A method for validating the aggregated signaling traffic model using simulations in ns-2 platform

Jordi Mongay Batalla
Warsaw University of Technology
ul. Nowowiejska 15/19
00-665 Warsaw, Poland
+48-22-2347564
jordim@interfree.it

Robert Janowski
Warsaw University of Technology
ul. Nowowiejska 15/19
00-665 Warsaw, Poland
+48-22-2347564
rjanowsk@tele.pw.edu.pl

ABSTRACT
In the architectures of the networks, which offer Quality of Service (QoS), the setup procedure is the exchange of signaling messages to agree the codec of the multimedia applications and to reserve the resources in the network. This exchange of packets between two or more signaling entities causes that this traffic strongly depends on the network scenario and network conditions. Therefore, it is very difficult to find general signaling traffic models, which are valid for any scenario.

In this paper we propose a method for validating the aggregated signaling traffic model. This validation is performed using our own developed module of the ns-2 simulator, which simulates the signaling message exchange during the connection setup in an IP based network with QoS guaranties. For this new simulation tool, we provide technical description and discuss some implementation issues under ns-2 platform. The final results of the validation process show under what traffic conditions the versatile Poisson model is valid. The drawn conclusions provide some guidelines for the usage of analytical models in the network provisioning process concerned with signaling traffic.

Categories and Subject Descriptors
C.2.2 **[Network Protocols**]: Signaling in Progress (SIP); Next Steps in Signaling (NSIS)

I.6.7 [**Simulation support systems**]: The Network Simulator (NS-2)

SIMUTOOLS 2008, March 03-07, Marseille, France

DOI 10.4108/ICST.SIMUTOOLS2008.2984

General Terms
Performance, Design, Experimentation.

Keywords
Signaling, ns-2, Class of Service, QoS, SIP.

1. INTRODUCTION
In contrast to many different traffic models for multimedia traffic (VoIP [5], VoD [6]), the Internet community has not defined valid models for signaling traffic. The signaling traffic is sporadic, variable and strongly depends on the signaling application and reliability conditions. Some proposals for modeling this kind of traffic have been presented but they are based on concrete examples and they cannot be extensively used. An example can be found in [7].

Unluckily, these proposals are not useful to model the signaling traffic inside the EuQoS system.

The IST FP6 End-to-End Quality of Service (EuQoS) project [8] [9] proposes, analyses and implements a new architecture based on different Classes of Service (CoS) to ensure Quality of Service from end user to end user over heterogeneous networks. The different classes of service are designed according to the necessities of the multimedia applications considered in the EuQoS project and following the guidelines of the lately appeared RFC 4594 [10]. One of these classes of service is the responsible for carrying signaling traffic and is named Signaling Class of Service (S-CoS).

In [11] is presented an efficient method to provision the S-CoS. This method determines the resources on the links where the S-CoS is supported to ensure target maximum setup latency. Nonetheless, the method from [11] does not provide the guidelines for provisioning the buffer resources since it requires a valid model of the aggregated signaling traffic.

This paper wants to be the conclusion of the provisioning method of the S-CoS. We will determine, by means of simulations, the behavior of the *aggregated* signaling traffic carried by the S-CoS and will demonstrate that we can model the aggregated signaling traffic as Poisson traffic. In this way, it is possible to easily provision the buffer resources of the S-CoS by using the model M/G/1.

For this purpose, we develop and implement modules in the ns-2 environment that simulate the signaling message exchange considering the signaling protocols involved in the EuQoS setup procedure. The implemented modules resulted a very useful tool to understand the behaviour of the signaling traffic in the network, and to take conclusions for implementing the Signaling Class of Service in the EuQoS system.

The rest of this paper carries on as follows: In Section 2, we explain the characteristics of the network and signaling traffic inside the EuQoS system and we propose the network model to perform the simulations. In Section 3, the implemented ns-2 [13] modules for the signaling traffic are presented. Next, in Section 4 we explain the method to validate the simulation model against the Poisson model and expose the results. In Section 5, we finish the paper with some conclusions.

2. TRAFFIC AND NETWORK FEATURES

This text focuses on the two signaling protocols involved in the EuQoS setup procedure: Session Initiation Protocol (SIP) [12] and Next Steps In Signaling (NSIS) [15].

The end users, who try to establish a connection, communicate with the SIP Proxies using the SIP protocol. The SIP Proxies trigger the network resource reservation contacting the Resource Managers (RM). There is one Resource Manager in each domain and the resource reservation is performed step by step. The RMs communicate between themselves using NSIS protocol.

The message exchange sequence of the signaling procedure is presented in Fig. 1. In this figure, we can see the SIP messages and the shadded NSIS messages. We consider only two domains and, to simplify, the SIP proxy and the RM are located at the same machine.

We model the signaling path as one unique bottleneck between proxies.

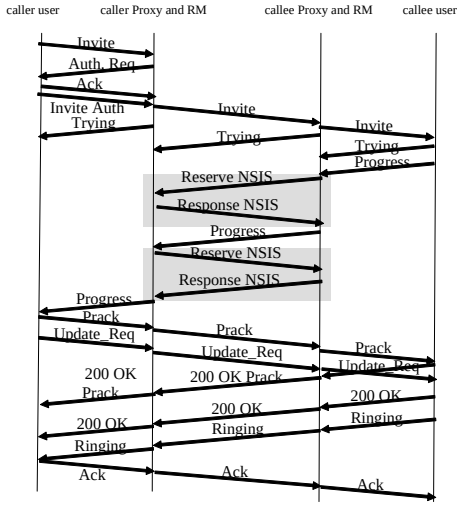

Fig. 1 Message sequence of the setup procedure in EuQoS

The S-CoS is implemented along the network path between the users and proxies and between proxies. In our simulations we will take the simplest scenario, i.e. there is no domain between end domains and no other Resource Manager is in the way (only in the end domains). This network scenario is presented in Fig. 2. The user *i* in the Ingress Domain establishes a setup connection with the user *i* in the Egress Domain.

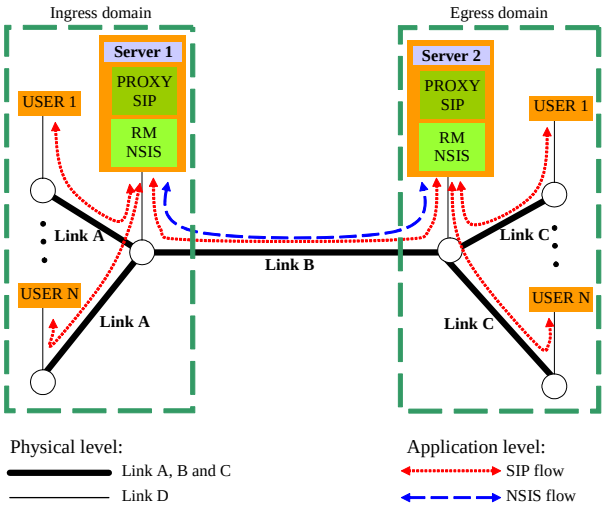

Fig. 2 Simulation network model

Table 2-1 presents the capacities of the links A, B and C calculated with the provisioning method [11] for ensuring target maximum transfer packet setup latency= 3 seconds and the call blocking probability= 0.005 (in case of the lack of network resources for the S-CoS, see [4]). The transfer packet setup latency is the time needed to complete the packet exchange of setup procedure, which takes into account the packet transferring times over the network but not the processing times in the signaling entities. In [4] the recommended value for the whole setup latency

(transmission and processing time) equals 11 seconds and the EuQoS studies stablished that the value of the transfer packet setup latency must be lower or equal to three seconds.

Since the messages in both directions are different, the capacities of the links A, B and C are different in both directions (see Table 2-1). The capacity of the links D is significantly higher and equals 1 Gbps. Note that the capacity of the link B depends on the number of users *N* in the system. Thus, the predicted number of users is a key point when provisioning the S-CoS.

Table 2-1 Link capacities calculated using the provisioning method

Link	Link A		Link B		Link C	
Direction	Ingress to Egress	Egress to Ingress	Ingress to Egress	Egress to Ingress	Ingress to Egress	Egress to Ingress
Capacity [kbps]	64.68	40.25	*N* x 47.38	*N* x 42.21	47.38	56.08

All the users connected to the caller SIP proxy make up the ingress user population. We consider infinite-size user population and hence, in the performed simulations, the arrivals of setup procedures follow a Poisson process [14]. In order to have on average *N* users present in the system simultaneously, we proceed as follows: since the setup procedure lasts, at the most, the target maximum transfer packet setup latency= 3 seconds, then the mean rate of the arrival of setup procedures must equal *N*/3 arrivals per second. This ensures that on average *N* users are together in the system. However, this does not prevent that in any moment more than *N* users could be together in the system. Therefore, in the simulations, the number of terminals, which can handle the user setup procedures must be much greater than *N*.

From now on we consider that in the routers the capacities of each CoS are well separated by a WFQ scheduler. Furthermore, the nodes or segments of network not supporting these mechanisms must be over-provisioned to avoid non-controllable bottlenecks. Therefore, we may treat the Signaling CoS as independent of the other CoSs.

3. SIMULATION TOOL

The simulator SIM-EuQoS-PTL is a tool implemented in the ns-2 environment. EuQoS Consortium created this tool to integrate in one unique simulator the packet level mechanisms of the EuQoS system. SIM-EuQoS-PTL integrates different network techniques (WLAN, WiFi, xDSL, IP Core), the mechanisms for all Classes of Service like policing, shaping, scheduling, and the different traffic models for multimedia applications.

In addition, we developed and integrated modules to simulate the signaling process in the EuQoS system. The integration of these modules permits its use in any network scenario (even in environments with different network techniques). These modules, which we are now presenting, simulate the signaling protocols: SIP and NSIS.

The elements responsible for simulating the signaling message exchange i.e. the generation, transferring and reception of signaling messages are implemented both in C++ and in OTcl which is in accordance with the overall philosophy of ns-2 [13] simulation platform. OTcl mainly provides the user interface to the functional elements which are hard coded in the C++ for the reasons of performance efficiency and convenience. Using OTcl commands a user can easily create and configure the elements responsible for simulating the signaling system while not going into details of their functional design and implementation.

The core elements of the signaling system are signaling applications and specialized transport agents.

Signaling applications implement the state machine for the SIP protocol, separately for the SIP User and the SIP Proxy. Each of them can act either as a Caller (the side which initiates a call) or a Callee (the side which receives the call). In case of the setup procedure the choice of acting roles implies the direction of setting up the connection. In the Fig. 3 which depicts mutual relationships between transport agents and signaling applications, the connection will be set up from the left side to the right side.

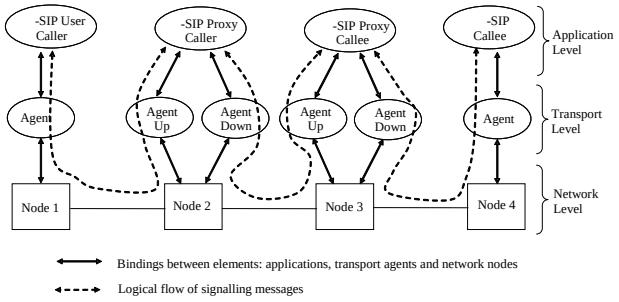

Fig. 3 Mutual relationship between transport agents and signaling entities available in OTcl

SIP Proxy application also implements the state machine for the NSIS protocol. However it is only a part of NSIS protocol related to the network resource reservation phase during the call invocation process. Therefore this part of the state machine of NSIS protocol is jointly implemented with SIP state machine.

Fig. 4 presents, in SDL (Specification and Description Language) convention, a part of the state machine corresponding to the setup procedure of the Callee User.

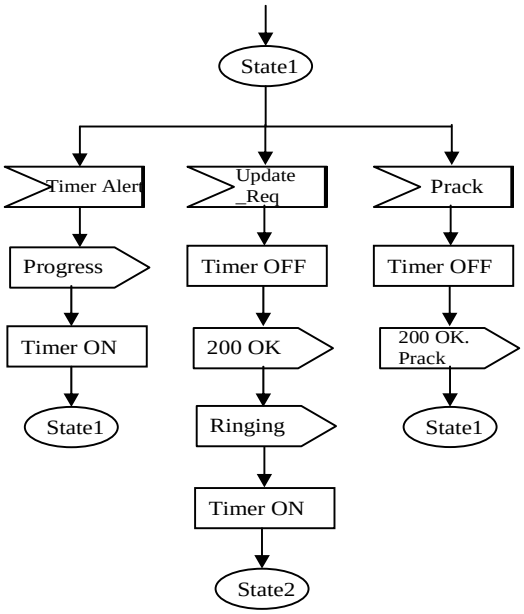

Fig. 4 Part of the State Machine of Callee User (SDL lang.)

Fig. 4 starts with State1, to which the state machine of a Callee User transits after the reception of Invite message and successful transmission of two other messages in response: Trying and Progress (see Fig 1). In State1 there are 3 possible events to occur: Timer Alert due to a timeout event, the arrival of Update-Req or Prack messages. In case of the timeout event it is assumed that the previously sent Progress message has been lost (or the Prack message sent in response from the Caller User was lost), so the Progress message is resent and the timer is rescheduled. In case of receiving Prack message it is implicitly known that the Progress message must have safely reached the Caller User so the timer is canceled. After this, 200 OK Prack message is sent and the state machine transits again to State1. It is still in State1 because, if this message is lost, Prack message will be repeated by the Caller User and the Callee User must respond in the same way. Only the reception of Update_Req message makes the state machine transit to the State2. Previously, the timer is cancelled, and two messages are sent: 200 OK and Ringing.

In Fig. 5, we present a part of the SIP module implementing in C++ the state machine corresponding to the Fig. 4 (only the arrival of Update_Req message). We can see the value of the SIP timer = *500 ms* following the guidelines of the SIP protocol in [12].

```
else if (sh_buf->type == 16 && process_state==1){
//UPDATE_Req has arrived
  //Timer OFF
   hash_progress* hashptr = del_hashprogress_elem_(sipseqno, from, to);
   if (hashptr) {
       hashptr->timerClient->cancel();
       delete hashptr;}
   //200 OK
   send_200ok_msg_(sipseqno, from, to);
   //Ringing
   send_ringing_msg_(sipseqno, from, to);
   //Timer ON
   RINGING_Timer_Client* ringing_tmr =
              new RINGING_Timer_Client(this);
   ringing_tmr -> seq = present_seq;
   ringing_tmr -> to = to_;
   ringing_tmr -> from = from_;
   ringing_tmr -> retrans = 0;
   ringing_tmr -> resched(0.500); //Timer value=500 ms
   hash_ringing* hash_ringing_ptr = new hash_ringing;
   hash_ringing_ptr -> timerClient = ringing_tmr;
   hash_ringing_ptr=%d"<<hash_ringing_ptr<<std::endl;
   hash_ringing_ptr -> seq = present_seq;
   hash_ringing_ptr -> from = from_;
   hash_ringing_ptr -> to = to_;
   hash_ringing_ptr -> next = 0;
   if (!hash_ringing_head) {
        hash_ringing_head = hash_ringing_ptr;
        hash_ringing_tail = hash_ringing_ptr;
        hash_ringing_head=%d"<<hash_ringing_head<<std::endl;
        } else {
        hash_ringing_tail -> next = hash_ringing_ptr;
        hash_ringing_tail = hash_ringing_ptr;
        }
   //State2
   process_state=2;
}
```

Fig. 5 SIP module for State1 – Arrival of Update_Req

Since SIP Proxy communicates with two sides i.e. one SIP User and the other SIP Proxy, its implementation requires that application can bind to two transport agents (Agent Up and Agent Down as depicted in Fig. 3) each for realizing communication in a different direction. Typically in ns-2 an application was bound to only one transport agent which uniquely determined the direction of sending the messages (as for example the SIP User in the Fig. 3). In our case the direction of sending a message is concluded from the context of the message exchange scenario and the Agent Up or Agent Down is chosen accordingly.

The behavior of an SIP User or SIP Proxy state machine can be controlled by configuring some parameters. The main OTcl parameters are: siptimer_, caller_, active_ and tracing_. The siptimer_ parameter turns on the usage of timers at the application level. These timers follow the behavior outlined in [12] i.e. they grow exponentially after each message loss with default timer granularity of 500ms. If the timers are turned off and UDP protocol is used then any lost message (due to the lost packet) will make the signaling procedure unable to be finished. Thus, the timers should be turned on unless a reliable transport protocol as TCP is used. The caller_ parameter determines the role of

the signaling entity. The value equal 1 means that the given signaling entity acts as the caller while other values mean that it acts as the callee. The active_ parameter indicates if the given SIP User is currently handling a signaling procedure or it is idle. This information might be used when new call requests arrive to the system and the connection setup procedure must be handled by one out of *N* SIP Users. The tracing_ parameter indicates if the application level tracing is turned on. It lets for collecting the information about all transferred messages e.g. source, destination, type of message and time, which can be later used to calculate the call level metrics e.g. setup latency.

The implementation of SIP Proxy and SIP User applications is not the same when using UDP or TCP as the transport protocol. This means that there are two sets of signaling applications destined to be used with a specific transport protocol. This solution was dictated by the ease of implementation under the ns-2 platform although such a solution is less flexible (it requires double number of application objects and it denies the rule of separation between the application and the transport used). The main problem and, thus the difference, lies in the way the information about the signaling messages is relayed between signaling applications. In case of UDP protocol it was much easier to develop a specialized UDP transport agent and encode this information in the packet headers. Implementation of a specialized UDP agent simply extends the common UDP agent available in ns-2 tool by adding new fields in the packet header to carry the information related to the signaling message. This information covers the type of signaling message, the message sequence number and the unique identification of the sending and receiving entity.

In case of TCP it was easier to use the existing general API to TCP protocol in ns-2 simulator and to develop a method for retrieving the information about signaling messages solely in the application level instead of modifying the fairly complex implementation of full (duplex) TCP agents. For this purpose, we used the scheme presented in Fig. 6. In this scheme the sender signaling application following its state machine only triggers a generation of TCP packets by means of standard ns-2 API to TCP (step 1). The signaling message which is to be transferred is not sent but kept locally. Only the appropriately sized TCP packets are sent over the network and received by the corresponding TCP agent at the remote signaling application for which a given signaling message is destined (step 2). On the packet reception the remote TCP agent relays the information about this event to the signaling application (step 3). Then the receiver application contacts (in the direct off-line communication, i.e. not over the network) the sender application (step 4) and retrieves the signaling message which was previously stored (step 5).

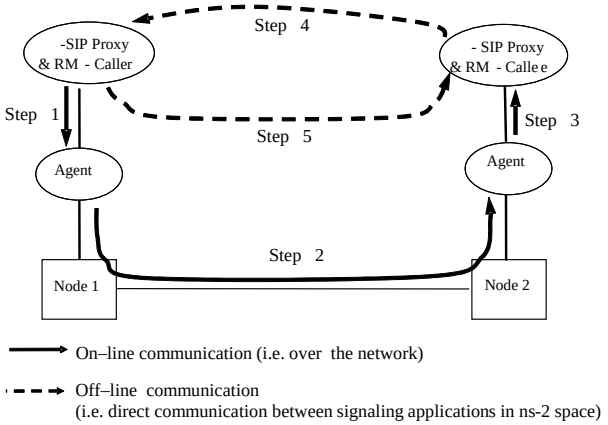

Fig. 6 The scheme of retrieving signaling messages when using TCP for communication between signaling applications

4. VALIDATING THE AGGREGATED TRAFFIC MODEL AGAINST POISSON

With the signaling modules, we can simulate the scenario presented in Fig. 2 to model the aggregated singaling traffic. The objective of these simulations is to demonstrate that the aggregated signaling traffic at the packet level behaves as Poisson traffic.

Our simulation model has two degrees of freedom, namely the choice of the number *N* of users and the choice of QoS parameters (IPLR/max. IPTD) values. We will prove that the modeling of the aggregated signaling traffic as a Poisson traffic does not depend on the values of the system parameters (*N* and IPLR/max. IPTD). For this purpose in section 4.1 we start with the concrete case of fixed *N* and IPLR/max. IPTD in order to generalize the results to other values of *N* (in section 4.2) and other values of IPLR/max. IPTD (in section 4.3).

4.1 The case of fixed *N* and fixed IPLR value

We present in detail the case when on average we have *N*=20 users in the system and IPLR equals 0.003488 (IPTD= 163.76 ms). We take these values following the guidelines from [11].

As proved in [1], traffic is Poisson if and only if the interarrival times of its packets follow an exponential distribution function and they are independent. So, we will simulate the aggregated signaling traffic and check these two conditions with statistical methods.

The measured parameter is the interarrival time between signaling packets on the link B (direction from the Ingress domain to the Egress domain, see Fig. 2) and the time of the simulations is 10 minutes, which is long enough to ensure at least 10^5 transferred packets on the link B (direction Ingress to Egress).

The histogram of interarrival packet times on the link B for N=20 users is shown in Fig. 7.

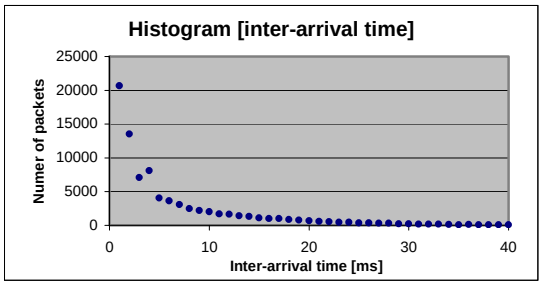

Fig. 7 Histogram of interarrival times on the link B

To compare the probability distribution function with the theoretical exponential function we use the Chi-square test with null hypothesis, which says "there are no differences between the theoretical exponential distribution and the distribution obtained by simulations". To avoid coincidences, we confirmed the test four times, taking for each test 40 samples (from 1 ms to 40 ms). In the tables of the Chi-square test, we must search the values for 39 degrees of freedom because we needed one degree of freedom to calculate the average. The values of χ^2_{exp} presented in Table 4-1 we compared with the value of the Chi-square tables ($\chi^2_{0.95,39}$=25.6954). So, based on the results reported in Table 4-1, there is no evidence to reject the null hypothesis because $\chi^2_{exp} < \chi^2_{0.95,39}$ and we may conclude that the curve follows the theoretical exponential model.

Table 4-1 Values of χ^2_{exp} from simulations for N=20 users and IPLR=0.003488

χ^2_{1exp}	$\chi^2_{2\,exp}$	$\chi^2_{3\,exp}$	$\chi^2_{4\,exp}$	$\chi^2_{0.95,39}$
7.4326	6.357	8.0356	7.1289	25.6954

The second condition for validating the packet arrivals to be Poissonian is the independence of the packet interarrival times. This independence is clear for packets of different setup procedures, as the arrivals of setup procedures are also independent; but we must demonstrate the independence of the interarrival times between packets of the same setup procedure. For this scope, we calculate the autocorrelation function of the interarrival times for all the packets (T) as follows:

$$R(k) = \frac{\sum_{i=2}^{T-k}(I_i - \bar{I})(I_{i+k} - \bar{I})}{\sum_{i=1}^{T}(I_i - \bar{I})^2}$$

where I_i is the interarrival time between packet i and its preceeding packet and I_{i+k} is the interarrival time of packet $i+k$ and its preceding packet. $R(k)$ depends on the lag k between packets. If the packets separated by the lag k are independent from each other then this autocorrelation function will be at least as low as the autocorrelation produced by an independent signal, i.e. white noise. We formulate the null hypothesis, which says "there is no significant correlation between packets of the same setup procedure". To ensure that we consider all the packets of the same setup procedure, we must know how many packets might be between two packets of the same setup procedure. Since there are, at the most, six packets of the same setup procedure together in the network (see message procedure in Fig. 1), and an average of N=20 users, we may find up to 20x6 packets between two packets of the same setup procedure. So, we must prove the low autocorrelation (as low as white noise) for all the lags until lag=120, to ensure that there is no correlation between packets of the same setup procedure.

Since the times between packets depend on the number of users in the system[1] and the number of users in the system alters with time, then the interarrival times depend on the moment when we took the samples. As we want values of autocorrelation independent of the simulations, we proceed as follows: for each lag, we take trials of 100 samples and compare the value of autocorrelation $R(k)$ of each trial with the value of autocorrelation produced by white noise: $R(k)_{white\ noise}$. Remember that $R(k)_{white\ noise}$ for any lag k is:

$$R(k)_{white\ noise} = 1.96/\sqrt{s} \quad \textit{where s is the number of samples}$$

Each time when $R(k) < R(k)_{white\ noise}$ we obtain a success. We repeat this process for 500 different trials and count how many successes m we obtain. Next, we calculate the probability of X≤m successes in a Binomial Distribution with a probability of success of p=0.95. This probability is:

$$\text{Prob}(X \le m) = \sum_{j=0}^{m} \text{Prob}(X = j) = \sum_{j=0}^{m} \binom{500}{j} 0.95^j (1-0.95)^{500-j}$$

We will accept the null hypothesis when Prob(X≤m)≥0.95.

Since for our case the worst results appear for the lag=23 and the lag=39 we only present here these results. There are 491 and 494 successes respectively. This supposes a Prob(X≤m) equal to 0.999945 and 0.999999 respectively as presented in Table 4-2.

Table 4-2 Values of the test of independence for N=20 users and IPLR=0.003488

lag k	R(k)$_{white\ noise}$	R(k)	M	m	p	Prob(X≤m)

[1] The more users are in the system, the more often packets are sent. When there is only one user in the system, the times between packets depend only on the transmission times, and the inter-arrival times are correlated.

23	0.196	0.160	500	491	0.95	0.999945
39	0.196	0.153	500	494	0.95	0.999999

Since Prob($X \leq m$)≥0.95, we accept the null hypothesis of independence of the interarrival times.

So, we may conclude that the aggregated signaling traffic at the packet level on the link B is Poisson traffic.

4.2 Varying the number *N* of users.

We repeat the simulations changing the arrival rate of the setup procedures. This will modify the average number *N* of users in the system. Note that the capacity of the link B changes in each simulation and it is appropriatly adjusted to the number of users *N*.

In Table 4-3, we show the values of χ^2_{exp} for different numbers of users *N*. The value of the tables at the 0.05 level is 25.6954. So, for $N \leq 10$ users, we must reject the null hypothesis: "there are no differences between the theoretical exponential distribution and the distribution obtained by the simulation" because $\chi^2_{exp} > \chi^2_{0.95,39}$. Therefore, we must say that the distribution of the interarrival times is not an exponential distribution when $N \leq 10$ users (at the 0.05 level).

Table 4-3 Values of χ^2 for different number of users *N*

N	1	5	10	15	20	25	30	50	$\chi^2_{0.95,39}$
χ^2_{exp}	91.0	46.4	26.0	14.1	7.69	4.47	3.18	2.24	25.6954

In Table 4-4, we present the results of the test of independence. We can observe that this test is even stricter in terms of *N*. Only for values of $N \geq 20$ users we may accept the assumption of independence of the packet interarrival times, inasmuch as Prob($X \leq m$)>0.95. Note that the number of lags, which we must check is 6x*N* lags. For each value of *N*, there is one of the lags that gives the smaller number of successes *m* ($R(k) < R(k)_{white\ noise}$). We present in Table 4-4 only this worst lag. The value of *m* successes refers to this lag (see Table 4-4).

Table 4-4 Values of the test of independence for different number of users *N*

N	1	5	10	15	20	25	30	50
m	287	405	471	482	491	496	500	500
Prob($X \leq m$)	0	5.3×10^{-29}	0.2316	0.944	0.999945	1	1	1

We can conclude that for few users in the system, we cannot consider the aggregated signaling traffic as Poisson traffic. This seems to be logical because the fewer users are in the system, the fewer packets are in the network, therefore the packets are more influenced by others of the *same* setup procedure.

4.3 Varying the value of IPLR and max. IPTD

To study other cases of IPLR and max. IPTD we use the analysis of variance (ANOVA [2]). We will demonstrate that the results of the Chi-square test and the test of independence are equivalent for any IPLR and max. IPTD, i.e. the traffic model does not depend on the values of IPLR and max. IPTD.

The analysis of variance (ANOVA [2]) uses the F-distribution as part of the test of statistical significance. We group together the samples in different groups and compare the variance inside the groups with the variance between groups. From the two variances we may calculate the F ratio (F_{calc}) which we compare with the tables of the F-distribution ($F_{level,groups-1/samples-groups}$). If $F_{calc} < F_{level,groups-1/samples-groups}$, then we may assert that the grouping of samples is random.

We will form the different groups as follows: For each value of IPLR (and max. IPTD, note that there is a biunivocal correspondence between IPLR and max. IPTD when provisioning the S-CoS, as explained in [11]), we perform the simulations as explained in the previous section and take 50 values of the Chi-square test. The 50 obtained results for the same IPLR form one group. Next we perform the same operations for another value of IPLR and the obtained results of the Chi-square test form another group. From the variances between groups and inside the groups, we calculate the F ratio and compare with the F-distribution's value (the F-ratio for 4 groups and 50 samples per group at the 0.95 level is $F_{95\%,3/46}=2.79$). If the results show that the grouping is random, then, we may conclude that the values of the Chi-square test are indepdendent of the values of IPLR.

We perform the same actions for the test of indepdendence. Remark that the ANOVA method by itself ensures independent values of autocorrelation because it uses many different simulations. Because of this, it is not necessary to compare the different autocorrelation values with the Binomial function as we performed it in Section 0.1.

In Table 4-5 we reported the data of each group in the same column. The first two rows show the values of IPLR and max. IPTD of each test group and the next ones show the obtained values of link capacities calculated for these values of IPLR and max. IPTD. The following rows present some results of the Chi-square test and the test of independence calculated from the performed simulations.

In the Table 4-5 we present only 3 out of 50 test results namely the first test (*Sim.1*), the second test (*Sim.2*) and the last one (*Sim.50*). The last three rows of the table refer to the ANOVA method. We may see the *Mean* of the samples, the *Variance* inside the groups and the comparison of the F ratio calculated from the simulations (F_{calc}) with the F ratio of the F-distribution ($F_{95\%,3/46}$).

Remark that, for all the cases, the average number of users equals 20.

Based on the F ratio showed in Table 4-5, we can conclude that the aggregated signaling traffic model does not depend on the values of IPLR and max. IPTD when we provision the capacity links with the method presented in [11] because $F_{calc}<F_{95\%,3/46}$.

Table 4-5 Results of the ANOVA method for the results of the Chi-square test and test of independence

		Chi square				*Test of independence*			
IPLR		0.000111	0.003488	0.012731	0.02629	0.000111	0.003488	0.012731	0.02629
IPTD [ms]		209.5	163.76	134.41	113.98	209.5	163.76	134.41	113.98
Link capacity [kbps]	*Link* $A_{I\to E}$	50.1	64.7	78.8	92.9	50.1	64.7	78.8	92.9
	Link $A_{E\to I}$	31.5	40.3	49.0	57.8	31.5	40.3	49.0	57.8
	Link $B_{I\to E}$	740	948	1154	1362	740	948	1154	1362
	Link $B_{E\to I}$	660	844	1028	1212	660	844	1028	1212
	Link $C_{I\to E}$	37.0	47.4	57.7	68.1	37.0	47.4	57.7	68.1
	Link $C_{E\to I}$	43.8	56.1	68.3	80.6	43.8	56.1	68.3	80.6
Sim.1		6.45	7.23	6.66	7.87	0.154	0.163	0.149	0.152
Sim.2		7.31	6.89	8.02	7.12	0.151	0.158	0.153	0.151
Sim.50		8.10	6.23	7.35	6.56	0.149	0.161	0.158	0.163
Mean		7.34	6.92	7.79	7.26	0.152	0.154	0.156	0.154
Variance		0.458	0.387	0.423	0.436	0.00012	0.00023	0.00021	0.00017
F ratio		$1.06=F_{calc}<F_{95\%,3/46}=2.79$				$0.43=F_{calc}<F_{95\%,3/46}=2.79$			

5. CONCLUSIONS

The studies presented in this paper aimed at validating the modeling assumption which proposed Poisson stream to characterize the aggregated signaling traffic at the packet level. The simulations were performed thanks to a specific module, which we developed under ns-2 platform and which took into account the traffic characteristics of the signaling procedures (mainly setup procedure) closely following the implementation from EuQoS system. Proving that Poisson stream can be, under certain conditions, appropriate model of the traffic generated by the setup procedures, is an important fact which can simplify calculations of the buffer size when provisioning the S-CoS proposed for the EuQoS system. The simulations revealed that these conditions are:

- the average number of users in the system is more or equal to 20.
- the link capacities are calculated by the provisioning method and the S-CoS is isolated (WFQ).
- the user population is „infinite".

Under these conditions, we can use well-known methods to dimension the buffer size for ensuring maximum values of IPLR; an example of these methods we can find in [3].

6. REFERENCES

[1] Queueing Theory – Donald Gross and Carl M. Harris ISBN 0-471-17083-6.

[2] Keith E. Muller, Bethel A. Fetterman, "Regression and ANOVA : An Integrated Approach Using SAS Software"

[3] Robert B. Cooper, "Introduction to the Queueing Theory". ISBN: 0-444-00379-7. London, 1981.

[4] ITU-T E.721."Network GoS parameters and target values for circuit-switched services in the evolving ISDN", 1999.

[5] A. Estepa, R. Estepa, and J. Vozmediano, "A New Approach for VoIP Traffic Characterization". IEEE Communications letters. October 2004.

[6] Christian Merkle, Andreas Kirstädter, Dominic Schupke, Eleni Palkopolou, "Service Oriented Traffic Models for Future Backbone Networks" In 8. ITG-Fachtagung Photonische Netze May 2007.

[7] Ilie, D. Erman, D. Popescu, A. “Transfer Rate Models for Gnutella Signaling Traffic”. ISBN: 0-7695-2522-9. 2006.

[8] O. Dugeon, D. Morris, E. Monteiro, W. Burakowski, M. Diaz, "End to End Quality of Service over Heterogeneous Networks (EuQoS)", In Proc. of the Network Control and Engineering for QoS, Security and Mobility, IFIP TC6 Conference, NetCon'05, Lannion, France, 14-18 November 2005.

[9] http://www.euqos.eu/

[10] J. Babiarz et all, RFC 4594: “Configuration Guidelines for DiffServ Service Classes”. August 2006.

[11] J. Mongay Batalla and R. Janowski, “Provisioning dedicated class of service for reliable transfer of signaling traffic”. ITC-20. June 2007.

[12] J. Rosenberget et al. Request For Comments 3261: “SIP: Session Initiation Protocol”. June 2002.

[13] http://www.isi.edu/nsnam/ns/

[14] Jose Brazio et all., “Analysis and Design of Advanced Multiservice Networks Supporting Mobility, Multimedia, and Internetworking”, COST Action 279 Final Report. ISBN 978-0-387-28172-8 ©2006 Springer The Netherlands, pp. 29-30

[15] R. Hancock, G. Karagiannis, J. Loughney, S. Van den Bosch, RFC 4080: “Next Steps in Signaling (NSIS): Framework”. June 2005