Received: 15 April 2015 | Revised: 23 May 2015 | Accepted: 1 Jun 2015
DOI: 10.37917/ijeee.11.1.10 | Vol. 11| Issue 1 | June 2015
Open Access
Iraqi Journal for Electrical and Electronic Engineering
Original Article
IJEEE University of Basrh College of Engineering

NTT: Network Topology Tool for Enhancing NS-2

Hayder Naser Khraibet AL-Behadili
Computer Science Department
Shatt Al-Arab University College
Basrah, Iraq
haider_872004 @yahoo.com

Abstract: Network Simulator-2(NS-2) is one of the most popular simulation systems that is widely used in the network community. C++ and the object-oriented Tool Command Language (TCL) are both used to write this simulator. C++ works as a background for this simulator, whereas TCL is responsible for scheduling discrete events and network configuration objects. The TCL language is used to write the code of the simulation scenario. NS-2 does not present enough graphical interfaces that could help a researcher reduce the time spent on writing long TCL scripts. Therefore, network researchers spend a great deal of time focusing on how to write the TCL simulation script, which consequently makes the simulation process more difficult. This study presents a novel tool that enhances simulation by using graphical interfaces. The graphical interface is used to create the network topology and convert it into a TCL script. Thus, the process is visualized easily, efficiently, and quickly. This work describes the Network Topology Tool(NTT),which is intended to help researchers who work under the network simulation environment of NS-2. In such a scenario, researchers can create the network topology through an interactive graphical user interface and also they can retrieve and edit it which considered a very important and unique service from the other previous works. This tool will allow professional users to focus on the development of new algorithms or architectures rather than spend time writing scripts for data processing.

***Index Terms*— Xgraph; NS-2 Simulation; Network performance.**

I. INTRODUCTION

Simulation is a technique that is used to mimic the behavior of a real system, including systems in the telecommunications field or in computer networks [1,2]. The behavior of the network system is modeled using software such as Network Simulator-2(NS-2), OMNeT++, and GNS3[3].NS-2 is a very popular network simulation tool [4].This discrete event simulator is used for research on computer networks. NS-2 can be used in different kinds of networks to simulate a variety of protocols, such as the multicast, transmission control, and routing protocols.

Researchers who use a network simulator have the advantage of handling real data [5]. A simulator makes an operation system function faster and approximates a real system [6]. The Tool Command Language (TCL) is used to create the scenario of the topology.

Most researchers have no sufficient experience with the use of TCL scripts. Therefore, researchers spend much of their time writing these scripts. Unfortunately, NS-2 does not provide a graphical user interface (GUI) to help researchers reduce the time that is spent on writing TCL scripts. Consequently, researchers need to learn the TCL language, which leads to an additional waste of time. The present work proposes a novel tool for enhancing NS-2 simulation through interactive graphical interfaces that help the researcher generate a TCL script. The overall functions of NS-2 are summarized in Fig.(1).

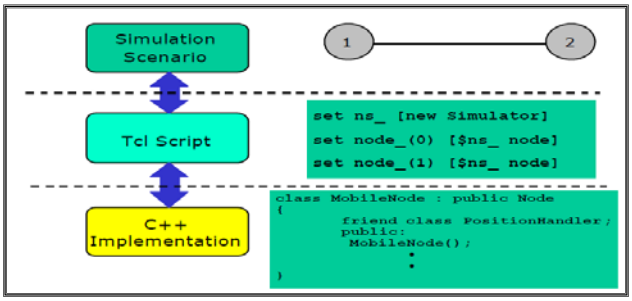

Fig. 1 Overall functions of NS-2.

II. Development and Architecture

The Network Topology Tool (NTT) was developed using Java, with a Windows operating system as its platform. NTT has an open source software license, which enables researchers to modify and update the source code, depending on their needs.

The intended mechanism of the simulation must first be conceptualized before a TCL script can be created. The researcher then writes the TCL script as the source code that can be executed using NS-2. NS-2 then generates a trace file and a Nam file, depending on the problem being addressed by the researcher. An overview of this stage is provided as an example in Fig.(2).

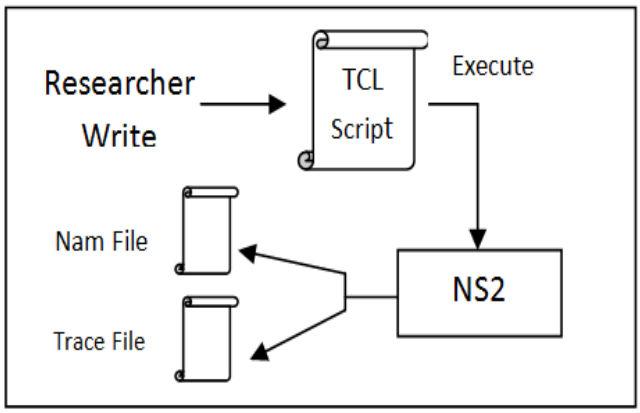

Fig. 2 Overall functions of TCL through NS-2.

By contrast, the NTT architecture works differently. Its architecture can be divided into three layers, from the user interface and to the generating layer. An overview of the NTT layers is shown in Fig.(3).

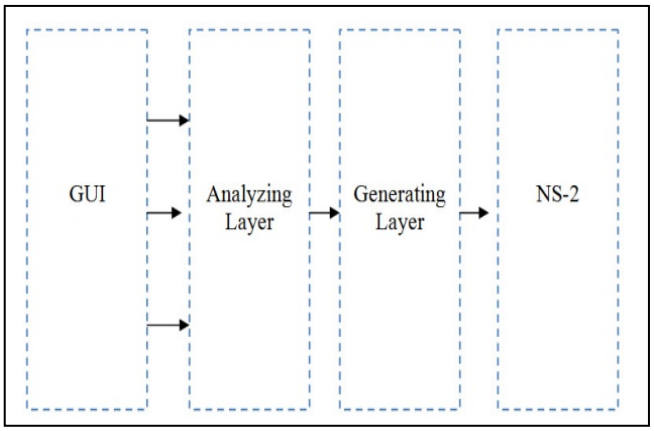

Fig. 3 NTT architecture.

A. GUI

Most researchers use an interface to make their work less difficult. A researcher selects a set of options, including the network type as well as the number of nodes, links, and colors. Finally, the GUI sends the entire configuration into the next layer.

B.ANALYZING LAYER

The analyzing layer extracts the entire configuration for the network then classifies the extracted data. The classified data is saved in a new file called the specification file. The analyzing layer is used to convert the network format of a GUI into details and other information that are saved in a new file. This file contains all the events that the user has encountered in the given network topology. The output of this stage is passed on to the next stage, which is the generating layer.

C.GENERATING LAYER

The generating layer presets the behavior of the analyzing layer. This layer converts the visualization components into clear text, which is a code that presents the above mentioned components. The generating layer extracts the useful parameters that are found in the specification file. This step can be considered the compiler stage because it is fully responsible for the conversion of graphical components into the TCL code. The result code should be clear and understood by NS-2. The flow diagram of the generating layer is presented in Fig.(4).

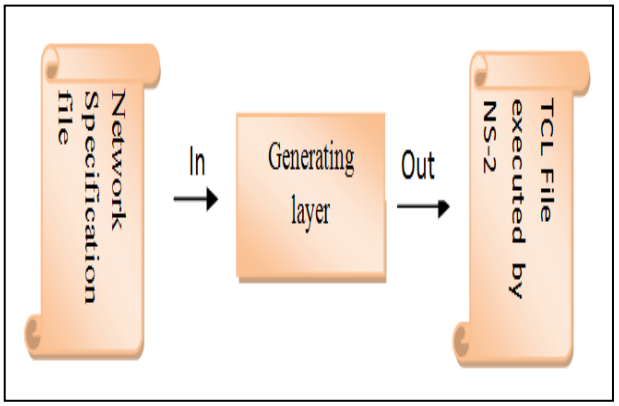

Fig. 4 Generating layer.

NTT tool has one more advantage; it has the ability to retrieve the TCL file. The researchers can retrieve and modify the file by using NTT tool. The source code converts by the NTT tool to visualization components as Icons. The researchers able to modifying on the file and regenerate it again. This step considered a very important and distinguishes this work from the other previous works. Now the researcher can create his topology with less knowledge and less suffering of using code. The flow diagram presented this step of retrieve architecture in Fig. (5).

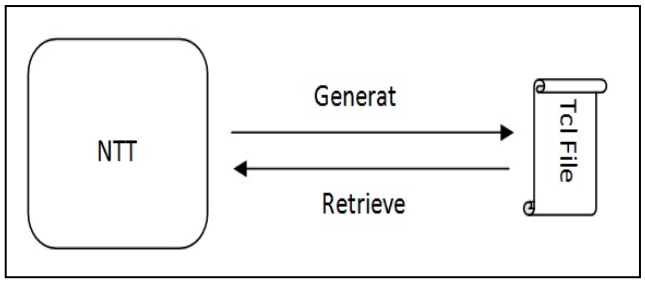

Fig. 5 NTT retrieves architecture.

III. NTT USAGE AND RESULT

In this section, we demonstrate the use of the NTT tool. This example shows how to create a scenario with four connected nodes in the topology. We proposed the creation of four nodes, namely, Node1, Node2, Node3, and Node4. The connections between the four nodes are "duplex-links," with a bandwidth of 5 MB and a delay of 2 ms. The queue is called a "Drop Tail."The user has to enter all the requirements to create the proposed TCL code. The main interfaces of the NTT tool are shown in Fig.(6) and (7).

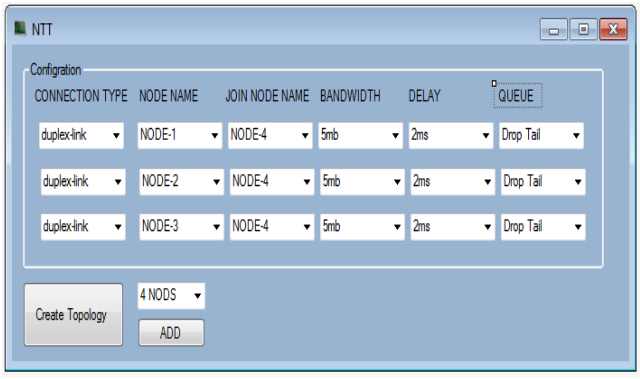

Fig. 6 NTT Configuration Interface.

With the NTT configuration interface, the user has the ability to enter the desired topology configuration, including the type of connection, bandwidth, time delay, and the type of queue. After applying the configuration, the NTT tool will display the final interface that contains the main topology, as proposed in the previous step.

The final step is to produce the TCL code via the generate button, which will convert all the required information into a TCL script. The generated TCL file can be used in NS-2 to run the topology and the Nam file. Fig.(7) shows this topology interface.

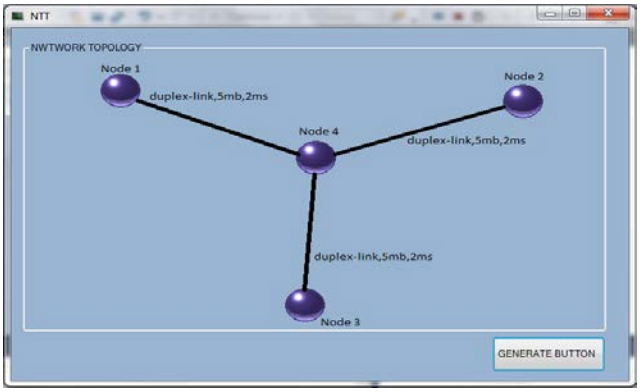

Fig. 7 NTT Topology Interface.

The generated code was run with NS-2 for testing. The result was a100% match with that of the code written via traditional programming. A sample of the testing results is shown in Fig. (8).

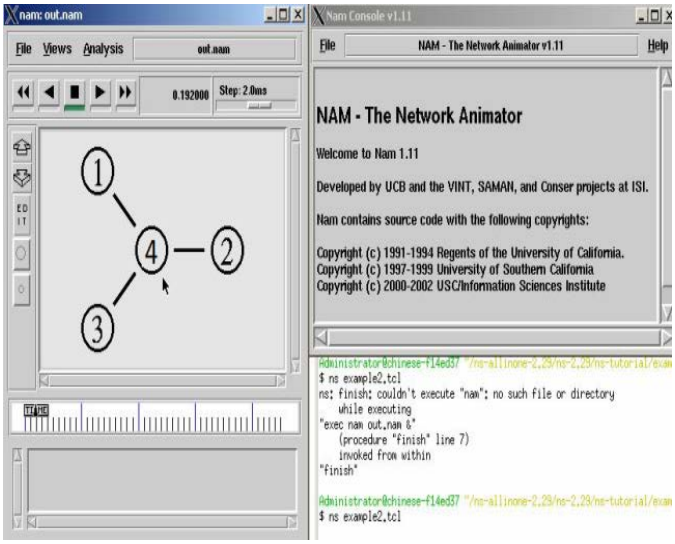

Fig. 8 Nam Topology.

IV. Conclusion and Future Work

The NS-2 simulator has been widely used in the network community. The growing number of researchers in computer networks has led to the use of assistance programs to increase the effectiveness of the system. NTT is a tool that was developed to help researchers in this field. NTT was developed based on three stages. The analyzing layer converts visual specifications into addressed events. The generating layer creates the TCL code based on the events that were generated in the previous step. The final TCL code was run and its result matched that of the code written with traditional programming.

With respect to future work, the NTT was developed in Java for the Windows XP operating system and was tested with the wire protocol. Therefore, the next stage towards making the tool more efficient would be to add wireless protocol functionality. In addition, 3D plotting could be utilized as a GUI to increase the use of NTT.

Acknowledgment

I would like to thanks all that made my research paper possible. Especially for parents, wife, colleagues, and friends through their help and support.

References

[1] L. Breslau, D. Estrin, K. Fall, S. Floyd, J.Heidemann, A. Helmy, P. Huang, S. McCanne, K. Varadhan, and Y. Xu, "Advances in network simulation." vol. 33: IEEE, 2000, pp. 59-67.

[2] J. Pan and R. Jain, " A Survey of Network Simulation Tools: Current Status and Future Developments," School of Computer Science-University of Lugano, Citeseer, Technical Report CSE574S, 2008.

[3] V. Shnayder, M. Hempstead, B.-r. Chen, G. W. Allen, and M. Welsh, "Simulating the power consumption of large-scale sensor network applications," international conference: ACM, 2004, pp. 188-200.

[4] T. Issariyakul and E. Hossain, "Introduction to Network Simulator 2 (NS2)," in Introduction to Network Simulator NS2, 2nd. ed., Ed. Burlingto-USA: Springer, 2009, pp.32.

[5]S. Bajaj, L. Breslau, D. Estrin, K. Fall, S. Floyd, P. Haldar, M. Handley, A. Helmy, J.Heidemann, and P. Huang, "Improving simulation for network research," Technical Report 99-702b, University of Southern California, 1999.

[6] Y. Cao, D. T. Gillespie, and L. R. Petzold, "The slow-scale stochastic simulation algorithm." vol. 122, 2005, p. 014116.