DYNA
ISSN: 0012-7353
Universidad Nacional de Colombia

Campo-Muñoz, Wilmar Yesid; Astaiza-Hoyos, Evelio; Muñoz-Sanabria, Luis Freddy
Traffic modelling of the video-on-demand service through NS-3
DYNA, vol. 84, no. 202, 2017, July-September, pp. 55-64
Universidad Nacional de Colombia

DOI: https://doi.org/10.15446/dyna.v84n202.61650

Available in: https://www.redalyc.org/articulo.oa?id=49655539007

UNIVERSIDAD NACIONAL DE COLOMBIA
SEDE MEDELLÍN
FACULTAD DE MINAS

DYNA
http://dyna.medellin.unal.edu.co/

Traffic modelling of the video-on-demand service through NS-3

Wilmar Yesid Campo-Muñoz [a], Evelio Astaiza-Hoyos [a] & Luis Freddy Muñoz-Sanabria [b]

[a] *Faculty of Engineering, Universidad del Quindío, Armenia, Colombia. wycampo@uniquindio.edu.co, eastaiza@uniquindio.edu.co*
[b] *Faculty of Engineering, Fundación Universitaría de Popayán, Popayán, Colombia. lfreddyms@fup.edu.co*

Received: December 19th, de 2016. Received in revised form: June 6th, 2017. Accepted: June 29th, 2017

Abstract
The principal characteristic of the video on demand service through video streaming technology is the consumption of large bandwidths, which is why network planners must consider the service when dimensioning said networks. To achieve this goal, one of the tools is traffic engineering and its associated models. Thus, this article presents a traffic model based on simulation through discrete events and developed through the open code software and of research on networks, denominated ns-3. This work describes the abstractions of network elements and the construction process of different evaluation scenarios, independent of the characteristics of the service sought to be analyzed, and the collection and graphic visualization de statistics. Finally, the work presents the analyses of the performance parameters that permit obtaining the minimum network characteristics to support the service.

Key words: Network performance; NS-3, simulation; traffic model; video service.

Modelado de tráfico del servicio de video bajo demanda mediante NS-3

Resumen
La principal característica del servicio de video bajo demanda a través de la tecnología de video streaming es el consumo de grandes anchos de banda, por lo que los planificadores de redes deben considerar este servicio en el momento de dimensionar dichas redes. Para alcanzar esta meta, una de las herramientas es la ingeniería de tráfico y sus modelos asociados. Así, este artículo presenta un modelo de tráfico basado en la simulación por eventos discretos y desarrollado a través del software de código abierto y de investigación en redes, denominado ns-3. Se describen las abstracciones de los elementos de red, y el proceso de construcción de diferentes escenarios de evaluación, independiente de las características del servicio que se desee analizar, y la recolección y visualización gráfica de estadísticas. Finalmente se presentan los análisis de los parámetros de desempeño que permiten obtener las características mínimas de red para soportar el servicio.

Palabras clave: Desempeño de red, NS-3; simulación; modelo de tráfico; servicio de video.

1. Introduction

Video streaming is an audiovisual service that permits consumption of multimedia content. This service can be provided through diffusion, point to multipoint, or point-to-point. Its principal advantage lies in not having to download contents onto a computer to be watched later; also, contents can be seen in real-time, as with a live recording of a television program [1].

IP video traffic through video streaming will represent 80% of all IP traffic in 2019, above 75% of 2015 [2]. Residential and business consumers and mobile consumers continue demanding advanced video services through all types of networks and devices, making quality, comfort, content, experience, and price the key success factors [2]. Hence, network planners must be able to dimension and update said networks to satisfy end users without causing them trauma, besides protecting company investments in infrastructure.

Among the tools available to network planners and engineers are traffic models as part of engineering, which constitutes an important branch for planning and implementation of communication networks, given that they permit previously analyzing the behavior of the traffic generated by these services and applications, and determines the network resources necessary for their implementation. The teletraffic theory is defined as the application of the probability theory to the solution of problems related to

How to cite: Campo-Muñoz, W.Y., Astaiza-Hoyos, E. and Muñoz-Sanabria, L.F., Traffic modelling of the video-on-demand service through NS-3 DYNA, 84(202), pp. 55-64, September, 2017.

planning, performance evaluation, operation, and maintenance of the telecommunication systems [3].

The ns-3 tool was used to conduct this research, which is framed within the traffic models based on discrete event simulation according to the classification presented in [4]. These models describe phenomena in a given moment, where each event has a specific incidence instant that can be represented on the discrete temporal scale of the simulation occupying a single position and organizing chronologically, which facilitates processing. Each event is an action and its result is the modification of state variables, which makes them suitable to simulate with programming languages aimed at objects [5, 6].

The ns-3 tool works through discrete events, programmed totally in C++, mainly focused on research and educational purposes. It is an open software project under the GNU GPLv2 license, which is why users can create or modify existing models. ns-3 is not only a tool for simulation and emulation, it is also a software development project for research on telecommunication networks, capable of supporting the latest technologies and in which world-renowned universities participate; besides companies and research groups aimed at networks [7]. For these reasons, NS-3 is the selected tool for this research. Furthermore, based on the IEEE Xplorer Digital Library searches, it is the most used free tool for telecommunication research.

The contribution of this article is the development of a traffic model and the evaluation of the performance of the video on demand service (VoD) on an Ethernet, through simulation techniques via discrete events, through the research tool in telecommunication networks of the ns-3 scientific community. Additionally, this study presents the construction process of the evaluation scenarios independent of the characteristics of the service to be analyzed, seeking to offer the community a reference in the construction of models based on discrete events through ns-3.

The development methodology of this research is based on an adaptation of that described in [8], founded on states and transitions, where the first state corresponds to the real experimentation scenario and the transition corresponds to the mathematical functions that describe the behavior of the videos. The second state corresponds to the simulation environment and the respective transition refers to the topologies. The third state corresponds to the modelling of the service through discrete event simulation (DES), whose transition corresponds to the model. The final state corresponds to the results yielded by executing the model.

This article is organized in the following manner: section 2 presents the real experimentation scenario and the tools used for its construction. Section 3 describes the construction of the simulation environment in ns-3, while section 4 studies the construction of the model through discrete events in ns-3. Section 5 describes the analysis and results of the research. Finally, section 5 presents the conclusions of the present work.

2. Real experimentation scenario

To develop the model, three videos were used with distinct characteristics regarding duration and mobility, which permit a comparative analysis among them. Table 1 presents their characteristics.

Table 1.
Characteristics of the videos

Characteristics of the videos			
	Video 1	**Video 2**	**Video 3**
Duration	94 s	109 s	28 s
Size	62.4 MB	58.9 MB	9.7 MB
Bit rate	5.2 Mbps	4.3 Mbps	2.8 Mbps
Complexity	Movement appreciable throughout its whole duration	Movement smaller than video 1	Low mobility

Codec: mpeg2. Container format: TS (Transport Stream). Resolution: 1920x1080. FPS (Frames Per Second): 29.97. Aspect resolution: 16:9

Source: The authors.

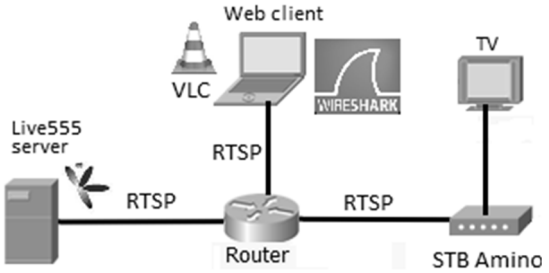


Figure. 1. Scenario of VoD experimentation service.
Source: The authors.

Fig. 1 presents the experimentation scenario used to capture the real traffic of the VoD service; the streaming server used was live555, which is in charge of disseminating previously encoded multimedia content in the network. As streaming flow clients we used a set top box (STB) of internet protocol television (IPTV) by Amino and a portable computer in charge of capturing and consuming the multimedia flow transmitted, using wireshark and VLC tools, respectively [9]. The VoD service is available for consumption via internet by using a web platform or also via IPTV directly to a television set by using an STB, through using the real-time transmission protocol (RTSP).

To consume this service through an IPTV network, the STB in particular needs the files consumed to be in MPEG-TS format (MPEG-Transport Stream) whose extension is .ts. To obtain this type of file, ffmpeg software was used. By using this software the codecs of the files were changed for experimentation so that they coincided with those used by the IPTV *STB*, these are: mpeg-2 for video and aac for audio; additionally, other parameters are configured, like resolution, aspect ratio, and bit rate. Fig. 2 shows the encoding and compression process carried out on a video file with .wmv extension by using ffmpeg. The change of codecs can be seen in video from wmv2 to mpeg2video and in audio from wmav2 to libfaac; in addition, the file is packaged in transport stream (TS). This process is conducted in the Ubuntu operating system. To achieve capturing the traffic traces between the server VoD live555 and the client, the wireshark protocol analyzer was used. This tool is capable of capturing all the traffic streaming through the device to which it is connected. The MPEG-2 encoding standard, which has been used to encode the videos, creates a flow through four types of data, these are: intra frames, predictable posterior frames, bi-directional predictable frames, and audio, referred to as I, P, B, and audio frames, respectively.

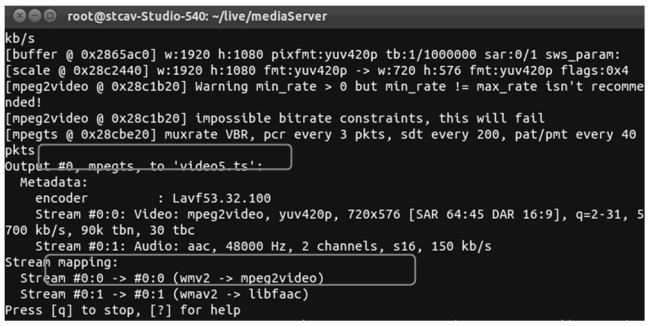


Figure 2. Change of video encoding with ffmpeg.
Source: The authors.

The traffic captured through wireshark contains all types of information on control, filler, synchronization, etc., besides the distinct types of frames that comprise the video in question and which in the end is the data of interest for the experiment. Hence, it is necessary to use filters in wireshark to show only the data of interest. Fig. 3 illustrates the filters to separate the frames from the video into I, P, B, and audio frames. This process helps to clean the data; however, the different frames that make up a video are sent to the network in interlaced form, that is, a frame tagged as type I may contain in it type P frames, which greatly hinders data extraction; to achieve a complete extraction the software developed in [8] was used.

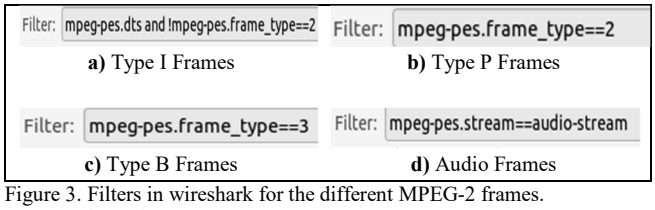


Figure 3. Filters in wireshark for the different MPEG-2 frames.
Source: The authors.

The mathematical characterization detailed process of the traces of traffic, through the probability distribution functions (PDF), is shown in [8] and its results are presented in Table 2.

3. Construction of the simulation environment in NS-3

Three distinct network topologies have been designed in which network performance parameters will be evaluated when streaming previously obtained real traffic in them. These topologies were chosen because they are commonly used in the implementation of LAN networks, but also with the intention of establishing differences between them as far as network performance parameters, like throughput and delay. Fig. 4 shows the distinct network topologies used. Topology 3 has two scenarios; the first of these is designed for a network with 10 users, and the second for 20 users, to analyze the performance of the network when information overload exists. The other two topologies have 10 users each. The number of users was decided, given that by analyzing the characteristics of the videos, if all the users simultaneously request any of these videos, traffic is generated in the order of tenths over any of the topologies.

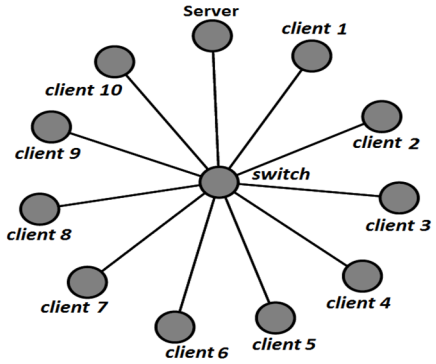
The aim of the simulation is to obtain the results of the performance parameters in the worst of cases, that is, in the hypothetical case of all the users requesting the same video at the same time. With this, it is possible to determine the network resources necessary to serve a number of users in the worst of cases. Hence, the simulation starts when the server begins to transmit the data required by the user and ends when the video has finished. The element necessary to construct the topologies is Topology Helpers.

On a real network, we can find computers with network cards; in ns-3, it could be said that Nodes exist with NetDevices. In a network, it is necessary to configure the connections among nodes, NetDevices, and communication channels.

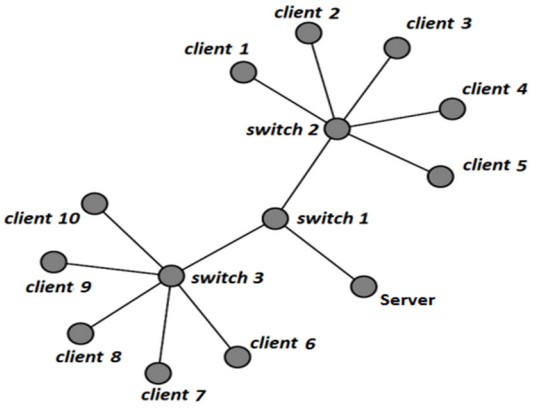
Table 2.
Probability distribution functions characterized by the mathematical behavior of the videos.

			Video1		Video2		Video3	
			T (s)	S (Bytes)	T (s)	S (Bytes)	T (s)	S (Bytes)
Type of frames	I	PDF	Lognormal	Weibull	Logistic	Weibull		
		μ	0.4026	57177.5	0.3993	61379.2		
		Σ	0.0352	6592.3	0.0289	2910.3		
		KS	0.03	0.04	0.06	0.06	N/A	N/A
	P	PDF	Weibull	Lognormal		Weibull	Exponential	Weibull
		μ	0.1358	38446.8	0.1396	3447.4	0.1375	26923.8
		Σ	0.0688	9491.8	0.0972	12343	0.1261	14113.4
		KS	0.09	0.09	0.1	0.07	0.1	0.09
	B	PDF	Gamma	Weibull		Lognormal		Gamma
		μ	0.0508	12270.8	0.0514	10706.5	0.05	59314
		Σ	0.0338	7051.2	0.0559	8272.7	0.0911	3895.4
		KS	0.01	0.1	0.07	0.09	0.05	0.1
	Audio	PDF	Gamma	Logistic	Gamma	Logistic	Weibull	Normal
		μ	0.1484	2743.1	0.1468	2738.3	0.1457	2740
		Σ	0.0466	139.1	0.0614	137.6	0.0933	118.9
		KS	0.04	0.07	0.02	0.07	0.05	0.08

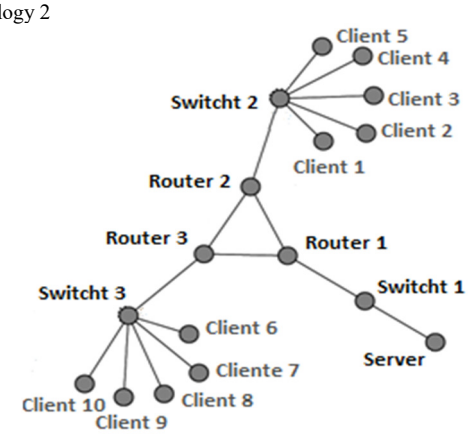
Source: The authors.



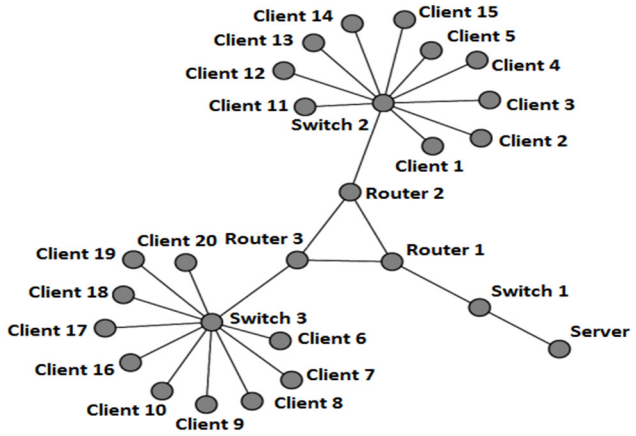
a) Topology 1



b) Topology 2



c) Topology 3, scenario 1



d) Topology 3, scenario 2

Figure 4. Network topologies for case studies.
Source: The authors.

In ns-3, the abstraction of the computing device is called Node; the abstraction is represented in C++ by the Node class, which permits adding functionalities, like applications, protocol stacks, and peripheral cards with their associated drivers. A node can be connected through an object representing a communication cannel; its abstraction is called Channel and it is represented in C++ by the class of the same name.

NetDevices are the network device abstraction that involve the software controller and simulated hardware. A network device is installed to enable communication among nodes through the Channels. A node can be connected to more than one channel through multiple network devices. The abstraction of the network device is represented in C++ by the class of the same name.

Topology helpers permit assigning physical addresses, installing network devices in a node, configuring the node's protocol stack, and – then – connecting the netdevices to the channel. Through the Node Container, node objects are created within ns-3 that will represent the computers in the simulation (Fig. 5).

The first line only declares a node container called nodes. The second line calls the Create method in the node objects and asks the container to create two nodes. The following step in the construction of a topology is to connect the nodes within the network. For example, to construct a point-to-point link, use the topology helpers object and within it use a PointToPointHelper to configure and connect the point-to-point network device objects (PointToPointNetDevice) and point-to-point channel helpers (PontToPointHelperChannel), as shown in Fig. 6.

Next, it is necessary to have a list of the objects, *NetDevice*, for which a *NetDeviceContainer* is used to manage the task. Fig. 7 presents the final configuration of the devices and the channel. The first line declares the device container and the second line of the install method of the point-to-point link helper takes a NodeContainer as parameter.

After executing the pointTopoint.Install call, we will have two nodes, each with a netdevice and a channel between them.

```
NodeContainer nodes;
nodes.Create (2);
```

Figure 5. Node container.
Source: The authors.

```
PointToPointHelper pointToPoint;
pointToPoint.SetDeviceAttribute ("DataRate", StringValue ("5Mbps"));
pointToPoint.SetChannelAttribute ("Delay", StringValue ("2ms"));
```

Figure 6. Construction of point-to-point link
Source: The authors.

```
NetDeviceContainer devices;
devices = pointToPoint.Install (nodes);
```

Figure 7. Final configuration of the devices and the channel.
Source: The authors.

```
InternetStackHelper stack;
stack.Install (nodes);
```

Figure 8. Installation of protocol stack.
Source: The authors.

```
Ipv4AddressHelper address;
address.SetBase ("10.1.1.0", "255.255.255.0");
```

Figure 9. Assignment of IP addresses.
Source: The authors.

```
Ipv4InterfaceContainer interfaces = address.Assign (devices);
```

Figure 10. Association of IP addresses.
Source: The authors.

```
Simulator::Run ();
```

Figure 11. Run the simulation.
Source: The authors.

The following step installs the protocol stack in the nodes (Fig. 8). The helper is the *InternetStackHelper*, in charge of installing the protocol stack in the nodes. The install method takes a node container object as parameter; when it is executed, it installs the internet protocols in each of the nodes.

Next, we need to associate the node devices to the IP addresses. A helper exists to manage the assignment of IP addresses. Fig. 9 presents the two code lines that declare an object helper of addresses. By default, the addresses assigned will star at one and will increase one by one. The ns-3 system at low level remembers all the addresses assigned and will generate an error if by accident the same address is assigned twice.

The following makes associations between IP addresses and the devices using an Ipv4Interface object (Fig. 10).

Now, we have a point-to-point network constructed with protocol stack and IP addresses assigned. At this point, the application needs to generate traffic. This process is presented in the following section.

To run the simulation, use the Simulator::Run global function (Fig. 11). When this function is called, the system will start searching through the list of events programmed and will execute them.

Finally, the cleaning process must be conducted through the global Simulator::Destroy function (Fig. 12).

```
   Simulator::Destroy ();
   return 0;
}
```

Figure 12. Simulation cleaning process.
Source: The authors.

To work according to the ns-3 guidelines, it is necessary to copy the file in the scratch directory where it is compiled through the . */waf* command and it is run through *./waf - -run scratch/miarchivo.*

4. Construction of the model via discrete events in NS-3

Is becomes necessary to create in the ns-3 simulation software a server capable of producing VoD traffic with equal behavior as the traffic produced by the real server. In addition, we must create a client capable of consuming this service and generating statistics of interest for the model. ns-3 has an application that generates data flows denominated OnOffApplication [10].

This application is a traffic generator that follows an OnOff pattern. After the application starts to run, the On and Off states alternate. The duration of each of these states is determined by the value of the OnTime and OffTime variables. During the Off state no traffic is generated, while during the On state traffic is generated.

The attributes to configure to develop the model are: bit DataRate when the application is in the On state, PacketSize of packets sent in the On state, Remote address of destination, OnTime variable used to determine the duration of the On state, OffTime: variable used to determine the duration of the Off state. MaxBytes total size of traffic in bytes, which the application sends, and Protocol to use.

Of the functions obtained, size of frames (I, P, B, and audio) corresponds to the On state of the application and the time between frames corresponds to the Off state.

Fig. 13 defines a helper for OnOff applications, called OnOffHelper. It is in charge of configuring and managing these types of applications. The constructor of the helper receives parameters that determine the characteristics of the service. The first parameter defines a socket for transmission under the user datagram protocol, UDP, (RTSP uses the UDP to transmit multimedia content, and the transmission control protocol, TCP, for control data) [11] and the second is already the socket as such; an IP address and a port.

Besides the aforementioned, a sink helper is defined, called PacketSinkHelper, to receive the data sent. In addition, this helper's constructor receives the same parameters received by the constructor of the helper of OnOff topologies; a chain of characters specifying a *socket* for UDP transmission and the socket as such.

With the SetAttribute method, seen in Fig. 14, the OnOff applications helper sets the application's attributes. This method receives two parameters; el first is a chain of characters specifying the attribute of the application sought to being set or modified, and the second is the attribute as such.

```
OnOffHelper onoff ("ns3::UdpSocketFactory", Address (InetSocketAddress (Ipv4Address::GetAny (), port)));
PacketSinkHelper sink ("ns3::UdpSocketFactory", Address (InetSocketAddress (Ipv4Address::GetAny (), port)));
ApplicationContainer app;
```

Figure 13. Creation of helpers for the OnOff and sink application.
Source: The authors.

```
onoff.SetAttribute ("Remote",AddressValue (InetSocketAddress
(IPv4Address (csmaInterfaces.GetAddress(i)), port)));
```

Figure 14. Configuration of the remote attribute for the OnOff application.
Source: The authors.

```
onoff.SetAttribute("OffTime",StringValue("ns3::
LogNormalRandomVariable[Mu=0.4026,Sigma=0.0352]"));
onoff.SetAttribute("OnTime",StringValue("ns3::
WeibullRandomVariable[shape=10.2063,Scale=57480.9]"));
onoff.SetAttribute("MaxBytes",UintegerValue (10989173));
```

Figure 15. Configuration of the OffTime, OnTime, and MaxBytes attributes for the OnOff application.
Source: The authors.

```
sink.SetAttribute ("Local", AddressValue (InetSocketAddress
(Ipv4Address (csmaInterfaces.GetAddress(i)), port)));
app = sink.Install (terminals.Get (i));
app.Start (Seconds (0.0));
app = onoff.Install (terminals3.Get (1));
app.Start (Seconds (1.0));
```

Figure 16. Installation of applications in the nodes.
Source: The authors.

```
sink.SetAttribute ("Local", AddressValue (InetSocketAddress
(Ipv4Address (csmaInterfaces.GetAddress(i)), port)));
app = sink.Install (terminals.Get (i));
app.Start (Seconds (0.0));
```

Figure 17. Data reception in clients.
Source: The authors.

The code shown in Fig. 15 sets the Offtime, Ontime, and Maxbytes attributes. The first two attributes of the application receives, as parameter through the SetAttribute method, a probability function, as shown in Fig. 15. This function was obtained from the characterization of the video components. For this specific case, the Lognormal function represents the time between type I frames from video1, while the Weibull function represents their size. For videos 2 and 3, the process is analogous.

With the instructions shown in Fig. 16, an application is extracted from the application container, the parameters established previously are configured in the OnOff applications helper, it is installed in the server node, and the time is set in which the function starts to send data.

For data to be received in the client nodes it is necessary to adhere a sink application (Fig. 17); the SinkHelper, through its SetAttribute method, can configure the client nodes to receive data. The Local attribute refers to the node where the data is received, and the following parameter of the method is a socket. Thereafter, an application is taken from the application container and it is installed in the node it will receive. Finally, the time is set, in which the node is ready to receive data.

5. Analysis and results

The statistics obtained through the ns-3 FlowMonitor application are total values along the simulation. These are:

```
std::cout << "  Tx Bytes: " << st.txBytes << std::endl;
std::cout << "  Rx Bytes: " << st.rxBytes << std::endl;
std::cout << "  Tx Packets: " << st.txPackets << std::endl;
std::cout << "  Rx Packets: " << st.rxPackets << std::endl;
std::cout << "  Lost Packets: " << st.lostPackets << std::endl;
```

Figure 18. Configuration of statistics.
Source: The authors.

```
std::cout << "  Mean{Delay}: " <<
(st.delaySum.GetSeconds() / st.rxPackets) << std::endl;
std::cout << "  Throughput: " << st.rxBytes * 8.0 /
(st.timeLastRxPacket.GetSeconds() - st.timeFirstTxPacket.GetSeconds
          ())<< " Mbps\n";
```

Figure 19. Mean value of delay and throughput.
Source: The authors.

```
Ptr<FlowMonitor> monitor;
FlowMonitorHelper flowmon_helper;
flowmon_helper.SetMonitorAttribute("StartTime",
TimeValue(Seconds(1)));
monitor = flowmon_helper.InstallAll();
```

Figure 20. Creation of FlowMonitor.
Source: The authors.

bytes transmitted and received, packets transmitted received and lost. Fig. 18 shows the configuration of these statistics.

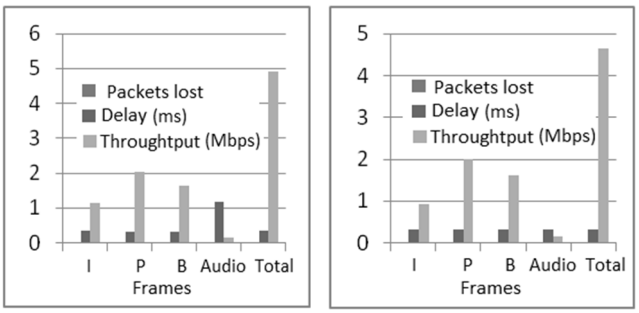
In addition, the delay and throughput as mean value are obtained. Fig. 19 shows the configuration of these parameters, while Fig. 20 shows the code lines that create the flow monitor that must be installed in the network previously created.

5.1. Obtaining the throughput

Throughput is obtained by knowing three parameters. The first is the time in which the first packet was transmitted; the second is the time in which the last packet was received; and the third was the number of packets received. Two TraceSources (Fig. 21) exist in the network device; these are useful to obtain throughput through samples. Fig. 21 presents the code used to measure throughput in the network. For the first packet through PhyTxBegin the FirstPacket (TraceSink) function is called whenever it is transmitted by the channel; said function registers the time in which it is transmitted. For the second packet through PhyRxEnd, the LastPacket (TraceSink) function is called each time it is received in the client node. Thus, this function registers the time in which it is received and registers the sum of packets received to that moment in the client node. These two instructions register the throughput, for example, within a switch node for a client node connected to it.

```
terminalDevices3.Get (1)->TraceConnectWithoutContext
("PhyTxBegin", MakeCallback (&PrimerPaquete));
switchDevices3.Get (1)->TraceConnectWithoutContext
("PhyRxEnd", MakeCallback (&UltimoPaquete));
```

Figure 21. Installation of trace sources.
Source: The authors.



a) Client 2 b) Client 5

Figure 22. Topology 1. Packets lost, delay and throughput.
Source: The authors.

5.2. Analysis of results for topology 1

Fig. 22 shows the results obtained of loss of packets, delay and throughput for each of the types of frames of clients 2 and 5 upon simulating video1 over topology1.

None of the clients had loss of packets. Delay values are 0.33 ms for client 2 and 0.3 ms for client 5. The parameter values of Fig. 22 may be different to clients since these values correspond to an average of multiple simulations, we set the seed differently. These values are within the ranges permitted to offer VoD with quality of service (QoS), according to [12-14] which specify a value below 20 ms for a local environment.

Fig. 22.a shows that throughput between the switch and client 2 has a mean of 4.9 Mbps and Fig. 22.b shows that throughput between the switch and client 5 it has a mean of 4.6 Mbps. Throughput for the link between the server and el switch is on average de 47.9 Mbps, as seen in Fig. 23.

Tables 3 and 4 consign the results obtained for the network performance parameters from the simulation of each video for topology 1.

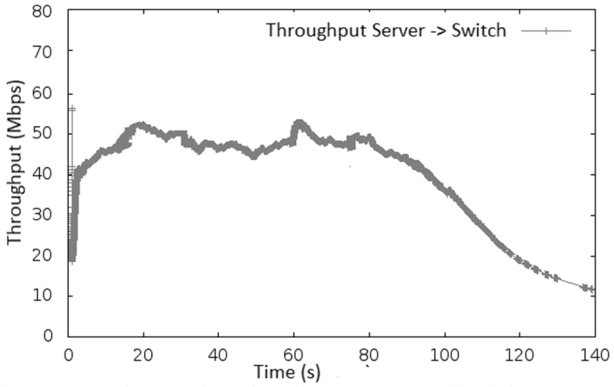


Figure 23. Topology 1. Throughput between server and switch. Source: The authors.

Table 3.
Topology 1, mean delay values for clients 2 and 5.

	Delay (ms)		
	Video 1	**Video 2**	**Video 3**
Client 2	0.33	0.5	0.31
Client 5	0.3	0.47	0.25

Source: The authors.

Table 4.
Topology 1, mean throughput values.

	Throughput (Mbps)		
	Video 1	**Video 2**	**Video 3**
Server -> Switch	47.9	51	26
Switch -> Client2	4.9	5	2.6
Switch -> Client5	4.6	5.1	2.6

Source: The authors.

According to Table 3, and taking as reference the design of networks with end to end service quality [14], the videos under these characteristics comply with the QoS parameter for the delay.

Observe that with respect to the throughput, the highest value appears for video 2; this may be explained by the degree of precision the PDF have with the real traffic, as noted in Table 2. Likewise, it may be seen that the necessary transference rate is around 51 Mbps between the server and the switch when all clients simultaneously request video two. This is an extreme case, which is why a network planner with the characteristics of topology 1 and a bandwidth in the order of tenths and above 51 Mbps – not a high value for current data networks – can offer VoD service with QoS.

5.3. Analysis of results for topology 2

Tables 5 and 6 present the results obtained for the network performance parameters, from the simulation of each video for topology 2.

Note that topology 2 also does not have problems regarding delay and its maximum value is below 20 ms. Its increase with respect to topology 1 may be explained by the presence of two switches between the server and the end client. With respect to throughput, note that the maximum value is quite close to the value of topology 1; this is because the number of clients is the same and only change the topology. According to the aforementioned, for a network planner the same concepts emitted for topology one are valid.

Table 5.
Topology 2. Delay for clients 3, 4, 7, and 9.

	Delay (ms)		
	Video1	**Video2**	**Video3**
Client 3	0.45	0.47	0.37
Client 4	0.45	0.46	0.38
Client 7	0.45	0.48	0.37
Client 9	0.47	0.46	0.38

Source: The authors.

Table 6.
Topology 2. Mean throughput values.

	Throughput (Mbps)		
	Video1	**Video2**	**Video3**
Server -> Switch1	47.9	50.8	25.8
Switch1 -> Switch2	23.6	25.8	13
Switch2 -> Client3	4.9	4.83	2.6
Switch2 -> Client4	4.8	4.9	2.6
Switch3 -> Client7	4.6	4.9	2.6
Switch3 -> Client9	4.9	5,1	2.6

Source: The authors.

Table 7.
Topology 3, Scenario 1. Delay.

	Delay (ms)		
	Video 1	Video 2	Video 3
Client 3	2.67	2.6	2.6
Client 4	2.68	2.6	2.6
Client 7	2.7	2.7	2.6
Client 9	2.7	2.7	2.6

Source: The authors.

Table 8.
Topology 3, Scenario 1, mean throughput values.

	Throughput (Mbps)		
	Video 1	Video 2	Video 3
Server -> Router1	47	49.5	24.8
Router1 -> Router2	23.7	24.8	12.2
Switch2 -> Client3	4.7	4.6	2.6
Switch2 -> Client4	4.7	5.5	2.5
Swtich3 -> Client7	4.7	5.2	2.5
Switch3 -> Client9	4.5	5.9	2.5

Source: The authors.

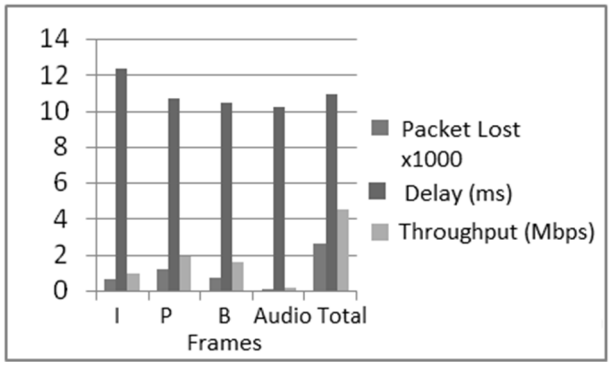
5.4. Analysis of results for topology 3 scenario 1

Tables 7 and 8 display the results obtained for the network performance parameters from the simulation of each video for topology 3, scenario1.

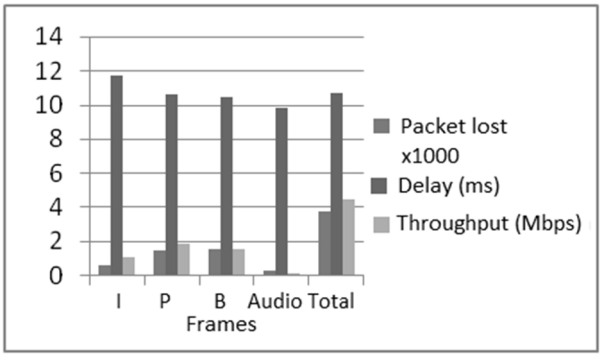
For topology 3 scenario 1, it may be noted that the delay increased drastically reaching a maximum value of 2.68 ms; however, it does not exceed the limit stipulated in [13,14]. This increase can be explained through the increase of intermediate devices between the server and the end client. For its part, throughput is within the values found for topologies 1 and 2.

5.5. Analysis of results for topology 3, scenario 2

The following shows the graphics corresponding to the results found for the network performance parameters, for the simulation of video 1 over topology 3, scenario 2. Fig. 24 illustrates the results obtained from loss of packets, delay, and throughput for clients 3 and 4. Note that on this occasion there was a considerable loss of packets for both clients, with 2608 packets lost for client 3 and 3681 packets lost for client 4. If the maximum size of the packets is 1478 (Ethernet frame size minus the header of the layer 3 and layer 4 protocol) and bearing in mind the size of the video shown in Table 1, it can

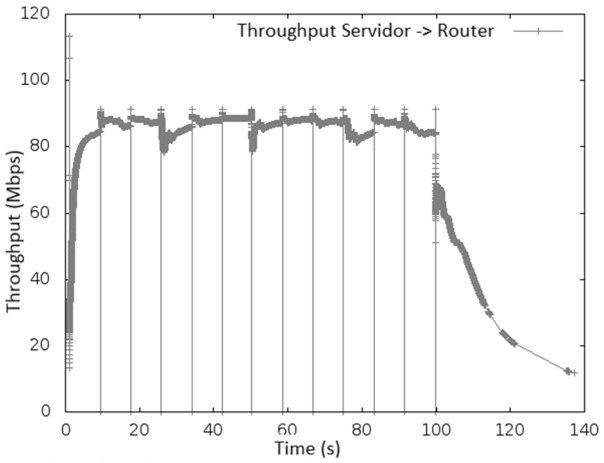


a) Packets lost, delay, and throughput for client 3

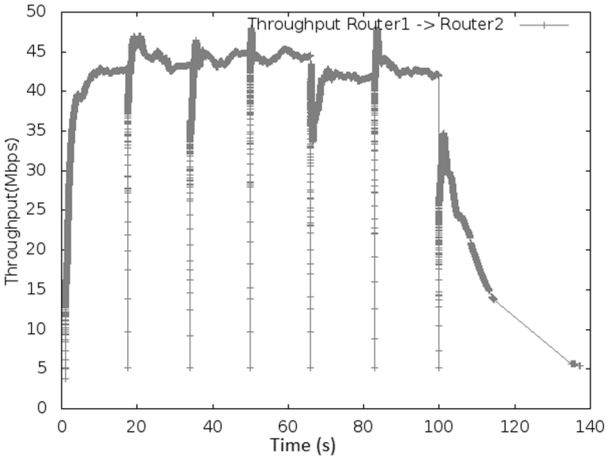


b) Packets lost, delay, and throughput for client 4

Figure 24. Topology 3 scenario 2. Packets lost, delay, and throughput; clients 3 and 4.
Source: The authors.



a) Throughput between server and router 1



b) Throughput between router 1 and router 2

Figure 25. Topology 3 scenario 2, throughput.
Source: The authors.

be concluded that loss of packets for client 3 is 6.17%, while for client 4 it is 8.72%; these values are above the 5% accepted for loss of packets in a local area network, according to [13].

It is also noted in Fig. 24 that the delay was on average 10.9 ms for client 3 and 10.7 ms for client 4, and throughput

Table 9.
Topology3 – Scenario 2. Delay for clients 3, 4, 7, and 9.

	Delay (ms)		
	Video 1	Video 2	Video 3
Client 3	10.9	9.3	2.6
Client 4	10.7	10	2.6
Client 7	10.6	10	2.7
Client 9	10.7	8.4	2.6

Source: The authors.

Table 10.
Topology3 – Scenario 2, Mean throughput values.

	Throughput (Mbps)		
	Video 1	Video 2	Video 3
Server -> Router1	89	89.8	50.4
Router1 -> Router2	44.4	45.3	25
Switch2 -> Client3	4.5	43	2.6
Switch2 -> Client4	4.2	4.9	2.5
Switch3 -> Client7	4.5	4.5	2.7
Switch3 -> Client9	4.5	4.2	2.6

Source: The authors.

was approximately 4.5 Mbps for client 3 and 4.2 Mbps for client 4. Also, as observed in Fig. 25.a, throughput was approximately 89 Mbps for the link between server and router 1, and 44.4 Mbps between router 1 and router 2, as noted in Fig. 25.b.

With respect to the delay, it is observed that this topology complies with the QoS parameters specified in [13,14]. Throughput presents a maximum value and reaches a value of 89.8 Mbps; this is an extreme case, which is why a network planner with the characteristics of topology 3 scenario 2 will require a bandwidth around 90 Mbps, which is not a high value for current local area data networks. However, under these conditions, QoS cannot be guaranteed due to the loss of packets exceeding the maximum permitted.

Tables 9 and 10 present the results obtained for the network performance parameters from the simulation of each video for topology 3 scenario 2.

6. Conclusions

The model described in this article permits replication and reconfiguration, according to the topology desired and the very characteristics of the services requested. For example, if a network planner wishes to provide the VoD service with QoS to users with conditions similar to those described in topology 3 scenario 2, it will be necessary to seek alternatives, like increasing bandwidth or using mirror servers.

Measurement of network performance parameters, like throughput and delay, is important in the planning and implementation of networks, given that these indicate how well the network is working with the load it must support, in addition to being important parameters for QoS.

The processing level of each topology influenced upon the performance delay parameter, showing that inasmuch as the topologies become more complex, the value of said parameter also increases.

Note that the network parameters measured between the server and the first device to which it is connected, and between a switch and a client, remain constant in spite of the change of topologies, which permits verifying how these do not influence upon these links.

The delay values in topologies 1 and 2 remained below a millisecond. The topology showed a delay increase caused mainly by adding routers and links between these, besides the delays of the links between the routers and the processing of the packets in each of the network devices.

Topology 3 scenario 2 displayed a percentage of loss of packets above 5% of the total and the delay value exceeds 10 milliseconds. In conclusion, topology 3 scenario 2 is not capable of supporting simultaneous consumption of the videos.

Loss of packets occurred because the queue size that devices can support in ns-3 is insignificant; thus, it is necessary to manage or bear in mind the buffer size in real devices. Network equipment have buffers that store much information in the queue to prevent loss of packets; the size of these buffer can vary according to the model or brand of the network equipment and can even be configured.

It is proposed as future work the validation of the results through the development of the experiments in other simulation tools such as OMNeT++ or OPNET Modeler as tools capable of supporting the video service modeling.

Acknowledgement

To Project 839 of the GITUQ group, financed by Vicerrectoria de investigaciones of the Universidad del Quindío.

References

[1] Savannah, B. and Osborne. Your video streaming guide [On line], 2011 [consulted, 11th July 2016]. Available at: https://store.kobobooks.com/en-us/ebook/your-video-streaming-guide.

[2] Cisco. White paper: Cisco VNI forecast and methodology 2015-2020. San Jose CA [Online]. 2016. [consulted, 11th July 2016]. Available at: http://www.cisco.com/c/dam/en/us/solutions/collateral/service-provider/visual-networking-index-vni/complete-white-paper-c11-481360.pdf.

[3] Iversen, V.B., Teletraffic engineering and network planning, COM DTU [online], 2015. [consulted, 12th July 2016] Available at http://orbit.dtu.dk/files/118473571/Teletraffic_34342_V_B_Iversen_2015.pdf

[4] Yin, P., Criminisi, A., Winn, J. and Essa, I.A., Bilayer segmentation of webcam videos using tree-based classifiers. IEEE Transactions on Pattern Analysis and Machine Intelligence, [Online], 33(1), pp. 30-42. 2011. [consulted, November 17th of 2016]. Available at: http://ieeexplore.ieee.org/document/5432210/ DOI:10.1109/TPAMI.2010.65.

[5] Inostrosa-Psijas, A., Gil-Costa, V., Solar, R. and Marín, M., Load balance strategies for DEVS approximated parallel and distributed discrete-event simulations, [Online]. 23rd Euromicro International Conference on Parallel, Distributed, and Network-Based Processing, pp. 337-340, 2015. [consulted, 17th November 2016] Available at: http://ieeexplore.ieee.org/document/7092741/ DOI: 10.1109/PDP.2015.13.

[6] Chistyakov, A., A software architecture for large multi-simulation Experiments over ad hoc networks using NS-3 discrete-event network simulator. European Modelling Symposium (EMS), 2014. pp. 403-408. DOI: 10.1109/EMS.2014.71.

[7] Riley, G.F. and Henderson, T.R., The ns-3 network simulator. modeling and tools for network simulation, in: Wehrle, K., Güneş, M.

and Gross, J., Eds., Springer Berlin Heidelberg, Berlin,[online]. 2010, pp. 15-34, [consulted, 2 October 2016]. Available at https://link.springer.com/book/10.1007%2F978-3-642-12331-3 DOI: 10.1007/978-3-642-12331-3.

[8] Campo, W., Modelo de tráfico para servicios interactivos de una comunidad académica virtual, con contenidos de audio y video de alta calidad, Tesis Dr., Departamento de Ingeniería Telemática, Universidad del Cauca, Popayán, Colombia, [online]. 2014. Available at: https://drive.google.com/file/d/0Bw_sZ3xa3yAsOFp2bnF5ZUs3TWc/view

[9] Solomon, T., Zungeru, A.M. and Selvaraj, R., Network traffic monitoring in an industrial environment. Third International Conference on Electrical, Electronics, International Conference on Electrical, Electronics, Computer Engineering and their Applications (EECEA), 2016. pp. 133-139. DOI: 10.1109/EECEA.2016.7470779.

[10] Assyadzily, M., Suhartomo, A. and Silitonga, A., Evaluation of X2-handover performance based on RSRP measurement with Friis path loss using network simulator version 3 (NS-3). 2nd International Conference on Information and Communication Technology (ICoICT), 2014. pp. 436-441. DOI: 10.1109/ICoICT.2014.6914102.

[11] Chanchí-Golondrino, G.E., Urbano-Ordoñez, F.A. y Campo-Muñoz, W.Y., Pruebas de estrés para servicios de videostreaming basados en el protocolo RTSP. Revista Tecnura [Online]. 19(46), 2015. [consulted, 5 August 2016]. Available at: http://revistas.udistrital.edu.co/ojs/index.php/Tecnura/article/view/9543.

[12] Higher Layer LAN Protocols Working Group. 802.1Q-2014 - IEEE Standard for local and metropolitan area networks-bridges and bridged networks. IEEE Std 802.1Q. 2014. [online]. Available at: http://ieeexplore.ieee.org/document/6991462/ DOI: 10.1109/IEEESTD.2014.6991462

[13] ITU., G.1010: End-user multimedia QoS categories. [online], International Telecommunication Union, 2001 [consulted 15th July 2016], Available at: https://www.itu.int/rec/T-REC-G.1010-200111-I/

[14] Szigeti, T. and Hattingh, C., End-to-end QoS network design: Quality of service in LANs, WANs, and VPNs. Cisco Press, 2004.

W.Y. Campo-Muñoz, is Electronics Eng. from Universidad del Cauca, Colombia (1998), MSc. in Engineering, Area of telematics, Universidad del Cauca, Colombia (2009), PhD in Telematics Engineering from Universidad del Cauca, (2014). Is professor at Universidad del Quindío, Colombia, Electronics Engineering program, researcher with the Telecommunications Research group at Universidad del Quindío – GITUQ, Colombia. Areas of interest: telematics, IPTV, traffic modelling of telematics services.
ORCID: 0000-0001-8585-706X

E. Astaiza-Hoyos, is Electronics Eng. from Universidad del Cauca, Colombia (1998). MSc. in Engineering, area of telecommunications, Universidad del Cauca, Colombia (2008). Current PhD candidate electronics sciences. Is associate professor at Universidad del Quindío, Colombia, Electronics Engineering program, researcher with the GITUQ. Areas of interest: wireless communications, spectrum sensing.
ORCID: 0000-0003-2706-0962

L.F. Muñoz-Sanabria, is MSc. in Computing, Universidad del Cauca, Colombia (2013), Systems Eng. Universidad Antonio Nariño, Colombia (1997). PhD in Electronics Sciences. Is associate professor at Fundación Universitaria de Popayán, Systems Engineer program, researcher with the LOGICIEL. Areas of interest: communications engineering, information engineering.
orcid: 0000-0002-8172-0530