A Survey of Network Simulation Tools: Current Status and Future Developments

Jianli Pan, jp10@cse.wustl.edu (A project report written under the guidance of Prof. Raj Jain)

Download

Abstract

In the network research area, it is very costly to deploy a complete test bed containing multiple networked computers, routers and data links to validate and verify a certain network protocol or a specific network algorithm. The network simulators in these circumstances save a lot of money and time in accomplishing this task. Network simulators are also particularly useful in allowing the network designers to test new networking protocols or to change the existing protocols in a controlled and reproducible manner. In this paper, we present a comprehensive survey on current network simulators. We introduce their main features, consider their advantages and disadvantages, and discuss the current and future developments. We hope this survey to be a good reference source for those who feel difficult to find the appropriate network simulators for their research or practical requirements.

Keywords: Network Simulation, Network Simulator, OPNET, NS2, NS3, OMNeT++

Table of Contents

1. Introduction

Simulation is a very important modern technology. It can be applied to different science, engineering, or other application fields for different purposes. Computer assisted simulation can model hypothetical and real-life objects or activities on a computer so that it can be studied to see how the system function. Different variables can be used to predict the behavior of the system. Computer simulation can be used to assist the modeling and analysis in many natural systems. Typical application areas include physics, chemistry, biology, and human-involved systems in economics, finance or even social science. Other important applications are in the engineering such as civil engineering, structural engineering, mechanical engineering, and computer engineering. Application of simulation technology into networking area such as network traffic simulation, however, is relatively new.

For network simulation, more specifically, it means that the computer assisted simulation technologies are being applied in the simulation of networking algorithms or systems by using software engineering. The application field is narrower than general simulation and it is natural that more specific requirements will be placed on network simulations. For example, the network simulations may put more emphasis on the performance or validity of a distributed protocol or algorithm rather than the visual or real-time visibility features of the simulations. Moreover, since network technologies is keeping developing very fast and so many different organizations participate in the whole process and they have different technologies or products running on different software on the Internet. That is why the network simulations always require open platforms which should be scalable enough to include different efforts and different packages in the simulations of the whole network. Internet has also a characteristic that it is structured with a uniformed network stack (TCP/IP) that all the different layers technologies can be implemented differently but with a uniformed interface with their neighbored layers. Thus the network simulation tools have to be able to incorporate this feature and allow different future new packages to be included and run transparently without harming existing components or packages. Thus the negative impact of some packages will have no or little impact to the other modules or packages.

Network simulators are used by people from different areas such as academic researchers, industrial developers, and Quality Assurance (QA) to design, simulate, verify, and analyze the performance of different networks protocols. They can also be used to evaluate the effect of the different parameters on the protocols being studied. Generally a network simulator will comprise of a wide range of networking technologies and protocols and help users to build complex networks from basic building blocks like clusters of nodes and links. With their help, one can design different network topologies using various types of nodes such as end-hosts, hubs, network bridges, routers, optical link-layer devices, and mobile units.

The following sections of the paper are organized as follows. In section 2, we introduce some basic concepts about the network simulations and network simulators. After that, we briefly give an overview of current developments in section 3. From section 4 to 7, we discuss four typical network simulators respectively. A summary is given in section 8.

2. Basic concepts in network simulation

In this section, we will introduce some basic concepts in the area of network simulation. We also try to differentiate and clarify those that easy to cause confusion among readers. We generally introduce the basic idea of the network simulation and simulator and then discuss the difference between simulation and emulation.

2.1. Network simulation and simulator

Generally speaking, network simulators try to model the real world networks. The principal idea is that if a system can be modeled, then features of the model can be changed and the corresponding results can be analyzed. As the process of model modification is relatively cheap than the complete real implementation, a wide variety of scenarios can be analyzed at low cost (relative to making changes to a real network). Network simulator always contain the

However, network simulators are not perfect. They can not perfectly model all the details of the networks. However, if well modeled, they will be close enough so as to give the researcher a meaningful insight into the network under test, and how changes will affect its operation.

2.2.Simulation and emulation

In the research area of computer and communications networks, simulation is a useful technique since the behavior of a network can be modeled by calculating the interaction between the different network components (they can be end-host or network entities such as routers, physical links or packets) using mathematical formulas. They can also be modeled by actually or virtually capturing and playing back experimental observations from a real production networks. After we get the observation data from simulation experiments, the behavior of the network and protocols supported can then be observed and analyzed in a series of offline test experiments. All kinds of environmental attributes can also be modified in a controlled manner to assess how the network can behave under different parameters combinations or different configuration conditions. Another characteristic of network simulation that worth noticing is that the simulation program can be used together with different applications and services in order to observe end-to-end or other point-to-point performance in the networks.

Network emulation, however, means that network under planning is simulated in order to assess its performance or to predict the impact of possible changes, or optimizations. The major difference lying between them is that a network emulator means that end-systems such as computers can be attached to the emu lator and will act exactly as they are attached to a real network. The major point is that the network emulator's job is to emulate the network which connects end-hosts, but not the end-hosts themsel ves. Typical network emulation tools include NS2 which is a popular network simulator that can also be used as a limited-functionality emulator. In contrast, a typical network emulator such as WANsim [WANsim] is a simple bridged WAN emulator that utilizes some Linux functionality.

2.3 Type of network simulators

Different types of network simulators can be categorized and explained based on some criteria such as if they are commercial or free, or if they are simple ones or complex ones.

1. Commercial and open source simulators

Some of the network simulators are commercial which means that they would not provide the source code of its software or the affiliated packages to the general users for free. All the user s have to pay to get the license to use their software or pay to order specific packages for their own specific usage requirements. One typical example is the OPNET [OPNET]. Commercial simulator has its advantage and disadvantage. The advantage is that it generally has complete and up-to-date documentations and they can be consistently maintained by some specialized staff in that company. However, the open source network simulator is disadvantageous in this aspect, and generally there are not enough specialized people working on the documentation. This problem can be serious when the different versions come with many new things and it will become difficult to trace or understand the previous codes without appropriate documentations.

On the contrary, the open source network simulator has the advantage that everything is very open and everyone or organization can contribute to it and find bugs in it. The interface is also open for future improvement. It can also be very flexible and reflect the most new recent developments of new technologies in a faster way than commercial network simulators. We can see that some advantages of commercial network simulators, however, are the disadvantage for the open source network simulators. Lack of enough systematic and complete documentations and lack of version control supports can lead to some serious problem s and can limit the applicability and life-time of the open source network simulators. Typical open source network simulators include NS2 [NS2][NS2-wiki], NS3[NS3]. We will introduce and analyze them in great detail in the following sections.

Table 1 Network simulators

	Network simulators name
Commercial	OPNET, QualNet
Open source	NS2, NS3, OMNeT++, SSFNet, J-Sim

Note that limited by the length, not all of them will be introduced in detail in this paper. However, we will focus on some typical ones and introduce others briefly.

2 Simple vs. complex

Currently there are a great variety of network simulators, ranging from the simple ones to the complex ones. Minimally, a network simulator should enable users to represent a network topology, defining the scenarios, specifying the nodes on the network, the links between those nodes and the traffic between the nodes. More complicated systems may allow the user to specify everything about the protocols used to process network traffic. Graphical applications also allow users to easily visualize the workings of their simulated environment. Some of them may be text-based and can provide a less visual or intuitive interface, but may allow more advanced forms of customization. Others may be programming-oriented and can provide a programming framework that allows the users to customize to create an application that simulates the networking environment for testing.

3 Overview of current developments

Currently there are many network simulators that have different features in different aspects. A short list of the current network simulators include OPNET, NS-2, NS-3, OMNeT++ [OMNeT], REAL [REAL], SSFNet [SSFNet], J-Sim [J-Sim], and QualNet [QualNet]. However, in this paper, we do not intend to cover all the available network simulators. We only select some typical ones (the first 4 simulators) and do some analysis and compare some from the others slightly to get a better view of the main features of a certain network simulator. What we select are the 4 typical network simulators that represent the recent development status in this area.

The network simulators we select for discussion in this paper include OPNET, NS2, NS3, and OMNet++. Of them, the OPNET is commercial software and is a little different from others and we will introduce in the first place. NS2 are the most popular one in academia because of its open-source and plenty of components library. A lot of non-benefit organizations contribute a lot in the components library and it has been proved that the development mode of NS2 is very successful. However, because of some natural design limitation of NS2, the NS3 is currently under development and test. Compared with NS2, NS3 put more emphasis on the documentation works and some specialized people are volunteered to manage different components. Moreover, NS3 is not just an updated version of NS2. NS3 redesigns a lot of mechanisms based on the successful and unsuccessful experiences of NS2. OMNet++ is another important network simulator which has very powerful graphical interface and modular core design. OMNet++ is also open sourced and widely acknowledged in academia.

---

4. OPNET

OPN ET [OPNET] is the registered commercial trademark and the name of product presented by OPNET Technologies incorporation. It is one of the most famous and popular commercial network simulators by the end of 2008. Because of it has been used for a long time in the industry, it become mature and has occupied a big market share.

4.1 Overview

OPNET ' s software environment is called â€œModelerâ€, which is specialized for network research and development. It can be flexibly used to study communication networks, devices, protocols, and applications. Because of the fact of being a commercial software provider, OPNET offers relatively much powerful visual or 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 is used to create the mapping from the graphical design to the implementation of the real systems. An example of the graphical GUI of OPNET can be seen in figure 1. We can see all the topology 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 easy operation through the GUI.

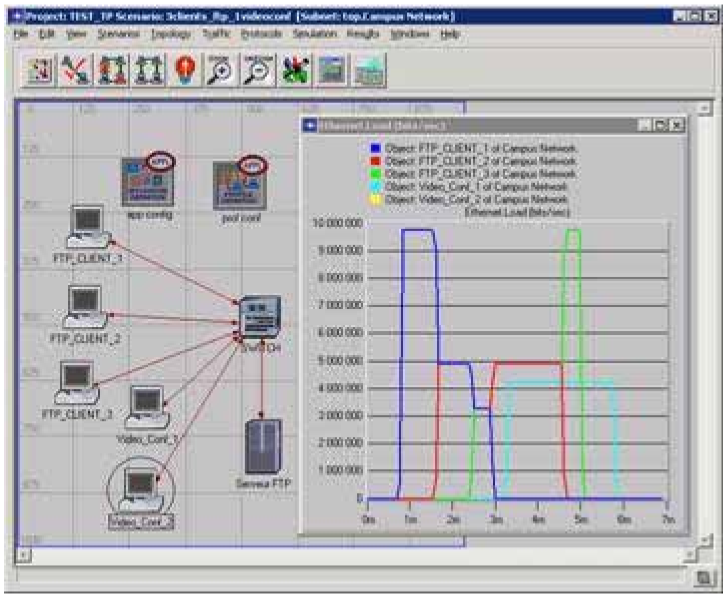


Figure 1. OPNET GUI [OPNET]

OPNET is based on a mechanism called discrete event system which means that the system behavior can simulate by modeling the events in the system in the order of the scenarios the user has set up. Hierarchical structure is used to organize the networks. As other network simulators, OPNET also provides programming tools for users to define the packet format of the protocol. The programming tools are also required to accomplish the tasks of defining the state transition machine, defining network model and the process module.

As of all, OPNET is a popular simulator used in industry for network research and development. The GUI interface and the programming tools are also useful to help the user to build the system they want.

4.2 Main features

OPNET inherently has three main functions: modeling, simulating, and analysis. For modeling, it provides intuitive graphical environment to create all kinds of models of protocols. For simulating, it uses 3 different advanced simulations technologies and can be used to address a wide range of studies. For analysis, the simulation results and data can be analyzed and displayed very easily. User friendly graphs, charts, statistics, and even animation can be generated b y OPNET for users ' convenience.

According to the OPNET whitepaper, OPNET ' s detailed features include:

1. Fast discrete event simulation engine

2. Lot of component library with source code

3. Object-oriented modeling

4. Hierarchical modeling environment

5. Scalable wireless simulations support

6. 32-bit and 64-bit graphical user interface

7. Customizable wireless modeling

8. Discrete Event, Hybrid, and Analytical simulation

9. 32-bit and 64-bit parallel simulation kernel

10. Grid computing support

11. Integrated, GUI-based debugging and analysis

12. Open interface for integrating external component libraries

4.3 Recent development and its future

Recently, about at August 7, 2008, OPNET Technologies announced the addition of two major application performance management capabilities. These capabilities include end-to-end visibility into application performance for organizations using WAN optimization solutions and the ability to capture and analyze NetFlow data.

OPNET recently upgrades its ACE Analyst software includes functionality and it is announced to â€œallow end-user organizations using Riverbed, Cisco, or Juniper WAN optimization appliances to maintain end-to-end visi bility into application performance while deploying WAN acceleration solutionsâ€

[OPNET]. OPNET also provides a module to collect and analyze NetFlow data.

Because of the consistent endeavor and operation of OPNET Inc., OPNET is becoming mature and its product maintain a high acknowledge in the industry. Moreover, OPNET always keeps an eye on the most recent users's requirements and keeps improving their product which make it very competitive compared with other commercial network simulators in the near expectable future.

5. Network Simulator 2 (NS2)

NS2 is one of the most popular open source network simulators. The original NS is a discrete event simulator targeted at networking research. In this section, we will give a brief introduction to the NS2 system.

5.1 Overview

NS2 is the second version of NS (Network Simulator). NS is originally based on REAL network simulator [REAL]. The first version of NS was developed in 1989 and evolved a lot over the past few years. The current NS project is supported through DARPA. The current second version NS2 is widely used in academic research and it has a lot of packages contributed by different non-benefit groups. For NS2 documentation on recent changes, refer to the NS 2 official webpage [NS2].

5.2 Main features

First and foremost, NS2 is an object-oriented, discrete event driven network simulator which was originally developed at University of California-Berkely. The programming it uses is C++ and OTcl (Tcl script language with Object-oriented extensions developed at MIT). The usage of these two programming language has its reason. The biggest reason is due to the internal characteristics of these two languages. C++ is efficient to implement a design but it is not very easy to be visual and graphically shown. I t's not easy to modify and assembly different components and to change different parameters without a very visual and easy-to-use descriptive language. Moreover, f or efficiency reason, NS2 separates control path implementations from the data path implementation. The event scheduler and the basic network component objects in the data path are written and compiled using C++ to reduce packet and event processing time. OTcl happens to have the feature that C++ lacks. So the combination of these two languages proves to be very effective. C++ is used to implement the detailed protocol and OTcl is used for users to control the simulation scenario and schedule the events. A simplified user's view of NS2 is shown in figure 2. The OTcl script is used to initiate the event scheduler, set up the network topology, and tell traffic source when to start and stop sending packets through event scheduler. The scenes can be changed easily by programming in the OTcl script. When a user wants to make a new network object, he can either write the new object or assemble a compound object from the existing object library, and plumb the data path through the object. This plumbing makes NS2 very powerful.

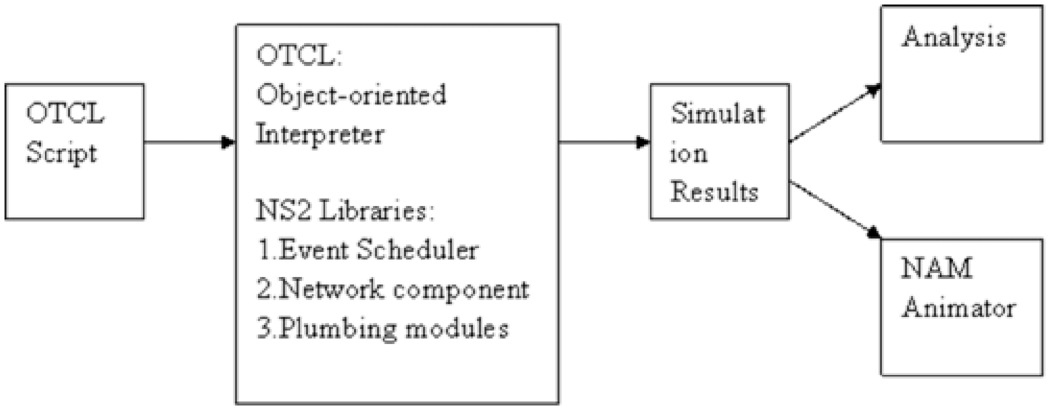


Figure 2. Simplified User's View o f NS2

Another feature of NS2 is the event scheduler. In NS2, the event scheduler keeps track of simulation time and release all the events in the event queue by invoking appropriate network components. All the network components use the event scheduler by issuing an event for the packet and waiting for the event to be released before doing further action on the packet.

5.3 Recent developments and its future

The most recent version of NS2 is NS 2.33 version which was released on Mar 31, 2008. Compared with the previous version, this newest version [NS2] has integrated the most recent extension on new 802.11 models which include the Ilango Purushothaman's infrastructure mode extensions, the 802.11Ext models from a Mercedes-Benz R&D, NA and University of Karlsruhe team, and the dynamic libraries patch and multirate 802.11 library from Nicola Baldo and Federico Maguolo of the SIGNET group, University of Padova. NS is now developed in collaboration between some different researchers and institutions, including SAMAN (supported by DARPA), CONSER (through the NSF), and ICIR (formerly ACIRI). Contributions have also come from Sun Microsystems and the UCB and Carnegie Mellon Monarch projects. Generation 3 of NS (NS3) has begun development as of July 1 , 2006 and is projected to take four years. It is deemed as the future of NS2, and we will discuss the new generation NS 3 in detail in the following section.

6. Network Simulator 3 (NS3)

Similar to NS2, NS3 is also an open sourced discrete-event network simulator which targets primarily for research and educational use. NS3 is licensed under the GNU GPLv2 license , and is available for research and development.

6.1 Overview

NS3 is designed to replace the current popular NS2 . However, NS3 is not an updated version of NS2 since that NS3 is a new simulator and it is not backward-compatible with NS2.

6.2 Main features

The bas ic idea of NS3 comes from several different network simulators including NS2, YANS [YANS], and GTNetS [GTNetS]. The major difference lying between NS3 and NS2 includes:

(1) **Different softw are core** : 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) **Attention to realism** : protocol entities are designed to be closer to real computers.

(3) **Software integration** : support the incorporation of more open-source networking software and reduce the need to rewrite models for simulation;

(4) **Support for virtualization** : lightweight virtual machines are used. Figure 3 gives an example virtualization testbed of NS3.

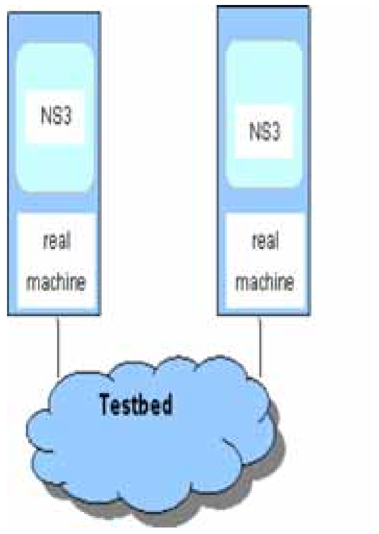


Figure 3 testbeds interconnect NS3 stacks

(5) **Tracing architecture** : NS3 is developing a tracing and statistics gathering framework trying to enable customization of the output without rebuilding the simulation core.

Through the comparison between NS2 and NS3, we can summarize the NS 3' s features as follows:

1. Modular, documented core

2. C++ programs and Python scripting

3. Alignment with real systems

4. Software integration

5. Virtualization and testbed integration

6. Attribute system

7. Updated models

6.3 Recent developments and its future

However, NS3 is still in the process and some major challenges still remain for NS3 to solve. The biggest one is that NS3 needs participation from the research community. Firstly, the simulation credibility needs to be improved. We know that one of the limitations of simulations, in general, is that it often suffers from lack of credibility. Generally there are four points that are important for NS3 to solve this problem. They are:

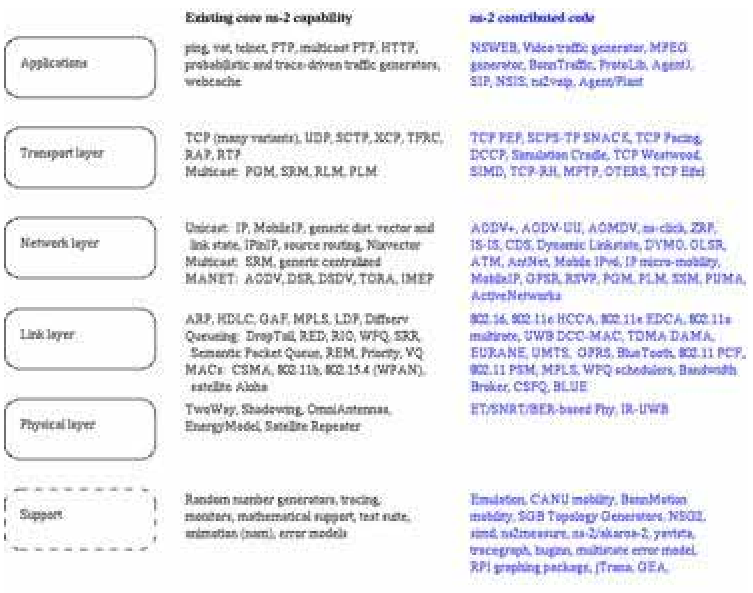
a) Hosting NS3 code and scripts for published work

b) Tutorials on how to do things right

c) Flexible means to configure and record values

d) Support for ported code should make model validation easier and more credible

Secondly, NS3 is intended to replicate the successful mode of NS 2 in which a lot of different organizations contributed to the models and components based on the framework of NS2. The following figure illustrates this status:



	Existing core ns-2 capability	ns-2 contributed code
Applications	ping, vat, telnet, FTP, multicast FTP, HTTP, probabilistic and trace-driven traffic generators, webcache	NSWEB, Video traffic generator, MPEG generator, BonnTraffic, ProtoLib, AgentJ, SIP, NSIS, ns2voip, Agent/Plant
Transport layer	TCP (many variants), UDP, SCTP, XCP, TFRC, RAP, RTP Multicast: PGM, SRM, RLM, PLM	TCP PEP, SCPS-TP SNACK, TCP Pacing, DCCP, Simulation Cradle, TCP Westwood, SIMD, TCP-RH, MFTP, OTERS, TCP Eifel
Network layer	Unicast: IP, MobileIP, generic dist. vector and link state, IPinIP, source routing, Nixvector Multicast: SRM, generic centralized MANET: AODV, DSR, DSDV, TORA, IMEP	AODV+, AODV-UU, AOMDV, ns-click, ZRP, IS-IS, CDS, Dynamic Linkstate, DYMO, OLSR, ATM, AntNet, Mobile IPv6, IP micro-mobility, MobileIP, GPSR, RSVP, PGM, PLM, SSM, PUMA, ActiveNetworks
Link layer	ARP, HDLC, GAF, MPLS, LDP, Diffserv Queueing: DropTail, RED, RIO, WFQ, SRR, Semantic Packet Queue, REM, Priority, VQ MACs: CSMA, 802.11b, 802.15.4 (WPAN), satellite Aloha	802.16, 802.11e HCCA, 802.11e EDCA, 802.11a multirate, UWB DCC-MAC, TDMA DAMA, EURANE, UMTS, GPRS, BlueTooth, 802.11 PCF, 802.11 PSM, MPLS, WFQ schedulers, Bandwidth Broker, CSFQ, BLUE
Physical layer	TwoWay, Shadowing, OmniAntennas, EnergyModel, Satellite Repeater	ET/SNRT/BER-based Phy, IR-UWB
Support	Random number generators, tracing, monitors, mathematical support, test suite, animation (nam), error models	Emulation, CANU mobility, BonnMotion mobility, SGB Topology Generators, NSG2, simd, ns2measure, ns-2/akaroa-2, yavista, tracegraph, huginn, multistate error model, RPI graphing package, jTrana, GEA

Figure 4. NS2 contributions model

Thirdly, NS3 need a lot of specialized maintainers in order to let the NS3 have the advantages as the commercial OPNET network simulators which is documented well. Specialized maintainers can play a key part in the system. In NS3, the active maintainers are required to respond to the user questions and bug reports, and help to test and validating the system.

All in all, NS-3 is an active open-source project and it is still under development. It has several simulator features designed to aid current Internet research. It is also a community-based development and maintenance model, which needs more people and organizations to participate to contribute before it become good enough for the Internet research community.

---

7. OMNeT++

Similar with NS2 and NS3, OMNeT++ is also a public-source, component-based network simulator with GUI support. Its primary application area is communication networks. OMNeT++ has generic and flexible architecture which makes it successful also in other areas like the IT systems, queuing networks, hardware architectures, or even business processes as well.

7.1 Overview

Like NS2 and NS3, OMNeT++ is also a discrete event simulator. It is a component-based architecture.

Components are also called modules and are programmed in C++. The components are then assembled into larger components and models by using a high-level language. Its function is similar to that of OTcl in NS2 and Python in NS3. OMNeT++ also provides GUI support, and due to its modular architecture, the simulation kernel can be embedded into all kinds of different user s' applications. Figure 5 is an OMNeT++ GUI screenshot.

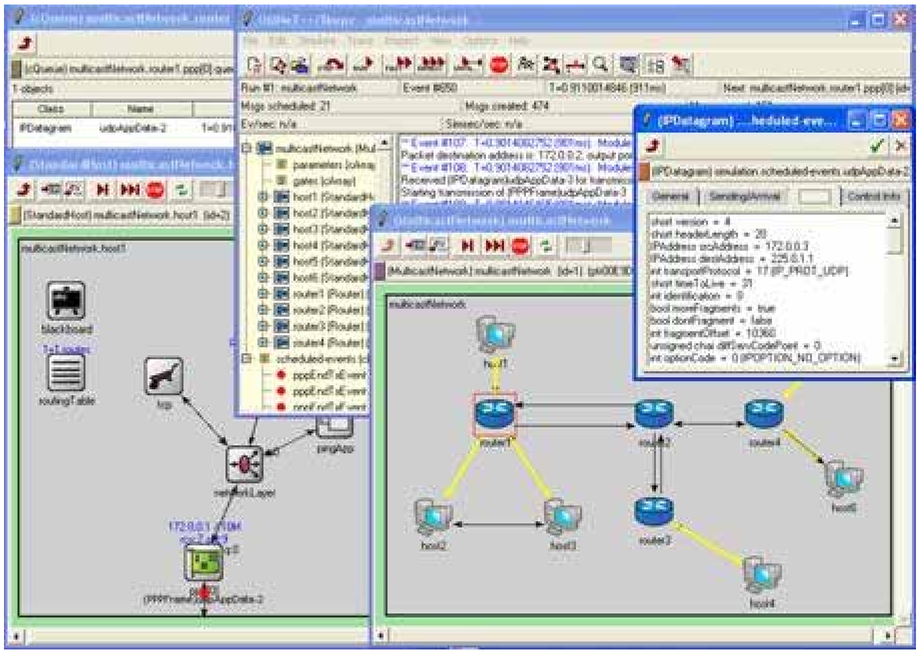


Figure 5. OMNeT++ GUI

7.2 Main features

Since OMNeT++ is designed to provide a component-based architecture, the models or modules of OMNeT++ are assembled from reusable components. Modules are reusable and can be combined in various ways which is one of the main features of OMNeT++. The OMNeT++ components [OMNET] include:

1. Simulation kernel library

2. Compiler for the NED topology description language (nedc)

3. Graphical network editor for NED files (GNED)

4. GUI for simulation execution, links into simulation executable (Tkenv)

5. Command-line user interface for simulation execution (Cmdenv)

6. Graphical output vector plotting tool (Plove)

7. Graphical output scalars visualization tool (Scalars)

8. Model documentation tool (opp_neddoc)

9. Utilities (random number seed generation tool, makefile creation tool, etc.)

10. Documentation, sample simulations, etc.

As the key feature of OMNeT++, the simulation kernel C++ class library consists of the simulation kernel and utility classes which will be used to create simulation components. The library also includes the infrastructure to assemble simulations from different components. Besides these, there are also runtime user interfaces or environments for simulations, and tools to facilitate and manage simulations. OMNeT++ can run on Linux, other Unix-like systems and on Windows (XP, Win2K).

OMNeT++ represents a framework approach. It provides an infrastructure for writing different simulations. Specific application areas' requirements are met by different simulation models and frameworks, most of which are open sourced. More important, these models are developed completely independently of OMNeT++, and follow their own release cycles. This is another important feature of OMNeT++.

7.3 Recent developments and its future

Currently, OMNeT++ is popular in academia for its extensibility since it is also open sourced and there are plentiful online documentations. There is also a mailing list for the general discussion.

OMNeT++ is being used in the academia as well as in industry. Several open source simulation models have been published in the field of network simulations such as IP, IPv6, MPLS, mobility and ad-hoc simulations.

For the future of OMNeT++, we need to note that OMNeT++ is not a network simulator itself. Actually it is currently popular as a network simulation platform in the academia as well as in industry, and build up a large user community. So we have the reason to believe that using OMNeT++ as a basic platform but not an overall single solution. OMNeT++ can have greater development if it could persuade more organizations to participate in and to contribute.

---

8. Summary

This paper is to offer a general overview on the current development status of network simulators for the people who are not very familiar with this topic, or for someone who want to get some general information related to it. So in this paper, we first gave a brief introduction on some key concepts of network simulation and network simulators. After that, we introduced four typical network simulators: OPNET, NS2, NS3, and OMNeT++. Their main features, current status and future development are also analyzed and discussed. As an assumptive expectation, the open sourced and well designed embedded Object-oriented programming feature will be appealing to the academia and industry in the long run. It seems that the easier for the users to learn and to use, the more organizations participate in, the more specialized in the documentation, the more development and popularity that simulator will achieve.

---

9. References

- [WANsim] WAN simulators and emulators, â€œ http://www.wan-sim.net/
- [OPNET] OPNET Modeler, http://www.opnet.com/
- [NS2] NS2 official website, http://www.isi.edu/nsnam/ns/
- [NS2-wiki] NS2 resource webpage, http://nsnam.isi.edu/nsnam/index.php/Main_Page
- [NS3] NS3 official website, http://www.nsnam.org/documents.html
- [OMNeT] OMNeT++ official website, http://www.omnetpp.org/
- [REAL] REAL 5.0 simulator overview, http://www.cs.cornell.edu/skeshav/real/overview.html
- [SSFNet] Scalable Simulation Framework (SSF), SSFNet homepage, http://www.ssfnet.org/homePage.html

- [J-Sim] J-Sim homepage, http://www.j-sim.org/
- [QualNet] QualNet official site, http://www.scalable-networks.com/products/
- [YANS], Yet Another Network Simulator , http://yans.inria.fr/code/yans/?summary
- [GTNetS] The Ge orgia Tech Network Simulator (GTNetS) , http://www.ece.gatech.edu/research/labs/MANIACS/GTNetS/

10. List of Acronyms

- QA - Quality Assurance
- DARPA - Defense Advanced Research Projects Agency
- NS - Network Simulator
- GUI - Graphical User Interface
- OTcl - Tcl with Object-oriented extensions).
- WAN - Wide Area Network

Last modified on November 24, 2008
This and other papers on latest advances in performance analysis are available on line at
http://www.cse.wustl.edu/~jain/cse567-08/index.html
SHARE Back to Raj Jain's Home Page