Test bed for Network Protocols Optimization

Afan Ceco, *Member, IEEE*, and Sasa Mrdovic, *Senior Member, IEEE*

***Abstract*—This paper describes a test platform for verifying the functionality of network protocols as well as optimizing their parameters. The test bed is made by using combined OPNET simulator and MATLAB development environment. This test platform connects OPNET network protocols simulator with MATLAB development environment in the way that OPNET runs simulations of network traffic, with predetermined parameter values, while MATLAB executes the script with a mathematical algorithm, which optimizes parameters listed in OPNET simulator. Video traffic is generally transmitted via UDP protocol on the transport layer in TCI/IP model. Recently, alternatives for the transmission of video traffic have emerged with the use of the transport layer TCP protocol. As an exemplary use of the test platform, we performed the analysis of these two approaches, in addition to their comparison using simulations.**

***Keywords* — Test platform, OPNET Modeler, MATLAB, network protocols, optimization, simulation, video streaming, TCP, UDP.**

I. Introduction

IT is very often necessary to test network protocols in the simulation environment since it is impossible to test them in real networks, so as not to disturb the real traffic. Network simulators (e.g. OPNET) have no support for non-standard parameters or their dynamic change. On the other hand, MATLAB is suitable for testing ideas and concepts, because it represents a high level programming language and a development environment. Enabling these two components to connect provides the best of both worlds.

The network simulators such as NS-2 [1], NS-3 [2], OPNET [3], OMNET++ [4] or others, allow for the validity check of networking concepts, as well as for changes in the network protocols. However, they lack the power of specialized software for mathematical calculations, such as MATLAB [5], when it comes to complex mathematical operations (optimization, 3-D models, etc.). In this paper, we present a test platform which can be used to verify the functionality of a new network protocol. The test platform would include optimization techniques, or verification of the existing protocols, in addition to optimization of their parameters. It consists of a connected simulator OPNET Modeler 14.5 and a development environment MATLAB R2012.

Paper received April 14, 2019; revised July 02, 2019; accepted July 13, 2019. Date of publication July 31, 2019. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Grozdan Petrović.

This paper is revised and expanded version of the paper presented at the 26th Telecommunications Forum TELFOR 2018 [10].

Afan Ceco, BH Telecom d.d. Sarajevo, Masarykova 46, 72000 Zenica, B&H (e-mail: afan.ceco@bhtelecom.ba).

Sasa Mrdovic, Faculty of EE, University of Sarajevo, Zmaja od Bosne bb, 71 000 Sarajevo, B&H (e-mail: sasa.mrdovic@etf.unsa.ba).

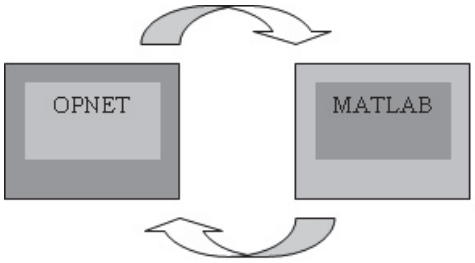

Fig. 1. Link between OPNET and MATLAB.

This test platform connects the OPNET simulator of network protocols with MATLAB development environment in the way that OPNET runs simulations of network traffic, with predetermined parameter values, while MATLAB executes the script with a mathematical algorithm, which optimizes the parameters listed in OPNET simulator.

This paper is an extended version of the work published under the same title [6]. The first section of this paper presents the situation in the subject area. The second section describes OPNET simulator, while the third section introduces MATLAB development environment. The fourth section explains the test platform consisting of paired OPNET and MATLAB, and its implementation. The final section of paper reports the results of simulations, obtained by optimizing the parameters, and executed on the subject test platform.

II. Related Work

Research papers found in this subject area describe the models for linking different simulators as well as for linking simulators and other software tools. Very often, this refers to a so-called co-simulation, which represents a problem division into two or more parts, which are simultaneously simulated on a variety of simulators. The second most common use is to connect two simulators to validate the results from one simulator, by checking them on the other simulator (from another manufacturer). This paper's main topic is co-simulation as well as the use of two different simulators together [7]. The authors described a platform for co-simulation consisting of connected simulators OPNET and SIMULINK. The SIMULINK environment for simulation is based on block diagrams and design of models [8]. Both simulators are executed in parallel, and interact with the synchronization, whereby the main simulator is OPNET. In this paper [9], the authors presented the implementation of connectivity of OMNET++ simulator and MATLAB development environments, with an emphasis on co-simulation. Paper [10] describes an example of connecting OPNET with MATLAB, provided that it is done to validate the results on MATLAB that are obtained on OPNET simulator.

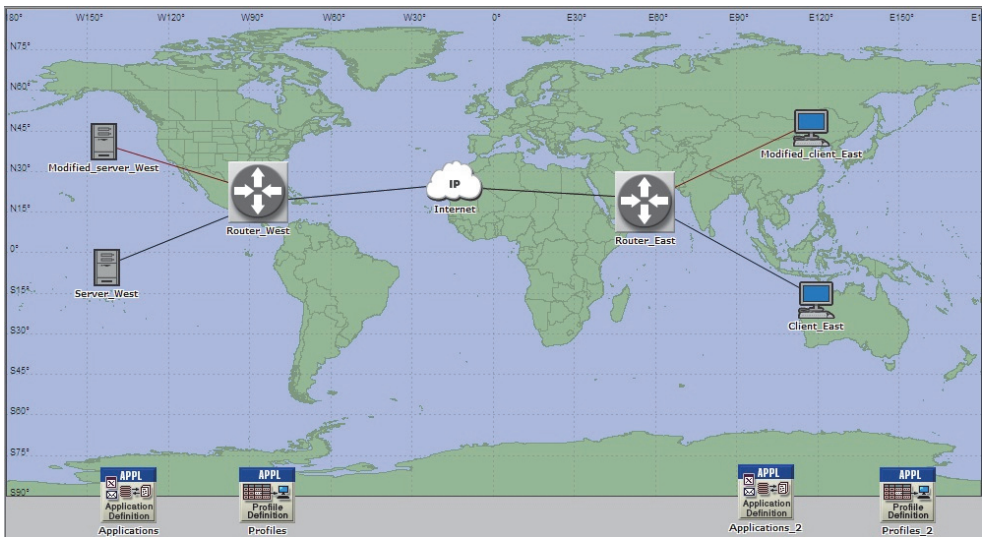

Fig. 2. OPNET simulator – Test network topology.

Unlike the above-described models of simulators linkage, we herein present the connection of OPNET and MATLAB on a common test platform, where they act as a single combined simulator using the advantages of both software environments. This test platform applies an iterative principle to reach the best possible results, and through a number of (automated) starting and running simulations, it leads to the optimization of the simulation parameters.

III. OPNET Simulator

OPNET is a simulator of network protocols and communications, manufactured by the company Riverbed. There are several different versions of this product. The most popular version is called "OPNET IT Guru Academic Edition", most commonly used in the academic community, for it was free of charge for educational use. However, in this version, simulator users were not allowed to modify the source code of the simulator. At present, the current replacement for this variant is called "OPNET Modeler Academic Edition", which was presented in 2014. In this paper, we used a commercial version of OPNET Modeler, mostly because this version of the simulator has the option of changing the source code. This allows for modification of the complete behavior of network protocols being a part of the simulator. Fig. 2 shows OPNET simulator GUI with a built topology network that is used to test the proposed linkage of OPNET and MATLAB. The topology consists of a server, which is connected to the internetwork via a router, then the internetwork itself (Internet), and a client, who is also connected to the internetwork via a router.

Fig. 3 presents the protocols for the client node, i.e. all protocols that are available on the simulator for that node and that can be changed. Fig. 4 shows a state diagram for the TCP protocol, one of the protocols on the client node. It is important to point out the fact that each state in the diagram can be opened, displaying its code in C/C++, which can be changed, if necessary.

In addition, Fig. 5 and Fig. 6 show the parameters of the TCP protocol, as well as the function block with the C/C++ code that can be changed. The parameter values in Fig. 5 can be changed programmatically by changing the code as shown in Fig. 6. Once we modify the simulator code, it is necessary to recompile the simulator.

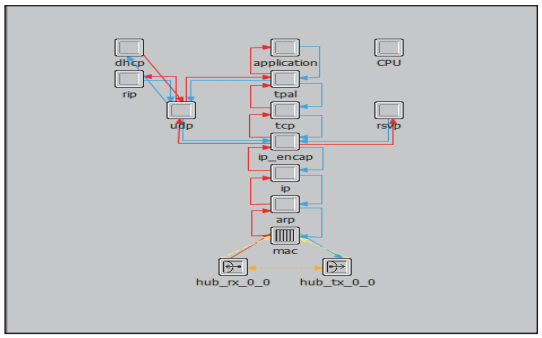

Fig. 3. OPNET simulator – Display of protocols for client node.

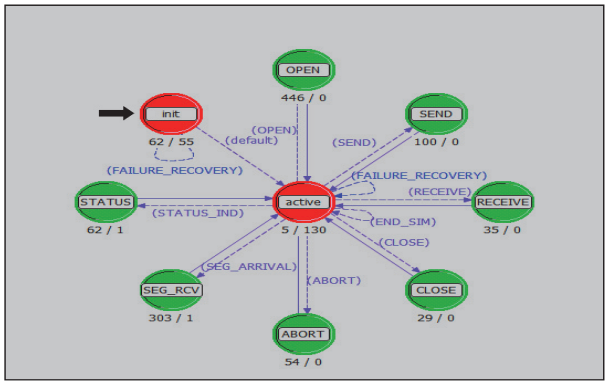

Fig. 4. OPNET simulator – State diagram for TCP protocol.

Attribute Name	Type	Units	Primary Key	Default Value	Prominent	Tags
Version/Flavor	string			Unspecified		
Maximum Segment Size	integer	bytes		Auto-Assigned		
Receive Buffer	integer	bytes		8760		
Receive Buffer Adjustment	integer			None		
Receive Buffer Usage Threshold	double	of RCV BUFF		0.0		
Delayed ACK Mechanism	integer			Segment/Clock Based		
Maximum ACK Delay	double	sec		0.200		
Maximum ACK Segments	integer			2		
Slow-Start Initial Count	integer	MSS		2		
Fast Retransmit	toggle			Disabled		
Duplicate ACK Threshold	integer			3		

Fig. 5. OPNET Simulator – Display of parameters.

```
static void
tcp_mgr_sv_init (void)
    {
    /** Initializes the state variables used in this model. **/
    FIN (tcp_mgr_sv_init ())

    /* Initialize variables used for process registry.  */
    /* Obtain the tcp2 module's objid. */
    own_mod_objid = op_id_self ();

    /* Obtain the node's objid. */
    own_node_objid = op_topo_parent (own_mod_objid);

    /* Obtain the tcp_manager2 process's prohandle. */
    own_prohandle = op_pro_self ();

    /* Obtain the name of the process. It is the "process model" attribute of the nod
    op_ima_obj_attr_get (own_mod_objid, "process model", proc_model_name);

    /* Initialize the diagnostic structure dynamic array pointer */
    diag_ptr = (TcpT_Diag *) op_prg_mem_alloc (CONNECTION_STATISTIC_COUNT * sizeof (T

    /* Init Vars. */
    conn_id_new = 1;

    tcb_list = op_prg_list_create ();
    if (tcb_list == OPC_NIL)
        {
        op_prg_log_entry_write (ll_loghndl, "TCP initialization failed - unable to cr
        op_sim_end ("TCP initialization failed - unable to create TCB list.",
```

Fig. 6. OPNET Simulator – Function block.

In the presented case, we have used MS Visual C++ 9.0 (MS Visual Studio 2008) on MS Windows 7. It is required to set the environment variables to link the compiler and simulator. All the details related to the change of the OPNET Modeler code are given in Book [11]. It should be noted that the operating system, simulator, and the compiler, should be the same generation, having appeared on the market in about the same time, because otherwise some compatibility issues arise.

By changing the code in OPNET simulator, the existing protocol can be modified to work in a different manner, or we can even create an entirely new network protocol. The precondition to do so is, of course, to install C/C++ compiler, and to set the environment variables.

IV. MATLAB Development Environment

MATLAB is a development environment and a high level programming language for numerical and matrix computation. It is manufactured by MathWorks, the company which offers a SIMULINK simulator as well.

MATLAB represents all data in the form of an array. The routines for manipulating the arrays in MATLAB begin with the prefix mx. Data can be imported and exported to and from MATLAB in multiple ways. Some of them are presented below.

MATLAB can invoke functions and procedures written in the C programming language or in Fortran. When invoking, the so-called MEX files are commonly used, which represent the interface between MATLAB and subroutines written in C, C++ or Fortran. When compiled, MEX files are dynamically loaded and allow other codes to be called within MATLAB, as if they were its own functions. With a MEX file, of course, we can read or write data. A detailed explanation can be found in a manual issued by the MATLAB development environment manufacturer [12].

In contrast, the so-called MAT files allow the import and export of data to or from MATLAB environment in a direct way. MAT files are actually files that MATLAB uses to store data on the disk. MAT files provide an easy and convenient mechanism for transporting data between different platforms. Also, they allow the import and export of data into MATLAB from some other MATLAB or stand-alone applications. In order to simplify the use of MAT files from other applications, an access routines library is offered, and they all begin with the prefix mat, which makes reading and writing of MAT files very easy using C or Fortran program [12].

It is also possible to load data into MATLAB from ASCII files. Such files must have data written in ASCII format, with fixed-length rows ending with the transition to a new line, as well as spaces that separate the numbers. Data is retrieved from this file by using the load command, and are recorded using the save command [12].

V. Coupling of OPNET and MATLAB

In this paper, we present a test platform that connects OPNET simulator of network protocols with MATLAB development environment. OPNET runs simulations of network traffic, with the predetermined values of parameters, such as the percentage of a packet loss, and other parameters of the TCP connection. Upon completion of the simulation, the values of the variables affecting the parameters we have been looking to improve are recorded, and the script with the mathematical algorithm, which optimizes the above parameters, is executed in MATLAB, for the purposes of OPNET simulator. Fig. 7 shows a flow diagram of the execution of simulation and optimization, which can be applied to any communication protocol. The initial parameter values (x1, y1) are used in OPNET simulator as input parameters. Once the simulation ends, the result obtained is shown on the diagram as the value (R). This value (R) is imported into MATLAB to carry out the effectiveness check of the initial parameter values. Subsequently, the optimization algorithm in MATLAB is changing the initial parameters (x1, y1) to (x2, y2), which is presented in Fig. 7 by operator λ. These parameters (x2, y2) are recorded now as the initial parameters (x1, y1).

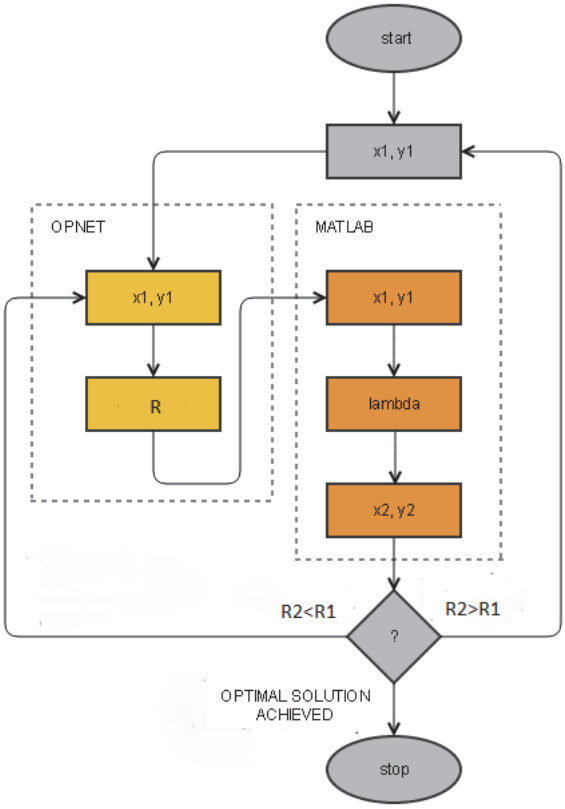

Fig. 7. Flow Diagram – Execution of simulation with the relationship between OPNET and MATLAB.

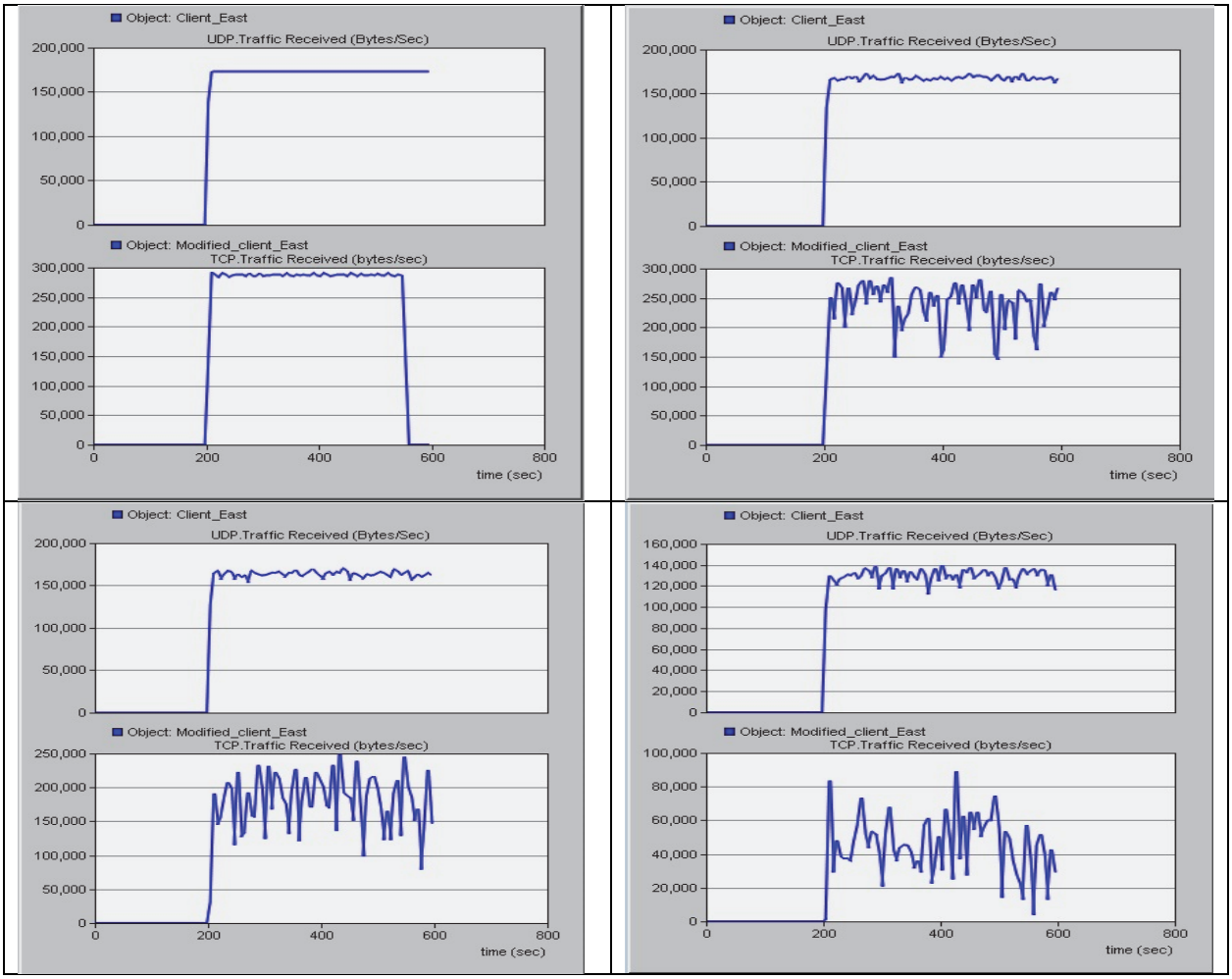

Fig. 8. Results of simulation for faster links (DS3); top left: no loss of packets; top right: for 0.5% packet losses (errors); bottom left: for 1% errors; bottom right: for 5% of errors.

During the next start of simulation in OPNET, the new input parameters are being loaded. The result (R) is returned to MATLAB and compared with the previous result. If the result is more favorable, the optimization algorithm continues to further modify the parameters. If the result is less favorable, the optimization algorithm rejects the previously caused modification. The link between OPNET and MATLAB can be made in several ways, and these methods can be divided into two groups: using files or software methods for direct applications linking. In this case, the connection is achieved using ASCII files and by recording the variables in the file from one software, and then loading the same variables from the given file in another software.

The connection between MATLAB and OPNET can be established without the files, using COM (Component Object Model) software components, or a similar method of communication between processes, or for the connection of various applications (such as OLE, ActiveX, etc., all of which are derived from a COM model). Starting OPNET simulation numerous times, along with the execution of MATLAB scripts each time, is provided by the software for action automation on computer system called "MurGee" [13]. OPNET simulator also has its own options for automating the execution of simulations, except that in this case we should ensure the calls of MATLAB scripts, for each iteration cycle.

VI. Performance Analysis of Video Traffic via TCP Protocol

The transport layer of the TCP/IP model requires a decision about the usage of the transport protocol based on the request from the application layer. In doing so, we either use the TCP, as a reliable protocol, which has acknowledgments and re-transmits lost segments, thus introducing certain slowness; or the UDP, which does not have acknowledgments, nor it re-transmits lost segments, but works faster accordingly.

In this performance analysis simulation of video traffic, we use the modified TCP protocol and compare it with the UDP based video transmission. The modification is based on using heuristics to optimize TCP connection parameters. This is achieved on the basis of the test platform, presented in this paper.

Fig. 2 shows the network topology used to perform the simulations. The topology consists of two servers, two clients and an inter-network between them. The first client initiates video traffic from the first server through the UDP transport protocol. The second client initiates video traffic from the second server via the TCP transport protocol.

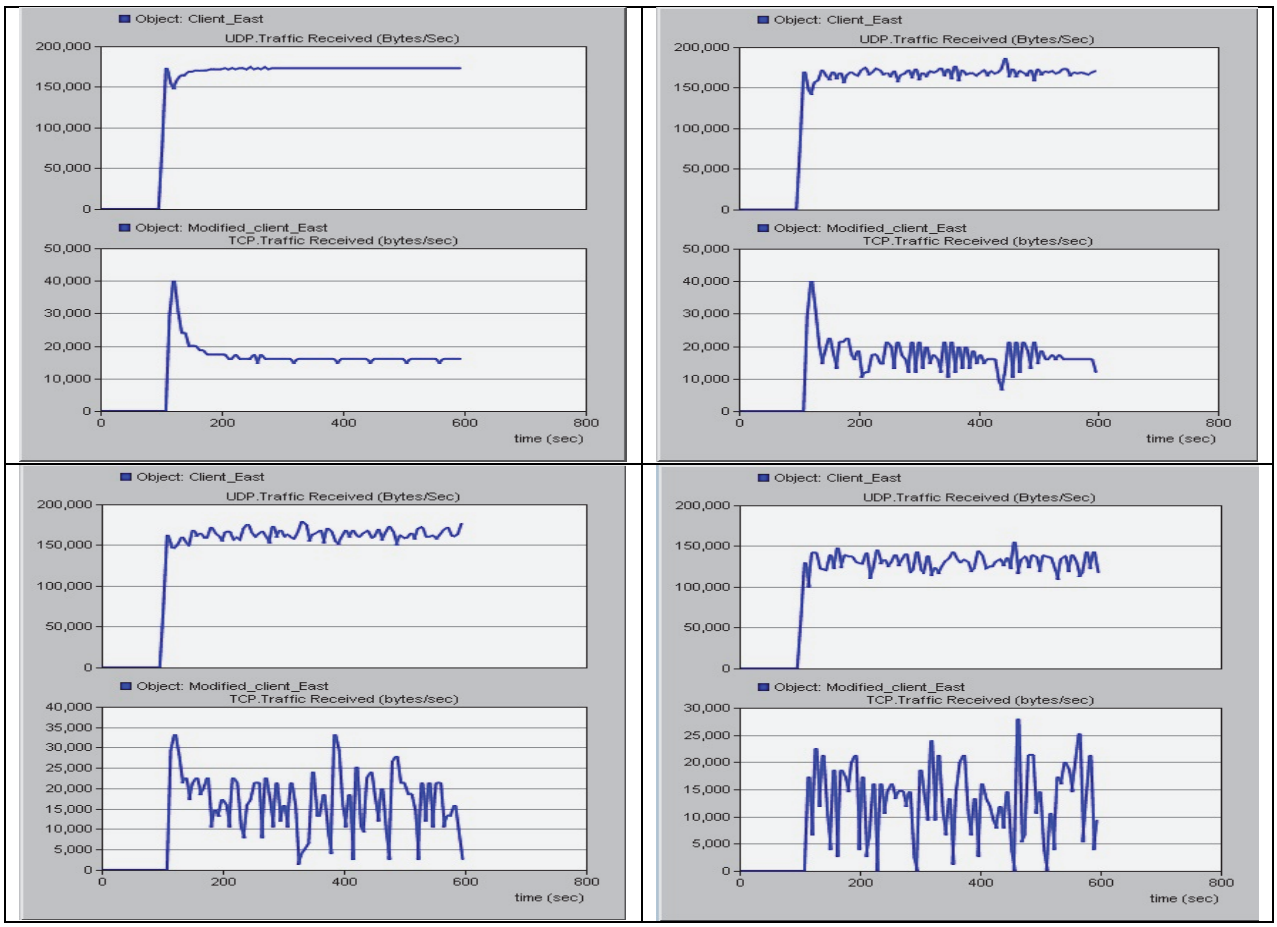

Fig. 9. Results of simulation for slower links (DS1); top left: no loss of packets; top right: for 0.5% packet losses (errors); bottom left: for 1% errors; bottom right: for 5% of errors.

Links between the routers were originally DS3 (45 Mbps) and then they are reduced to DS1 (1.5 Mbps).

Fig. 8 shows that performance scores are better for the TCP than for the UDP when it comes to faster links (DS3 - 45 Mbps). The errors or the losses of packets gradually increased. At first, they were 0%, then 0.5%, 1%, and finally 5%. And with such scaled errors, the TCP achieves a better performance in terms of throughput, except in the last case - for 5% errors.

Fig. 9 demonstrates that performance results are not better for the TCP than for the UDP, in terms of slower links (DS1 - 1.5 Mbps). This is in contrast to previous results obtained for high-speed links. In other words, in case of slower speed links, the UDP achieves better throughput results.

However, if the UDP, lost packets, or datagrams, are not re-transmitted, so as the errors are more enhanced, the final result is less favorable in terms of the video image quality, since some of the parts of the image are missing. When the TCP is used as a transport protocol for video traffic, the re-transmission of lost packets/segments is performed, and the video image is ultimately completed before displaying.

The emphasis of this work is on linking MATLAB and OPNET. The specified algorithm is only one of the possible ways to use this connection. The intention was to show that in OPNET we can choose arbitrary parameters that are possible to optimize in MATLAB, and then return the result to OPNET.

VII. Conclusion

This paper describes an example of OPNET simulator and MATLAB development environment connection in a common test platform, using files as a means of communication between these two software environments. The presented test platform can be used to verify the functionality of the new network protocol that would include optimization techniques, or for the verification of the existing protocols in addition to optimization of their parameters. OPNET runs network traffic simulations, with the predetermined parameter values, while MATLAB executes the script with a mathematical algorithm, which optimizes the parameters in OPNET simulator. This test platform operation has been tested on the example of optimization of parameters for the TCP protocol, which is shown in the form of results, along with the performance obtained by simulation, before and after the application of optimization.

The presented simulations show the results of performance comparisons in terms of video traffic throughput, using TCP and UDP transport protocols. Simulations are made for scaled errors, as well as for faster and slower links. On the basis of the obtained simulation results, it can be concluded that the results in terms of transmission performance are more favorable for the TCP

than for the UDP, when it comes to faster links. However, when it comes to slower links, performance is less favorable for TCP than for UDP. When considering the UDP, lost packets (datagrams) are not re-transmitted, while the TCP is performing their re-transmission. Therefore, as the errors (packet loss) are more enhanced, the more important becomes the role of the TCP protocol in obtaining the final result in terms of video image quality, for there are no missing parts of the video image.

REFERENCES

[1] NS-2 URL: http://nsnam.isi.edu/nsnam/index.php/User_Information
[2] NS-3 URL: http://www.nsnam.org/
[3] Riverbed OPNET Modeler URL: http://www.riverbed.com/products-solutions/products/networkplanning-simulation/NetworkSimulation.html
[4] OMNeT++ Homepage. Available: http://www.omnetpp.org/
[5] http://www.mathworks.com/products/matlab/
[6] A. Ceco and S. Mrdovic, "Test Bed for Network Protocols Optimization," *2018 26th Telecommunications Forum (TELFOR)*, Belgrade, Serbia, 2018, pp. 1-4.
[7] M.S. Hasan, H. Yu, A. Carrington, and T.C. Yang, "Co-simulation of wireless networked control systems over mobile ad hoc network using SIMULINK and OPNET," *IET Communications*, vol. 3, no. 8, pp. 1297–1310, Aug. 2009.
[8] http://www.mathworks.com/products/ simulink/index-b.html
[9] Z. Zhang, Z. Lu, Q. Chen, X. Yan and L. Zheng, "COSMO: CO-Simulation with MATLAB and OMNeT++ for Indoor Wireless Networks," *2010 IEEE Global Telecommunications Conference GLOBECOM 2010*, Miami, FL, 2010, pp. 1-6.
[10] G. E. Pérez and I. Kostanic, "Comparing a Real-Life WSN Platform Small Network and its OPNET Modeler Model using Hypothesis Testing", *Journal of Systemics, Cybernetics and Informatics*, vol. 12, no. 7, pp. 66–73, 2014.
[11] Z. Lu and H. Yang, *Unlocking the Power of OPNET Modeler*, Cambridge University Press, 2012.
[12] MATLAB - The Language of Technical Computing (Application Program Interface Guide v5), The MathWorks Inc, 1998
[13] MurGee URL: http://www.murgee.com