A Comprehensive Overview on Different Network Simulators

Christhu raj M.R[#1], Namrata marium Chacko[#2], John major [*3], Shibin. D [#4]
[#1 2 4,] Department of Information Technology, Karunya University
Coimbatore, India
[1] mrchristhuraj@gmail.com
[2] namratamarium@gmail.com
[4] Shibin.ku@gmail.com
[*3] Department of Electronics and Communication, Karunya University
Coimbatore, India
[3] johnmajor010@gmail.com

***Abstract*— Network simulators are used worldwide for education, commercial and Industrial purposes to simulate any part or entire network. There are numerous network simulators used day by day. Network simulators can be used to produce an approximation result of the network, which lays foundation for real time application or implementation. Researchers are not aware about facility and various network simulators available. Our Work gives an overview on different network simulators.**

Keywords- ns2 , ns3 , Qualnet , OPNET, OMNET++, TrSiM , Agent J

I. INTRODUCTION

Computer networks are being widely used in all fields of life, Net banking, Military operations, disaster relief and social networking. As the usage of various computer networks grows so does the need to make these networks faster, efficient and secure greatly increase [1][2]. Various research and studies leads to development of various new protocols claiming to be more efficient than the other. A network is more than just end user devices; it compromises of routers, gateways, servers as hardware components, several routing protocols, security mechanisms, Error correction and detection techniques. When we consider all these components, setting up a network is quite a costly deal. Besides cost; the factors that need to be taken into considerations are, efficiency, security, speed etc. therefore when introducing a new techniques into a network is quite a risk. The size of the computer network, makes to almost impossible to conduct experiments directly on it, this could lead to serious damage of the network and loss of data and would require a lot of effort and cost for the damage recovery. Consequently there is a need to have test beds which can accurately imitate the working and interactions of various network components. As stated in [3] large networks exhibit very complex behaviour as a result of three basic factors:

(1) Subtle protocol interactions,

(2) Complicated network topologies, and

(3) Complex traffic patterns.

These factors have an obvious impact on simulation: we must be able to simulate topologies with hundreds of hosts, and the simulator must be able to recreate the traffic patterns found on real networks. Simulating protocols is the most difficult. This is because most protocols have different behaviours and different mechanisms of operation and when a new protocol is introduced to existing mechanisms the entire behaviour of the network may be affected. Hence it is crucial to have tested the proposed mechanisms before they are actually deployed into a network, this can be done through simulation of the networks using some tools, which many be either hardware or software. The networks simulation techniques many vary for different simulators. The behaviour of the network is analysed based on the interaction between various network entities and compared with mathematical formulas or graphs [4]. A simulated environment is like a test bed, where the behaviour of the networks can be observed when some parameters of the network are modified.

To simulate and get nearly accurate results we require powerful simulation tools, these tools are known as network simulators. A network simulator is a piece of software or hardware that predicts the behaviour of a network, without an actual network being present [5].Since a network simulator imitates the working of a computer network, it is a well suited tool to test all the newly proposed mechanisms. One can define the various types of devices used in the network and also model the network links, capacity of the links, the traffic load capacity, the type of traffic, behaviour of each device and also various attacks can be simulated to observe the behaviour in the presence of an attacker. Today there exist various simulators; some are NS2, OPNET, SENSE and Net sim. These network simulators are mostly GUI driven and can be easily installed on a desktop pc or a laptop. Some network simulators require input scripts, some other require command and still other are fully GUI based where the user can click and drag network components into a simulated topology example of one such

simulator is OPNET. In a simulator one can define the placement of various network components like nodes, servers, routers, gateways and links. Also various events can be defined like packets to be sent, packets to be drooped, Time intervals and various attacks. Cryptographic operations can also be defined into the protocols and thus one can observe the outcome on the packet size, delivery ration, number of packets dropped and traffic load in the network.

The results of the simulation may be in the form of graphs, animations of a trace file. A trace file can record every event that has occurred in the simulation. The graphs are used as a comparison technique between various protocols. Graphs can show the changes in the packet delivery ration, throughput of the network, delay of the network and many other parameters used to measure network performance. Most network simulators are discrete event simulation, which means that before the execution of the protocol the user defines a list of events to be triggered at particular time periods or some conditional events, which are stored. These stored events are then processed in order when the command to begin the simulation is given.

II. NETWORK SIMULATORS

A. NS-2

We start our survey with one of the famous network simulator "ns2". Statistics says about 70% education purposes use ns2. It is open source, discreet event simulators for computer networks [6]. Ns2 code comprises of OTCL and C++. OTCL is an interpreter used to execute the commands. NS2 follows two levels of hierarchy namely C++ Hierarchy and the interpreted OTcL, which is one to one correspondence [7]. Two languages are linked because to achieve efficiency. C++ Hierarchy allows faster execution and to achieve efficiency. This gives detailed description, definition and operation of protocols, packets and processing time. On the other hand OTcL enables user to define network topology, protocols, applications that user tend to simulate. OTcL can make use compiled C++ Object through an OTcL linkage [9].OTcL Linkage creates a matching between OTcL and C++ Objects. Whenever you run a tcl file it will produce two outputs or two files namely trace file and namfile. Trace file differs for both wired and wireless scenario. It defines the event discreet simulators. It records the data for each millisecond and gives an output regarding packets send, received, dropped, initial energy of nodes, consumption of energy for transmitting, receiving, idle power, sleep power. It denotes the traffic model, simulation packets, packet size, and mac address. Nam file is a visual graphical window which shows the node movements, radio range, and packet transfer including time. Trace file can be given input to a new scenario file called NS- VISUAL TRACE ANALYZER [8].

Apart from ns2 tcl code can be generated using two most important network tools like

1. nscript
2. NSG JAR File
3. Mannasim generate
4. TCL 830.exe

The above tools are user graphic tool to generate tcl node. The drag and drop components like Node, link, traffic, applications, parameters are predefined. NSG Jar file is a java applet file which has inbuilt java code for nodes, links, application, agent and parameters. Fig 1 shows the user interface module of NSG Jar window to create tcl codes. As explained has different drop-down boxes with options to create nodes , agents , link and parameters like range , energy , Link layer , antenna can be modified. nscript and Mannasim are similar like NSGJAR and drawback for those three tools is that we can't add any new protocols, C++ Code into the files. Ns2 can be executed only in Linux platform and windows needs Cygwin platform. To execute ns2 in windows environment we need a special tool called TCL 830.Exe, nam.exe and ns.exe. It enables to run the tcl command in windows command prompt. Again the main drawback of the tool, we can't include C++ Codes. The above tools are not simulators, but just an alternative and easy solution to generate tcl codes. For adding new routing protocols, modifying existing protocol we should stick to ns versions like ns2.29, 2.34 2.35. TCL codes were created by John Ousterhout.

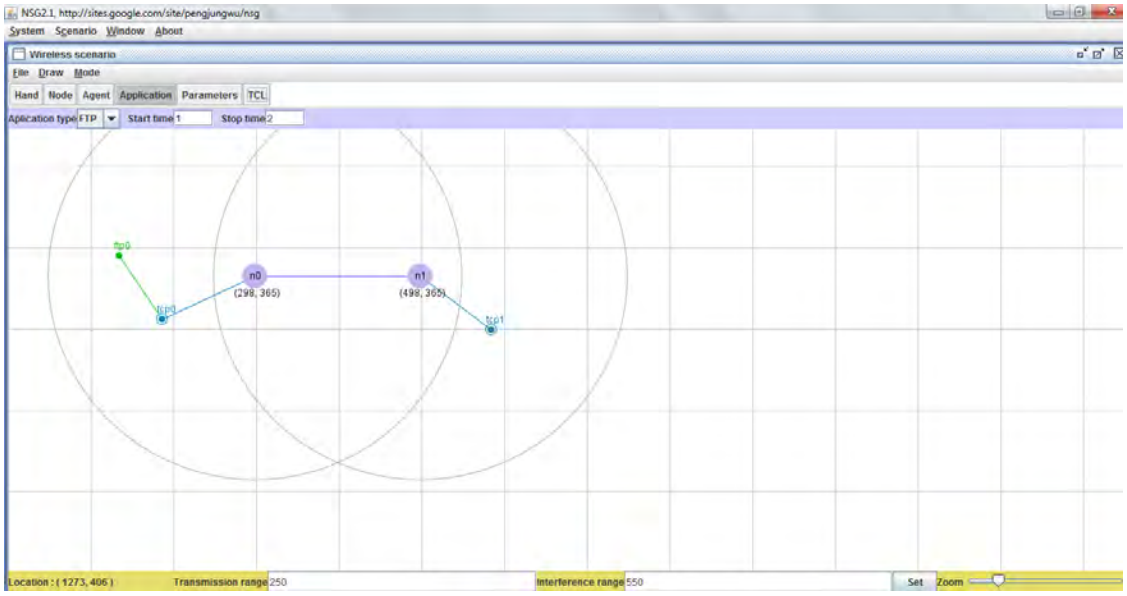


Fig 1. NSG Jar User interface to generate tcl codes

B. NS-3

Ns-3 is a discreet-event simulators primarily targeted for research and educational purposes. It was started in 2006 [11] and NS-3 is not an extension of ns2. NS3 is a new simulator. The similarity between ns2 and ns3 are both written in C++ Codes but ns3 does not support ns2 API. In ns3 simulators is written in C++ and python. The new modules ns3 has when compared with ns2 are as follows

1. NS 3 can handle multiple interfaces or nodes correctly [11]
2. Use of IP Addressing and more alignment with Internet protocols and more detailed 802.11 models[11]
3. Ns2 can be ported to ns3.

The initialization and termination in ns3 is done by using *run()* and *stop()* commands. To free memory Destroy() method is used. The structure of ns3 is as follows [10]

1. Initialization and termination of ns3 objects
2. Definition of Network Topology
3. Transport protocols and applications in ns3
4. Scheduling events in ns3
5. Tracing events in ns3

There are seven logs messages levels namely [6], NS_LOG_ERROR, NS_LOG_WARN, NS_LOG_DEBUG, NS_LOG_INFO, NS_LOG_FUNCTION, NS_LOG_LOGIC, and NS_LOGIC_ALL

C. TrSim

TrsSim (Trust and Reputation Model simulators for Wireless sensor networks) is java based simulator used to calculate the trust and reputation in Wireless sensor networks.[12] It provides various trust schemes and reputation models and new ones can be added. It allows used to define whether network is static or dynamic and enables the researchers to test their reputation and trust models against a range of applications. It gives percentage of selfish or malicious nodes, percentage of nodes acting as client or server [12]. Fig 2 shows a user interface module of TrSim simulator.

D. Agent J (Lan Taylor)

Agent J is a java based network simulator for ns2. AgentJ is essentially java virtual machine (JVM) [13] for simulation environment. Agent J Consists of byte code subsystem that swaps used code on the use. It contains own implementation of network, threading, timing functions for ns2 [13]. Agent j contains implementation for the java.net package through a tool kit called Protolib, which is used to integrate with ns2.

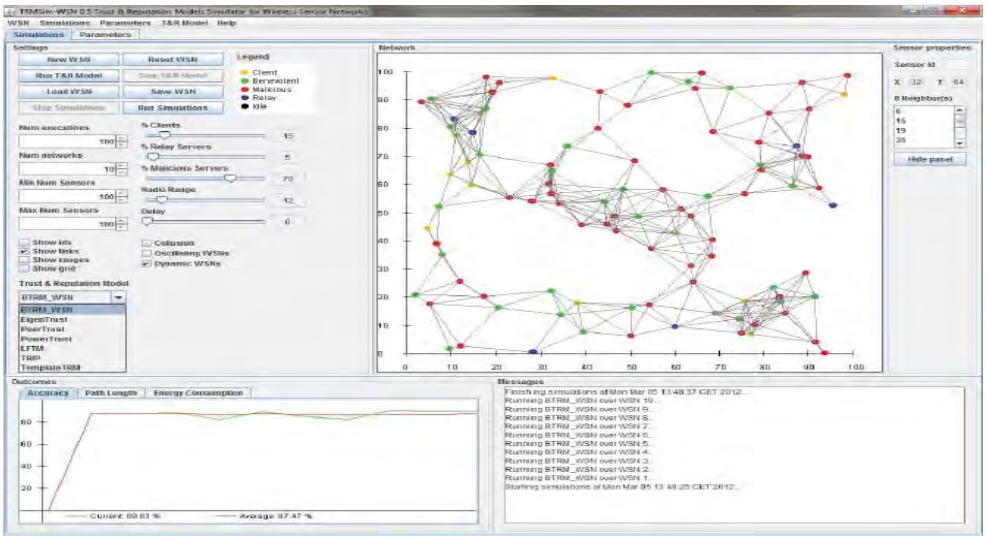
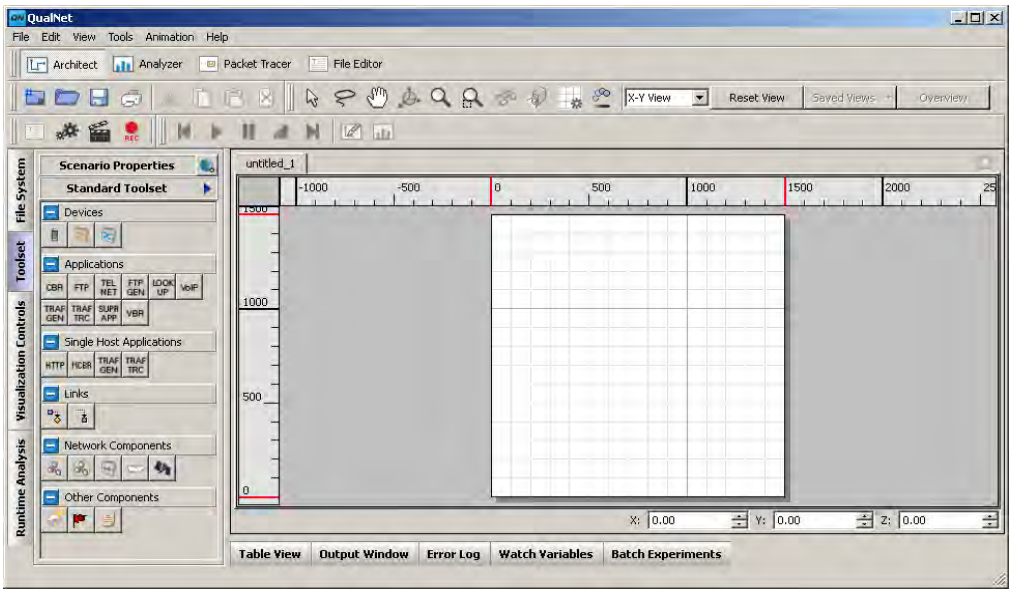


Fig 2. TrSim User Interface module

E. Qual Net

It is a discreet event simulator and mainly successor of GloMoSim. It was particularly built for large Scale networks. We can simulate wired, wireless and mixed networks. The main features of Qual net [14] Robust set of wired and wireless network protocols which is useful for simulating networks. Qual net executes equivalent networks 5-x times faster than other networks. Qual Net runs on all platforms like Linux, windows. Qualnet needs a single parameter, which sets up the experiment or network called as *default.config.*



F. OPNET

OPNET is another network simulator, which again a best User Interface. It can be downloaded in IT Guru website. Test technology design in realistic conditions and evaluate enhancement to standard based protocols, develop new protocols and technologies [15].OPNET supports four simulation technologies as

1. DISCREET EVENT SIMULATOR
2. FLOW ANALYSIS
3. ACE QUICK PREDICT
4. HYBRID SIMULATION

Discreet event simulators provide models that explicitly simulate protocols and simulate messages. [15].It executes in same way as production environment. Flow analysis provides analytical techniques and algorithms to model Stead state network behaviour. [15]ACE Quick Predict uses an analytical technique for studying the impact on application response time of changing network parameters (e.g., bandwidth, latency, utilization,

packet loss) this technique is supported within the OPNET Application Characterization Environment (ACE). OPNET is a high level event based network level simulation tool .Simulation operates at “packet-level” .originally built for the simulation of fixed networks. OPNET contains a huge library of accurate models of commercially available fixed network hardware and protocols. Nowadays the possibilities for wireless network simulations are also very wide .Accurate radio transmission pipeline stage for the modeling of the physical layer (radio interface) .The simulator has a lot of potentiality but here exists typically a lack of the oneself. OPNET can be used as a research tool or as a network design/analysis tool (end user).The threshold for the usage is high for the developer but low for the end user.

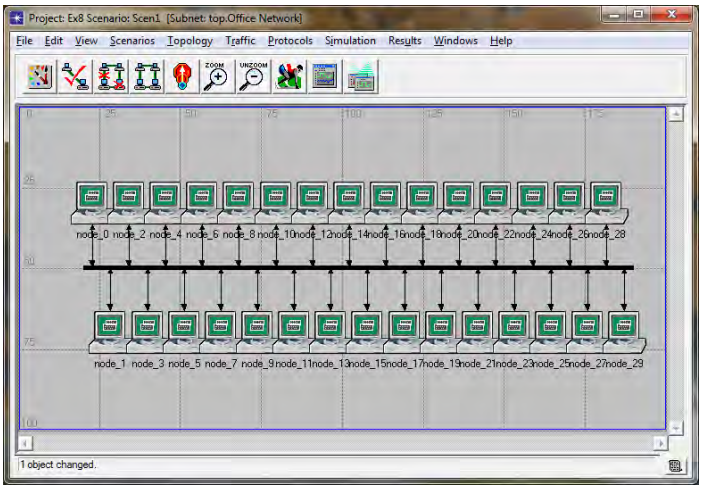


Fig 4. OPNET User Interface

G. OMNET ++

OMNeT++ is a discrete event simulation environment. Its primary application area is the simulation of Communication networks, but because of its generic and flexible architecture, is successfully used in other areas like the simulation of complex IT systems, queuing networks or hardware architectures as well. OMNeT++ provides component architecture for models. Components *(modules)* are programmed in C++, and then assembled into larger components and models using a high-level language *(NED)*. Reusability of models comes for free. OMNeT++ has extensive GUI support, and due to its modular architecture, the simulation kernel (and Models) can be embedded easily into your applications. Although OMNeT++ is not a network simulator itself, it is currently gaining widespread popularity as a network simulation platform in the scientific community as well as in industrial settings, and building up a large user community. The components of OMNeT++ are as follows simulation kernel library compiler for the NED topology description language, OMNeT++ IDE based on the Eclipse platform GUI for simulation execution, links into simulation executable (Tkenv) command-line user interface for simulation execution (Cmdenv) utilities (makefile creation tool, etc.) documentation, sample simulations.

H. SHARPE

Sharpe tool is an old network tool and well known package in reliability and perform ability [16]. It is one of the important tool and modeling tool. It is a toolkit that provides a specification language and solution methods for most of the commonly used model types for performance, reliability and per formability modelling.[17]. To increase the usability of this modelling tool, a graphical user interface (GUI) for SHARPE has been implemented. The SHARPE GUI implements eight interchangeable modelling description techniques for reliability engineering: fault trees, Markov chains, reliability block diagrams, reliability graphs, generalized stochastic Petri nets, product queuing networks, multi-chain product form queuing networks and task graphs. Future, all the modelling description techniques contained in SHARPE will be available in the GUI (phase mission, multi-components fault trees, and semi-Markov chains). The hierarchy feature is also implemented in the GUI. Java is the language chased for GUI implementation. [17][20].

I. SPNP

This package was developed by Ciardo [19]. The model type used for input is a stochastic reward net (**SRN**). SRNs incorporate several structural extensions to **GSPNs** such as marking dependencies, marking dependent arc cardinalities, guards and allow reward rates to be associated with each marking. The reward function can be marking dependent as well [18]. They are specified using **CSPL**, C based SRN Language which is an extension of the C programing language with additional constructs for describing the SRN models. SRN specifications are

automatically converted into a Markov reward model which is then solved to compute a variety of transient, steady-state, cumulative, and sensitivity measures [19][20].

J. SENSE

Sensor network Simulator and Emulator (SENSE) address the problem left by ns2 likeobject-oriented design that introduces much unnecessary interdependency between modules by binding functionalities with types. SENSE is designed to be an efficient and powerful sensor network simulator and ease of use. The three major factors of sense are Extensibility, scalability, usability G-SENSE is first graphical user interface developed for SENSE Simulator. SENSE Simulator has been used to implement protocols like SSR (Self –Selective Routing) , SHR (Self-Healing Routing), Self –Selective Reliable path(SSRP).[21] SENSE, has been developed for simulating wireless sensor networks.[21] The primary design goal is to address three factors explained above like extensibility, reusability, and scalability, and to take into account the needs of different users. The recent progresses in component-based simulation, namely the component-port model and the simulation component classification, provided a sound theoretical foundation for the simulator. Practical issues, such as efficient memory usage, sensor network specific models, [21] were also considered. Consequent, SENSE becomes an ease-of-use and efficient simulator for sensor network research. [21]

K. Query Cycle

Query Cycle is a common file sharing simulators. It has realistic models for content distribution, query activity and download behavior. [22] Content distribution is mainly based upon model where each file belongs to one category and category is defined by the popularity of the file. [22] Simulations proceed in query cycle representing the time period between issuing a query and receiving a response [22]. The queries are passed out in First-In First-Out basics and it is used for Simulating P2P Networks.

L. Maisie

Maisie is C- Based simulation language used for parallel and sequential execution of discrete-event simulation modules. It is used as a parallel programming language [23]. The Maisie Language begins describing the Maisie language constructs interactively building a small example program and advanced Maisie facilities a simulation model may be used to predict the behavior of a physical system under a variety of operating conditions. In the process-interaction approach to simulation, a physical system is assumed to consist of a set of physical processes that interact with each other at discrete points in time; these interactions are referred to as events. [23] In its simulation model, a logical process (LP) is used to model one or more physical processes (PP) [24] the events in the physical system are modeled by message exchanges among the corresponding logical processes in the model

M. Neurogrid

Neurogrid [25] is a routing technique for peer-to-peer networks. Each node in a Neurogrid searches the network by forwarding queries to a subset of nodes that may be able to match the query. Neurogrid operates under the assumption that objects in the network (e.g. documents) are referenced by a number of 'keywords'. Each node maintains a knowledge base of keyword-node associations that are based on the nodes belief about the contents of remote nodes. So, for example, given that a node receives an incoming search consisting of keywords A, B, and C, the node will consult its knowledge base and retrieve any remote nodes that are associated with these keywords. When the nodes match to a degree it is ranked likewise. Neuro grid search results will help each node to update the knowledge database and thus facilitate better forwarding of document throughout the network. Before set up of nodes, each node is given knowledge of its immediate neighbours. Neurogrid nodes also utilize the results of searches in order to update their knowledge bases and add new connections to the nodes that provide results to search queries. When the search is successful then the nodes update their knowledge base and associates the remote node with the key works, this is the adaptive nature of Neurogrid The querying node also establishes a direct link to the remote node, adding to the systems initial random connections, leading to a gradual increase in connectivity. One major flaw of this is that all of the nodes in the system gradually become a little smarter, being able to route queries to more and more nodes.

N. Netsim

Netsim [26] is a stochastic discrete event simulator which allows for simulation of various networks including Wireless sensor networks, wirelesses LAN, WiMAX, TCP, IP networks. This simulator was first developed by Tectcos in association with Indian Institute of Science. Netsim is capable to simulate various levels of the network, the entire network, a sub network, each node definition and also packet trace. The Netsim has a library of inbuilt protocols, which can be used directly or it can also be modified as it is in simple C code. The GUI of the Netsim is an inbuilt development environment which serves as an interface between the protocol library and the user code.

O. Tossim

TOSSIM [27] is a library: you must write a program that configures a simulation and runs it. TOSSIM supports two programming interfaces: Python and C++. Python allows you to interact with a running simulation dynamically, like a powerful debugger. However, since the interpreter can be a performance bottleneck when obtaining results, TOSSIM also has a C++ interface. Usually, transforming code from one to the other is very simple. When you start TOSSIM, no node can communicate with any other. In order to be able to simulate network behaviour, you have to specify a *network topology*. Internally, TOSSIM is structured so that you can easily change the underlying radio simulation, but that's beyond the scope of this tutorial. The default TOSSIM radio model is signal-strength based. You provide a set of data to the simulator that describes the propagation strengths. You also specify noise floor, and receiver sensitivity. There are some very early results that describe current sensor platforms (e.g., the mica2) in these terms. Because all of this is through a scripting interface, rather than provide a specific radio model, TOSSIM tries to provide a few low-level primitives that can express a wide range of radios and behaviour.

P. J-Sim

J-Sim [28] is a component based network simulator which is developed in Java. *J-Sim* is built upon a software based component architecture know as *autonomous component architecture (ACA).J-Sim* allows each individual component to be independently designed, implemented and tested. Components communicate with each other by sending and receiving data at ports. Since *J-Sim* is built upon ACA and is implemented in Java it becomes platform independent. It also has an extensible and reusable environment. *J-Sim* can be easily integrated with different script languages such as Perl, TCL and Python. *J-Sim* has the classes written in JAVA and this can be accessed using the TCL script just as done in NS2. Another salient feature of *J-Sim* is that public classes, methods and fields can directly be called from the TCL environment.

Q. GloMoSim

GloMoSim [29] is a network simulator which tries to mimic the various layers of the OSI layers. In GloMoSim, each entity represents a geographical area of the simulation. Hence the network nodes which a particular entity represents are determined by the physical position of the nodes. Since GloMoSim implements different levels new protocols at various levels can be easily integrated. There are also various functions which can be used to send message through different layers. GloMoSim runs only on Parsec compiler. New protocols should be added in Parsec and C language. Parsec code is used extensively in the GloMoSim kernel.

R. INSANE

INSANE is a network simulator designed to test various IP-over-ATM algorithms with realistic traffic loads derived from empirical traffic measurements. INSANE's ATM protocol stack provides real-time guarantees to ATM virtual circuits by using Rate Controlled Static Priority (RCSP) queuing. ATM signaling is performed using a protocol similar to the Real-Time Channel Administration Protocol (RCAP). Internet protocols supported include large subsets of IP, TCP, and UDP. In particular, the simulated TCP implementation performs connection management, slow start, flow and congestion control, retransmission, and fast retransmits. Various application simulators mimic the behavior of standard Internet applications to provide a realistic workload, including: telnet, ftp, WWW, real-time audio, and real-time video .INSANE is designed to run large simulations whose results are processed off-line. It works quite well on distributed computing clusters (although simulations are all sequential processes, a large number of them can easily be run in parallel).Although there is no graphical user interface, a (optional) Tk-based graphical simulation monitor provides an easy way to check the progress of multiple running simulation processes. The bulk of INSANE is written in C++. Customization and simulation configuration is performed with Tcl scripts.

III CONCLUSION AND FUTURE SCOPE

The other network simulators we left to explain are GosIP, NetViz, NIST, REAL, NEST, PTOLEMY with TCP Simulation. The various network simulators are explained with some user interface figure. Various network simulators enable to simulate the network but only very few simulators are used by researchers. Our work stated many simulators apart from the traditional network which will explain network researchers. The future work may include explaining new simulators which we stated above.

IV ACKNOWLEDGMENT

We would like our mentor Mr. Edwin prem Kumar .G(Assistant Professor, Karunya University , Coimbatore India) and Kartheek Kusumpudi(PG Scholar Department of Information Technology, Karunya University) who were grateful for us in completing this paper. We thank for their encouragement and effort took to give us idea in completing a paper

V REFERENCES

[1] Kotilainen N., Vapa M., Weber M., Töyrylä J. and Vuori J., "P2PDisCo Java Distributed Computing for Workstations Using Chedar Peer-to-Peer Middleware", *Proceedings of the 19th IEEE International Parallel & Distributed Processing Symposium (IPDPS 2005)*, Denver, Colorado,USA, 2005.

[2] Lv Q., Cao P., Cohen E., Li K. and Shenker S., Search and Replication in Unstructured Peer-to-Peer Networks, *Proceedings of the 16th International Conference on Supercomputing*, ACM Press, 2002, 84-95.

[3] Vapa M., Kotilainen N., Auvinen A., Kainulainen H. and Vuori J., "Resource Discovery in P2P Networks Using Evolutionary NeuralNetworks", *International Conference on Advances in Intelligent Systems- Theory and Applications (AISTA 2004)*, Luxembourg, 2004.

[4] Yang W. and Abu-Ghazaleh N., "GPS: A General Peer-to-Peer Simulator and its Use for Modeling Bit Torrent", *Proceedings of the 13th Annual Meeting of the IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems(MASCOTS '05)*, Atlanta, USA, 2005.

[5] Ting N. and Deters R., "3LS - A Peer-to-Peer Network Simulator",*Proceedings of the 3rd International Conference on Peer-to-Peer Computing (P2P 2003)*, IEEE Press, 2003, 212-213.

[6] Neal Charbonneau nealc.com Using Tcl and ns2 01/2010 Tcl and ns2: Tutorial

[7] NS Simulation for Beginners, Lecture notes, 2003 -2004 by Eitan Altman ,Tania Jimenez

[8] Fernando Rocha "Ns2 Visual Trace Analyzer Manual of ns2- Visual Trace Analyzer 0.272"

[9] The *ns* Manual (formerly *ns* Notes and Documentation)1 The VINT Project "A Collaboration between researchers at UC Berkeley, LBL, USC/ISI, and Xerox PARC. Kevin

[10] Eitan Altman, Tania Jimenez "NS Simulator for Beginners" by Morgan and Claypool publishers

[11] NS-3 Simulator ns-3 Tutorial Release ns-3.15 ns-3 project November 13, 2012

[12] http://ants.dif.um.es/~felixgm/research/trmsim-wsn/

[13] Ian Taylor *A PROTEAN Research Group Project Naval Research Laboratory*, Code 5522. "AgentJ Java Network Simulations in NS-2 -An Installation and User Manual"

[14] Scalable Network Technologies, Inc. 11022 Santa Monica Blvd., Suite 260 Los Angeles, California, 90025 "QualNet Simulator user manual version 3.1"

[15] Roman Dunaytsev Department *of Communications Engineering Tampere University of Technology*: Network Simulators OPNET Overview and Examples

[16] C.Hirel, R. Sahner ,X.Zang, K..Trivedi Department *of electrical and computer engineering* "Reliability and Perform ability Modelling using SHARPE2000"

[17] C.Hirel, R. Sahner ,X.Zang, K..Trivedi Department *of electrical and computer engineering* "SPNP Stochastic Pertri Nets 6.0 "

[18] http://people.ee.duke.edu/~chirel/IRISA/sharpeGui.html

[19] http://www.ee.duke.edu/~chirel/IRISA/spnp.html

[20] http://people.ee.duke.edu/~kst/software_packages.html

[21] Gilbert Chen, Joel Branch, Michael J. Pflug, Lijuan Zhu, and Boleslaw K. Szymanski Department of Computer Science, *Advances in Pervasive Computing and Networking , Springer ,NewYork* "SENSE"- A Wireless Sensor Network Simulator

[22] Niko Kotilainen, Mikko Vapa , Teemu Keltanen, Annemari Auvinen and Jarkko Vuori P2PRealm – Peer-to-Peer Network Simulator

[23] http://may.cs.ucla.edu/projects/maisie/tutorial/simulation/

[24] http://may.cs.ucla.edu/projects/maisie/

[25] Sam Joseph Ph.D., NeuroGrid, http://www.neurogrid.net/publications/publications.html

[26] en.wikipedia.org/wiki/NetSim

[27] docs.tinyos.net/tinywiki/index.php/TOSSIM

[28] Ahmed Sobeih, Wei-Peng Chen, Jennifer C. Hou, Lu-Chuan Kung , Ning Li , Hyuk Lim , Hung-Ying Tyan , and Honghai Zhang " J-Sim: A Simulation Environment forWireless Sensor Networks" Proceedings of the 38th Annual Simulation Symposium (ANSS'05) 2005 IEEE

[29] pcl.cs.ucla.edu/projects/glomosim/GloMoSimManual.html