It provides network performance metrics at various abstraction level such as Network, sub-network, Node and a detailed packet trace. It has unique features and functionality.NetSimis available as Standard or Academic versions and is built on a common design framework of high level architecture and code. In a word, NetSim is truly a fantastic product that is not only versatile, but also robust and provides those features that are hard to come with any simulators.

Type: Proprietary [5]

Language: c and java.

Supported Operating Systems: Windows (Windows 7, Windows Vista, Windows XP). It also requires .NET Framework: Microsoft .NET Framework Version 3.5

Hardware Requirements: Processor: 1-GHz Pentium processor or equivalent (Minimum); 3-GHz Pentium processor or equivalent (Recommended),**RAM:** 512 MB (Minimum); 2 GB (Recommended),**Hard Disk:** Up to 100 MB of available space, **Display:** 1024 x 768, 256 colors (Minimum); 1024 x 768 high color, 32-bit (Recommended), and Active Internet connection.

Features:

- NetSim modeling and simulation are supported for Aloha, Slotted Aloha, Token Ring/Bus, EthernetCSMA/CD, Fast and Gigabit Ethernet, WLAN - IEEE 802.11 a/b/g/n and e, X.25, Frame Relay, TCP, UDP, IPv4 and IPv6, Routing - RIP, OSPF, BGP,MPLS, Wi-Max, MANET, GSM, CDMA, Wireless Sensor Network, Zigbee, Cognitive radio)[5].
- It simulates a wide variety of Cisco routers, including 2500 series, 2600 series, 2800 series, and 3600 series routers, as well as the Cisco Catalyst 1900 series, 2900 series, and 3500 series switches.
- Protocol libraries are available as open C code for user modification. This can help avoid the time consuming process of programming, customization and configuration commercial simulators to meet customer specific needs.
- Along with the Boson Virtual Packet Technology engine NetSim utilizes Boson's proprietary Network Simulator, Router Simulator, and EROUTER software technologies, to create individual packets. These packets are routed and switched through the simulated network, allowing NetSim to build an appropriate virtual routing table and simulate true networking. Other simulation products on the market do not support this level of functionality.
- It can be used to create a simulation of the topology of corporate network and help practice troubleshooting without using devices on the production network.

Advantages:

1. NetSim has a GUI which features drag and drop functionality for devices, links etc. i.e. Modeling in NetSim is simple and user friendly.
2. It has a built in analysis framework that provides intra and inter-protocol[19] performance comparison with graphical options.
3. Data packet and control packet flow can be visualized through NetSim'sbuilt-in packet animator.
4. It is easy to learn all about NetSim.

Limitations:

1. NetSim is a single process discrete event simulator.A single event queue is used for the simulation which at any given time contains one entry for each station on the network.
2. Free version of NetSim is not available.

2.6 OMNeT++

OMNeT++ (Objective Modular Network Tested in C++)[6] is an open source, extensible, modular, component-based discrete event simulator tool like NS-2 and NS-3 to simulate networks both wired and wireless. It is completely written in C++. It is mostly used in research and educational purposes and in the global scientific community.It offers an Eclipse-based IDE, a graphical runtime environment and a host of other tools. It is a general-purpose simulator that capable of simulating any system composed of devices interacting with each other. The support for wire and wireless simulation is fairly incomplete. OMNeT++ provides a component-based, hierarchical, modular and extensible architecture. Componentsand modules are programmed in C++, and then assembled into larger components and models using a high-level language (NED*)*. It provides reusability of model for free. OMNeT++ has vast GUI support and due to its modular architecture, the simulation kernel (and models) can be embedded easily into applications. Beside the simulation kernel library, the simulation environment contains a Graphical Network Editor (GNED), a NED compiler, graphical (Tkenv) and command line (Cmdenv) interfaces for simulation.

Type: Free for academic and non-profit use. Commercial user must obtain its license.

Language: Full system is developed with C++. But it also supports C# and JAVA in its IDE environment.

Supported Operating System: Windows, Linux, Mac OS X, and other Unix-like system.

Hardware Requirement: 512 MB RAM and 400 MB space in hard disk.

Features:

- It is a rich-featured and powerful simulation tool. Itprovides infrastructure and tools forwritingsimulations. One of the fundamental ingredients of this infrastructure is component architecture for simulation models.
- Modules can be connected with each other via gates and combined to form compound modules. The depth of module nesting is not limited. Modules communicate through message passing, where messages may carry arbitrary data structures.

- OMNeT++ simulations can be run under various user interfaces. Graphical, animating user interfaces are highly useful for demonstration and debugging purposes.Command-line user interfaces are the best for batch execution.
- Released with full source code.It is free to use.
- Domain-specific functionality such as support for sensor networks, wireless ad-hoc networks, Internet protocols, performance modeling, photonic networks, etc. are available in it.
- OMNeT++ also supports parallel distributed simulation[19]. The parallel simulation algorithm can easily be extended and/or new ones can be plugged in. Models do not need any special instrumentation to run in parallel - it is just a matter of configuration.

Advantages:

1. Provides a powerful GUI environment.
2. Tracing and debugging are much easier than other simulators.
3. Accurately models most hardware and include the modeling of physical phenomena.

Limitations:

1. It does not offer a great variety of protocols and very few protocols have been implemented, leaving users with significant background work.
2. Poor analysis and management of typical performance.
3. The mobility extension is relatively incomplete.

2.7 OPNET

OPNET (Optimized Network Engineering Tools)[7] can be flexibly used to study communication networks, devices, protocols and applications. It offers relatively much powerful graphical support for the users. The graphical editor interface can be used to build network topology and entities from the application layer to the physical layer. Object-Oriented Programming technique[28] is used to create mapping from the graphical design to the implementation of the real systems. All topologies' configuration and simulation results can be presented very intuitively and visually. The parameters can also be adjusted and the experiments can be repeated easily through GUI.OPNET is based on a mechanism called discrete event system. Hierarchical structure is used to organize the networks. OPNET simulator is very useful when working with complex networks with a big number of devices and traffic flows or in networks where a little change could be critical[30]. Before implementing any change, it is possible to predict the behavior and to verify the configurations of the devices. OPNET has different tools (NetDoctor, ACE and MVI)that allow administrators to analyze their networks and the future implementations they want to do. In this part of the lab, students can evaluate these tools using complex scenarios provided by OPNET.

Type: Commercial.

Language: Flavor of C and C++[7].

Supported Operating System: Windows XP, Vista, 7 and Windows NT 4.0.

Hardware Requirement: 256 MB RAM and 200 MB disk space is required.

Features:

- OPNET inherently has three main functions: modeling, simulating and analysis.
- It provides intuitive graphical environment to create all kinds of protocol models.
- OPNET consists of high level user interface which is constructed from C and C++ source code blocks with a library of OPNET specific functions.
- It arranges its hierarchical structure model into three specific domains. Network domain, Node domain and Process domain.
- It is also possible to run external code component.
- It supports scalable wireless simulation.
- Model network protocols, resources, algorithms, applications and queuing policies in detail using OPNET Modeler's powerful object-oriented modeling approach.

Advantages:

1. Leverage three different simulation technologies to efficiently tradeoff simulations detail and speed.
2. Fast discrete event simulation engine.
3. Customizable wireless modeling.
4. Integrated GUI based debugging and analysis [28].

Limitations:

1. Complex GUI operation.
2. It does not allow much number of nodes within a single connected device.
3. Accuracy of results is limited by the sampling resolution.
4. Simulation is inefficient if nothing happens for long periods.

2.8 TOSSIM

TOSSIM is a discrete event simulator for TinyOS[8] sensor networks that provides several mechanisms for interacting with the network; monitoring packet traffic, statically or dynamically injecting packet into the network and invocation of TinyOS call. It (bit-level discrete event network emulator) has built-in Python and C++. It has mechanisms that allow GUIs to provide detailed visualization and actuation of a running simulation. It takes advantage of TinyOS structure and whole system compilation to generate discrete-event simulations directly from TinyOS component graphs. By exploiting the sensor network domain and TinyOS's design, TOSSIM can capture network behavior at a high fidelity while scaling to thousands of nodes. By using a probabilistic bit error model for the network, TOSSIM remains simple and efficient, but expressive enough to capture a wide range of network interactions. Applications can connect to TOSSIM over a TCP socket and send control messages to add, delete or modify network links. TOSSIM compiles TinyOS code directly. It has an extensible network, Analog-Digital-Converter (ADC) and spatial model interface. The default models provided are very simplistic, but the interfaces allow models of significant complexity if needed. In a word, TOSSIM is an extension for TinyOS Mote Simulator[8] that provide simple and powerful emulator for Wireless Sensor Network (WSN).

Type: Open Source

Language: Python, C++ and NesC

Supported operating systems: Linux Operating Systems, Cygwin on Windows.

Features:

- TOSSIM provides interaction with the networks due to its graphical support.
- Its architecture includes two different models. Radio models for all kinds of transmission aspects and ADC models.
- Packet can be dynamically injected into the network to monitor traffic easily.
- TOSSIM uses a very simple abstraction for a network signal; it is either a one or a zero. All transmission signals have equal strength and collision is modeled as a logic OR.
- TOSSIM provides approach to simulate the TinyOS sensor networks at bit level granularity.
- It is to provide a high fidelity simulation of TinyOS applications.
- It supportsthousand of Nodes[25].
- It allows users to modify the loss topology at runtime.

Advantages:

1. TOSSIM compiles TinyOS code directly.
2. It allows GUI to provide detailed visualization and actuation of a running simulation.
3. It is expressive enough to capture a wide range of network interactions.
4. It provides simple and powerful emulator for Wireless Sensor Network.
5. It explores configuration difficult to construct.
6. It observes interaction difficult to live-capture.
7. It also helps cost effective design and improvement.
8. In TOSSIM, hardware in the loop is possible[25].
9. As TOSSIM is Open Source, every online document is free of cost.

Limitations:

1. TOSSIM is not always the right simulation solution; like any simulation, it makes several assumptions and focus on making some behaviors accurate while simplifying others.
2. It is specially designed for TinyOS, not for simulation performance metrics of other new protocols.
3. In the simulation process, every node has to run on NesC code, that's why TOSSIM can only emulate the type of homogeneous applications.
4. In TOSSIM, radio models does not affect signal strength.
5. It may not represent accurate real-world result.

2.9 J-SIM

J-SIM is a Java-based simulation system for building quantitative numeric models and analyzing them with respect to experimental reference data. It has been built upon the notion of the autonomous component programming model.JSim models can inter mix Ordinary Differential Equations (ODEs), Partial Differential Equations (PDEs), implicit equations, integrals, summations, discrete events and procedural code as appropriate[9]. J-SIM's model compiler can automatically insert conversion factors for compatible physical units as well as detect and reject unit unbalanced equations. JSim also imports and exports model archived formats SBML and CellML. Components can be plugged in a software system, even during execution. The model defines the generic structure of a node (either an end host or a router) and the generic network components, both of which can then be used as base classes to implement protocols across various layers. The model is derived by featuring out the common attributes of network entities in the current best-effort Internet, it is general enough to accommodate other network architectures such as the IETF differentiated services architecture, the mobile wireless network architecture and the WDM-based optical network architecture.

Type: Open Source.

Language: Java.

Supported Operating System: Windows XP, Vista & 7, MAC OS X, Linux.

Hardware Requirement: 1.5 GB disk space of hard disk, At least 2GB memory to run J-SIM properly.

Features:

- It is loosely-coupled, component-based programming model. The ability to handle data in independent execution contexts along with the fact that components are loosely coupled and only bound to one another at system integration time, is the key reason that a component can be reused in other software systems with the same contract context.It isas the same fashion as IC chips are used in hardware design.
- It provides dynamic thread execution framework forreal-time process-driven simulation. In J-SIM, the simulation engine extends the Worker Pool class and monitors the activities of all Worker Threads[9]. It maintains a globally-observed, virtual system time that is proportional to the real time.
- Implementation of a complete suite of Internet Integrated/Differentiated/Best Effort Services protocols. The implication of the implementation is three fold: With all the Internet protocol classes available, the abstract classes and by virtue of the component hierarchy.
- A dual-language environment that allows auto-configuration and on-line monitoring. This closely mimics the IC debugging and testing process[25].
- Definition and implementation of generic interface classes for trace-driven simulation, with which (i) the network of interest can be emulated based on real network traffic loads and (ii) fidelity of *J-SIM* can be validated (and the *(in)accuracy* quantitatively characterized) through analysis/comparison of real network performance data and simulation data.

Advantages:

1. A simple and well-defined component-based software architecture with object oriented paradigm. This facilitates hierarchical modeling of complex systems[9].
2. Simulation engine is built in the runtime and is transparent to components.
3. It provides Process-oriented modeling andwait() methods, synchronization methods to further extend programming flexibility.
4. It can work with both discrete event simulation and real-time process-based simulation.
5. It will implement a parallel simulation in the autonomous component architecture

Limitations:

1. Java has some security restrictions. So forth JSIM can be prevented from persistence in data.
2. Operation of JSIM required clear concept of queuing algorithm.
3. The graphical model designer, which has limited capabilities as of now (can only be used to design a model), is intended to be a GUI-based model builder that would do much of the code generation that has to be currently done manually.

2.10 NCTUns

The NCTUns (National Chiao Tung University network simulator)[10][33] is an innovative tool that seamlessly integrates simulation and emulation of both wired and wireless to provide unique advantages over most existing network emulators[34]. The ancestor of the NCTUns is the Harvard network simulator, which was authored by S.Y. Wang in 1999. The Harvard network simulator has several limitations and drawbacks that need to be overcome and solved. Some useful features and functions need to be implemented and added to it. For these reasons, S.Y. Wang developed the NCTUns. It uses a distributed architecture to support remote simulations and concurrent simulations. It uses an open-system architecture to enable protocol modules to be easily added to the simulator. The GUI program contains four main components. They are the “Topology Editor,” “Node Editor,” “Performance Monitor,” and “Packet Animation Player,” respectively. Among these four components, the Node Editor is relevant to module developers.

Type: Open Source.

Language: C++.

Supported Operating System: FreeBSD, Fedora, Red hat, Ubuntu, and Debian.

Hardware Requirement: 1.6 GHz processor, 256 MB RAM, 300 MB on hard disk.

Features:

- It can easily be used as an emulator. This feature is very useful as function and the performance of real-

world devices can be tested under various simulated network conditions.
- It supports seamless integration of emulation and simulation. Real-life network traffic can pass through a complicated simulated network and interact with simulated network traffic.
- It can use any real-life UNIX network configuration and monitoring tools.
- It simulates various important networks.They are IEEE 802.11(b) wireless LANs, mobile ad-hoc (sensor) networks, GPRS cellular networks, optical networks (including both circuit-switching and burst-switching networks), IEEE 802.11(b) dual-radio wireless mesh networks, IEEE 802.11(e) QoS wireless LANs etc.
- It simulates various important protocols.For example: IEEE 802.3 CSMA/CD MAC, IEEE 802.11 (b) CSMA/CA MAC, IEEE 802.11(e) QoS MAC, IEEE 802.11(b) wireless mesh network routing protocol, DVB-RCS satellite MAC and PHY, spanning tree protocol, IP, Mobile IP, Diffserv (QoS), RIP, OSPF, UDP, TCP, RTP/RTCP/SDP, HTTP, FTP, Telnet, Bit Torrent, etc.
- It supports remote and concurrent simulations. The GUI and simulation engine are separately implemented and use the client-server model to communicate.This scheme can easily support the cluster computing model in which multiple simulation jobs are performed in parallel on different servers. This can increase the total simulation throughput.
- It is continuously supported, maintained and improved.New functions and network types are continuously added to NCTUns to enhance its functions, speed and capabilities.

Advantages:

1. It provides easy navigation GUI environment.
2. Realistic network traffic can be generated by realistic life applications to generate more realistic simulation results[34].
3. The performance of any real-life application can be easily evaluated on NCTUns under various simulated network conditions.

Limitations:

1. Connection through dispatcher with simulation server is not much stable.
2. The programming is not supported by the NCTUns. So, the parameters are needed to set via GUI.
3. The manipulation of every node has to be done node by node or all nodes in the same time.

2.11 DRMSim

DRMSim (Dynamic Routing Model Simulator) is a Discrete-Event Simulation (DES) software that aims at the evaluation of routing models at a large scale[11]. Its development has started in the context of a study focusing on dynamic compact routing. DRMSim is a Java based simulator software in which the design philosophy made to compromise on the quality of the code written so that extensibility and reusability are maximized. It is primarily a routing model simulation platform that provides adequate means for in-depth investigation of various routing schemes (including compact routing schemes). The main goal of DRMsim is to quantitatively evaluate the main performance metrics of routing schemes especially, the metrics related to the scalability and dynamic properties of these schemes[31].A careful analysis of the data structures that will be used in the network model as well as on the granularity and time management of the simulation model has been performed in order to reach the goal. It has been developed as part of a research project conducted by Alcatel-Lucent Bell, Universite de Bordeaux (LaBri) and INRIA labs at Sophia Antipolis (Mascotte project)

Type: Open Source

Language: Java

Supported operating systems: UNIX, Linux (Ubuntu, Fedora, CentOS), Mac OS.

Hardware requirements: Processor: 32-bit (x86 compatible) or 64-bit (x86-64 compatible),**Memory:** 4 GB free, **Disk:** 5 GB disk space.

Features:

- DRMsim addresses the specific problem of large-scale simulations of (inter-domain) routing models on large networks.
- It enables the fast simulation of routing schemes on large dynamic networks.
- DRMSim is a simulator that is made to compromise on the quality of the code written so that extensibility and reusability are maximized.
- It attempts to simulate up to ten thousand nodes on typical workstations.
- It maintains for every node the local knowledge of the whole network; this technical challenge is addressed by means of efficient data structures.
- The operation of a system is represented as a chronological sequence of events.
- It features efficient graph structures and algorithms as well as a lightweight (yet experimentally validated) implementation of the BGP routing scheme[31].

- DRMsim also provides an implementation of the NSR compact routing scheme that facilitates a tree-based routing scheme which takes advantage of the specific property of networks that have low(logarithmic) diameter and a high clustering coefficient (such as the Internet).

Advantages:

1. DRMsim is dedicated for a routing model simulation.
2. It provides efficient graph structures and algorithms.
3. It has capability to import external topologies (e.g., CAIDA maps).
4. It is a "100% pure Java" application which makes it executable on most platforms.
5. It is free of cost for research purposes.

Limitations:

1. By using DRMSim only the routing protocol can be simulated.
2. It does not make use of parallel/distributed discrete-event simulation techniques. It optionally relies on distribution for the parallel execution of simulation batches.
3. This software is packaged to be used on a machine must have at least 4G of memory. If memory isinsufficient, the performance of the software may decrease or may corrupt the simulation process.

2.12 SSFNet

SSFNet (Scalable Simulation Framework Network Models) is a discrete-event network simulator for network lab experimentation and research. It is a collection of Java SSF-based components for modeling and simulation of Internet protocols and networks at and above the IP packet level of detail [12]. SSF provides a single, unified interface for discrete-event simulation (the SSF API). The primary design goal of this framework is to support high performance simulation. Domain Modeling Language (DML) is used to model networks in SSFNet. DML has helped SSFNet users to create complex topology Internet models with 100,000 multi-protocol hosts and routers [35]. SSFNet organizes the processors by using multiple queues to order events. SSF uses multiple threads running in parallel to ensure the queues are executed in order. SSFNet hides all these details from the modelers that tell SSFNet how to partition the network between processors by using the alignment attribute in the DML file. It has two derivative frameworks, SSF.OS (for modeling of the host and operating system components, esp. protocols) and SSF.Net (for modeling network connectivity, creating node and link configurations) through which efforts have been made to enhance the modeling scalability or the number of nodes, traffic patterns, bandwidth, etc. The simulator can handle and perform scalability or the number of processors the simulator can run on.

Type: Open Source

Languge: java and C++

Supported operating system: Parallel execution under Linux, Solaris, and Windows NT using JDK1.2 and higher. Other platforms may support parallelism as well.

Features:

- SSFNet allows for multiple processor machines to simulate a network using multiple processors that increase performance significantly.
- Core Internet protocol models (IP, BGP4, OSPF, TCP, UDP), Sockets and various workload-generating client-server application models can be simulated using SSFNet[12]. Protocol validation tests are also included.
- It facilitates topological network component addressing and automated IP address allocation (CIDR compliant).
- SSFNet facilitates the management of global traffic patterns and also the Management of parallel random number streams employing a suite of strong random number generators and statistics from the CERN Colt package.
- SSF provides a single, unified interface for discrete-event simulation (the SSF API) in order to support high performance network simulation.
- Object-oriented models of SSF that utilize and extend the framework can be portable across SSF-compliant simulation environments which maximize the potential for direct reuse of model code while minimizing dependencies on a particular simulator kernel implementation.
- SSF makes it possible to build network models that are efficient and predictable in their use of space, able to transparently utilize parallel processor resources and scalable to very large collections of simulated entities [35].

Advantages:

1. Scalable high performance Java simulation platform.
2. Provides simple, standardized syntax for high-level model description DML.
3. Allow Management of global traffic patterns.
4. Support high performance network simulation.

5. It is free or available at nominal cost for research purposes.

Limitations:

1. Its slow convergence may occur in presence of long-range correlated traffic.
2. Understanding of scaling conditions: some emergent phenomena can be seen in sufficiently large networks, with sufficiently many traffic flows.
3. Need to understand relations between different abstraction levels.
4. Need to predict internet behavior under alternative-futures scenarios.

2.13 GrooveNet

GrooveNet is a hybrid simulator for geographic routing that address the need for a robust, easy-to-use realistic network and traffic simulation [13]. It is designed to be an opportunistic broadcast protocol with minimal handshaking between sending and receiving parties with little or no shared state information among neighboring vehicles. It facilitates communication between simulated vehicles, real vehicles and between real and simulated vehicles. It is able to form UDP connections with other vehicles, spanning multiple hops with the routing protocol. It can simulate and evaluate protocols across thousands of vehicles driving along US roads while communicating with neighbors within transmission range. GrooveNet is unique for its capacity to analyze the performance and scalability of inter-vehicular communication protocols[24], through realistic traffic density, speed, trip and communication models.

Type: Open Source.

Language: codedin C++ and Matlab provides GUI for drawing structures and graph.

Supported operating systems: Linux Operating Systems with kernel version-2.6*(Ubuntu-12.4 tested and SUSE).It also requires Qt 3.x graphic library.

Hardware Requirements: Processor: Intel® Pentium® II processor (or equivalent), 400 MHz or higher, **RAM:** 256 MB RAM required, 512 MB RAM recommended, **Hard Disk:** 100 MB available hard disk storage, Display: Display resolution 800 x 600.

Features:

- GrooveNet is a modular event-based simulator with well-defined model interfaces through which models can be added without concern of conflicts with existing models as dependencies are resolved automatically.
- It operates in five modes capable of actual on-road inter-vehicle communication, simulation of traffic networks with thousands of vehicles, visual play-back of driving logs, hybrid simulation composed of real and simulated vehicles and easy test-scenario generation.
- It can support thousands of concurrently moving and communicating vehicles where each vehicle may have its own mobility, trip and communication model.
- It supports three types of simulated nodes: (a) vehicles which are capable of multi-hopping data over one or more Dedicated Short-Range Communications (DSRC) channels, (b) fixed infrastructure nodes and (c) mobile gateways capable of V2V and vehicle-to infrastructure (V2I) communication. GrooveNet supports multiple message types such as GPS messages which are broadcast [24].
- It aims to be an opportunistic broadcast protocol with minimal handshaking between sending and receiving parties and with little or no shared state information among neighboring vehicles.
- Its modular architecture incorporates mobility, trip and message broadcast models over a variety of link and physical layer communication models.

Advantages:

1. GrooveNet is able to support hybrid simulation.
2. Supports multiple vehicle, trip and mobility models over a variety of network link and physical layer models.
3. It provides well-defined model interfaces that make easy to add different network models.
4. It also implements multiple rebroadcast policies to investigate the broadcast storm problem.
5. The well defined graphical user interface makes it easy to auto-generate simulations.

Limitations:

1. As GrooveNet is Open Source, every online document and online support is not available all the time.

2.14 TraNS

TraNS (Traffic and Network Simulation Environment) is an open source GUI simulation tool that integrates traffic and network simulators (SUMO and ns2) to generate realistic simulations of Vehicular Ad-hoc networks (VANETs) [14]. This architecture allows us to generate the mobility traces prior to the network simulation. We stress here that TraNS is the first open-source project that attempts to realize this highly pursued coupling for application-centric VANET evaluation. The goal of TraNS is to avoid having simulation results that differ significantly from those obtained by real-world experiments, as observed for existing implementations of mobile ad-hoc networks [27]. TraNS allows the information exchanged in a VANET to influence the vehicle behavior in the mobility model. It also provides a GUI that allows quick and simple setup of all the required simulation parameters like road network topology, simulation time, TCP communication ports, dump and scenario files, etc. Thus, the network simulator can use realistic mobility models and influence the behavior of the traffic simulator based on the communication between vehicles. **TraNSLite** is a stripped-down version of TraNS suitable for quickly generating realistic mobility traces for ns2 from SUMO.

Type: Open Source

Languge: Java and C++

Supported operating systems: Linux and Windows (trace-generation mode)

Features: The main features of TraNS v1.2 are:

- TraNs generate realistic simulations of VANETs.
- It has two distinct modes of operation. These are network centric and application centric.
- Mobility trace generation for ns2 from TIGER and Shape file maps (using the net convert tool from SUMO).
- Possibility to simulate road traffic events, e.g., accidents (both vehicle-related and location-related).
- Framework for developing new VANET applications.
- It provides two ready-to-use VANET applications: **Road Danger Warning** (safety) and **Dynamic Reroute** (traffic efficiency).
- Large scale simulations (tested up to 3000 vehicles).

Advantages:

1. It facilitates automated generation of random vehicle routes.
2. Map cropping and speed rescaling (*only for TIGER maps*)
3. Google Earth visualization of simulations (*only for TIGER maps*)
4. It provides TraCI interface [23] for mobility trace generation by coupling SUMO and ns2.

Limitations:

1. The development of **TraNS** is suspended. Hence, **TraNS** does not support the latest version of both Sumo and ns2.
2. Although free version of this type is available, No proper documentation is available for **TraNs** simulator.

Table (1):Table for Comparison of simulators based on general information General Information of simulators:-

SI. #	Name	License Type	Language	Supported Operating System	GUI Support	Document Available	Ease of Use
1	**Ns2**	Open source	C++ and OTCL	GNU/Linux,FreeBSD, Mac OS X, Windows XP, Windows Vista and Win. 7.	Limited	Excellent	Hard
2	**Ns3**	Open source	C++and Optional Python Bindings	GNU/Linux,FreeBSD, Mac OS X, Windows XP, Windows Vista & Windows 7.	Yes	Excellent	Hard
3	**QualNet**	Commercial (Separate license for academicians and others)	C++	UNIX, Window-MAC, Linux	Yes	Excellent	Moderate
4	**GloMoSim**	Open source	C	Windows, Linux, Sun SPARC Solaris	Limited	Poor	Hard
5	**NetSim**	Proprietary	C and Java	Windows (7, Vista) and windows XP	Yes	Excellent	Easy
6	**OMNET++**	Open source (for study and research	C++	Windows XP or Later, Linux, Mac OS X,	Yes	Good	Easy

		purpos-es),Commercial(for industrial purposes)		and other Unix-like system.			
7	**OPNET**	Commercial	C and C++	Windows XP, Vista, 7 & Windows NT 4.0.	Yes	Good	Easy
8	**TOSSIM**	Open source	Python, C++ and NesC	Linux Operating Systems or on Cygwin on Win-dows.	Yes	Good	Easy
9	**J-SIM**	Open source	Java	Windows XP, Vista & 7, MAC OS X, Linux.	Yes	Medium	Easy
10	**NTCUns**	Open source	C++	FreeBSD, Fedora, Red hat, Ubuntu, and Debian.	Yes	Good	Hard
11	**DRMSim**	Open source	Java	UNIX, Linux (Ub-untu, Fedora, Cen-tOS), Mac OS	Yes	Good	Moder-ate
12	**SSFNet**	Open source	Java and C++	Linux, Solaris, and Windows NT using JDK1.2 and higher	Yes	Good	Hard
13	**GrooveNet**	Open source	C++	Linux kernel ver-sion-2.6*(Ubuntu-12.4 tested and SUSE) and Qt 3.x graphic library	Yes	Medium	Easy
14	**TraNs**	Open source	Java and C++	Linux and Windows (trace-generation mode)	Yes	Poor	Moder-ate

Table(2):Table for Comparison of simulators based on the properties of simulators :-

SI. #	**Name**	**Simulation Event Type**	**Available Module**	**Scala-bility**	**Number of node support**	**Paral-lelism**	**Comment**
1	**NS2**	Discrete-event	Wired,Wireless, AdHoc and Wire-less Sensor Net-works	Limited	Up to 3000	No	Specially designed for protocol simula-tion and network research.
2	**NS3**	Discrete-event	Wired,Wireless, Adhoc and Wire-less Sensor Net-works	Limited	-------	-------	Targeted primarily for research and ed-ucational use and its performance is sta-ble.

3	**QualNet**	Discrete-event	Wired &Wireless(like WiFi, Sensor network,MANET,WIMAX)network	Large	500-20000	Yes(SMP/Beowulf	Unique platform for designing new protocol & analyzing their performance but costly
4	**GloMoSim**	Discrete-event	Wired, Wireless & Ad-Hoc Networks But currently pure wireless support	Large	Up to 10,000	Yes(SMP/Beowulf	It provides modular simulation for protocol stack and free for academic research but not updated regularly.
5	**NetSim**	Stochastic Discrete-Event	Wired &Wireless sensor network (wireless LAN, WiMAX)	Large Enough	-------	-------	It is a leading simulation software for protocol modeling and simulation, allowing us to analyze computer networks with unmatched depth, power and flexibility
6	**OMNET++**	Discrete-event	Wired, Wireless, Ad-hoc and Wireless Sensor Networks.	Enough	--------	MPI/PVM	It provides an extensible and component based analyzation and discrete event powerful simulation in academic and research.
7	**OPNET**	Discrete-event	Wired, Wireless, Ad-hoc and WSN.	Large	210 to 290 in all topologies	Yes	It provides flexible GUI to make easy of complex scenarios
8	**TOSSIM**	Bit level Discrete-event	TinyOS based wireless sensor network emulation	Enough	850	No[not sure]	Provide simple and powerful emulator for Wireless sensor Network.
9	**JSim**	Diverse numerical method	Wired and wireless sensor networks.	Enough	Maximum 1000 nodes for a tree	RMI based	Specially featured for generic network components.
10	**NTCUns**	Kernel reenteringmethod	Wired, Wireless, Ad-hoc and Wireless Sensor Networks.	Medium	Maximum 4096 nodes	------	It seamlessly integrates protocol modules and can be used for both MANET and VANET simulation perfectly.
11	**DRMSim**	Discrete-	Dynamic route	Large	Up to	No	It enables the fast

		Event	model simulation		10,000		simulation of routing schemes on large scale dynamic networks.
12	**SSFNet**	Discrete-Event	Modeling and simulation of Internet protocols and networks	Very Large	Up to 100,000	Yes	Unique to enhance the modeling scalability, traffic patterns, bandwidth, etc.
13	**GrooveNet**	Hybrid Simulator	Realistic network and traffic simulation	Large	--------	Yes	Unique for its capacity to analyze the performance and scalability of inter-vehicular communication protocols.
14	**TraNs**	Discrete-Event	Vehicular Ad-hoc networks (VANETs)	Large	Up to 3000	--------	Provides TraCI interface for mobility trace generation by coupling SUMO and ns2 with Google earth visualization.

CONCLUSION

The comparative study reveals that choosing a suitable, necessary and efficient simulator for the specific job of a research work has become easy. The Tables 1 and 2 have depicted the properties of the simulators clearly and extensively. The general information compiled in Table (1) e.g. license type, language, supported OS & GUI, document availability etc. are also useful along with the properties as compared in the Table(2). Some cells in the Tables (1& 2) are left blank because of lack of information found in related websites and documents. Even though, we think our study will become useful to the researchers in the computer communication fields.

REFERENCES

[1] The Network Simulator - ns-2. http://www.isi.edu/nsnam/ns/.
[2] NS3 official website, http://www.nsnam.org/documents.html.
[3] The QualNet, http://www.qualnet.com.
[4] The GloMoSim, http://en.wikipedia.org/wiki/GloMoSim
[5] The NetSim, http://en.wikipedia.org/wiki/NetSim
[6] OMNeT++ Community Site. http://www.omnetpp.org/.
[7] OPNET Modeler, http://www.opnet.com/.
[8] TOSSIM,http://tinyos.stanford.edu/tinyoswiki/index.php/TOSSIM
[9] J-sim Official. http://sites.google.com/site/jsimofficial/.
[10] NCTUns Official. http://nsl.csie.nctu.edu.tw/nctuns.html.
[11] The DRMSim, http://ercim-news.ercim.eu/en94/ri/drmsim-a-routing-model-simulator-for-large-scale-networks
[12] Scalable Simulation Framework (SSF), SSFNet homepage, http://www.ssfnet.or/homePage.html.
[13] The GrooveNet, http://www.groovenet.htm/
[14] Traffic and Network Simulation Environment http://wiki.epfl.ch/trans/
[15] OTcl, http://otcl-tclcl.sourceforge.net/otcl/.
[16] Nam: Network Animator. http://www.isi.edu/nsnam/nam/.
[17] Xgraph. http://www.isi.edu/nsnam/xgraph/index.html.
[18] Gnuplot Homepage. http://www.gnuplot.info/.
[19] Network Simulators: http://www.nsnam.com/
[20] The REAL network simulator.http://www.cs.cornell.edu/skeshav/real/overview.html.
[21] Sujata V. Mallapur, Siddarama . R. Patil, "Survey on Simulation Tools for Mobile Ad-Hoc Networks", *IRACST International Journal of Computer Networks and Wireless Communications (IJCNWC)*, ISSN: 2250-3501 Vol.2, No.2, April 2012.
[22] Mrs. Saba Siraj ,Mr. Ajay Kumar Gupta, MrsRinku-Badgujar," Network Simulation Tools Survey", *International Journal of Advanced Research in Computer and Communication Engineering*, ISSN : 2278–1021,Vol.1, Issue4, June 2012.
[23] Aamir Hassan, "VANET Simulation", *Technical report*, IDE0948, May 2009.
[24] Francisco J. Martinez, Chai KeongToh, Juan-Carlos Cano, Carlos T. Calafate and Pietro Manzoni," A survey and comparative study of simulators for vehicular adhoc networks (VANETs)", *WIRELESS COMMUNICATIONS AND MOBILE COMPUTING Wirel. Commun. Mob. Comput. (2009) Published online in Wiley InterScience(www.interscience.wiley.com)* DOI: 10.1002/wcm.859
[25] Mrs. PoonamChhimwal, Dhajvir Singh Rai, DeepeshRawat, "Comparison between Different Wireless Sen-

sor Simulation Tools", *IOSR Journal of Electronics and Communication Engineering (IOSR-JECE)* e-ISSN: 2278-2834,p-ISSN: 2278-8735. Volume 5, Issue 2 (Mar.-Apr. 2013), PP 54-60 www.iosrjournals.org.

[26] Martin J. GloMoSim. Global mobile information systems sim-ulation library. UCLA Parallel Computing Laboratory, 2001.Available at: http://pcl.cs.ucla.edu/projects/glomosim

[27] Michał Pi ´ orkowski, Maxim Raya, Ada Lezama Lugo, PanosPapadimitratos, and Jean-Pierre Hubaux, "TraNS: Realistic Joint Traffic and Network Simulator forVANETs", *Available at:http://icapeople.epfl.ch/panos/trans_poster.pdf*

[28] Luc Hogi , Pascal Bouvry, Fr´ed´ericGuinand,"An Overview of MANETs Simulation", Available at: www.researchgate.net

[29] Mohammad ShahidulHasan,ChristopherHarding,Hongnian Yu, Alison Griffiths, " Modeling Delay and Packet Drop in Network Control Systems Using Network Simulator NS2 ", International Journal of Automation and Computing 2(2005) 187-194

[30] JarmoProkkola, "Simulations and Tools for Telecommunications 521365S: OPNET - Network Simulator","VTT Technical Research Centre of Finland", University of Oulu. April, 2006.

[31] Luc Hogie∗, Dimitri Papadimitriou, IssamTahiri, " Simulating Routing Schemes on Large-Scale Topologies", Available at: www.irisa.fr/prive/Federic.Majorcayk/articles/Hal01.pdf.

[32] J. Lessmann, P. Janacik, L. Lachev, and D. Orfanus. "Comparative Study of Wireless Network Simulator", The Seventh International Conference on Networking, pages 517-523, 2008.

[33] Shie-Yuan Wang*,y and Yi-Bing Lin, "NCTUnsnetworksimulation and emulationfor wireless resource management", Wirel. Commun. Mob. Comput. .Page No. 899–916, 2005.

[34] S.Y. Wang and Y.M. Huang, " NCTUns Distributed Network Emulator," Internet Journal, Vol. 4, Num. 2, pp. 61-94, Nova Science Publisher (ISSN 1937-3805), 2012.

[35] Prof. Dr. MesutGüneş ▪ Ch. 4 "Introduction to network simulators",Freie University, Berlin