

## 5. Network Simulation

In this chapter we discuss methods and techniques to simulate the operations of computer network systems and network applications in the real-world environment. Simulation is one of the most widely used techniques in network design and management to predict the performance of a network system or network application before the network is physically built or the application is rolled out.

### 5.1 . Types of simulation

#### 5.1.1 . Systems, models, discrete-event simulation

A network system is a set of network elements [1], such as routers, switches, links, users, and applications working together to achieve some tasks. The scope of a simulation study may only be a system that is part of another system as in the case of subnetworks. The state of a network system is the set of relevant variables and parameters that describe the system at a certain time that comprise the scope of the study. For instance, if we are interested in the utilization of a link, we want to know only the number of bits transmitted via the link in a second and the total capacity of the link, rather than the amount of buffers available for the ports in the switches connected by the link.

Instead of building a physical model of a network, we build a mathematical model representing the behaviour and the logical and quantitative relations between network elements. By changing the relations between network elements, we can analyse the model without constructing the network physically, assuming that the model behaves similarly to the real system, i.e., it is a valid model. For instance, we can calculate the utilization of a link analytically, using the formula  $U = D/T$ , where  $D$  is the amount of data sent at a certain time and  $T$  is the capacity of the link in bits per second. This is a very simple model that is very rare in real world problems. Unfortunately, the majority of real world problems are too complex to answer questions using simple mathematical equations. In highly complex cases simulation technique is more appropriate.

Simulation models can be classified in many ways. The most common classifications are as follows:

- *Static and dynamic simulation models:* A static model characterizes a system independently of time. A dynamic model represents a system that changes over time.
- *Stochastic and deterministic models:* If a model represents a system that includes random elements, it is called a stochastic model. Otherwise it is deterministic. Queueing

systems, the underlying systems in network models, contain random components, such as arrival time of packets in a queue, service time of packet queues, output of a switch port, etc.

- *Discrete and continuous models:* A continuous model represents a system with state variables changing continuously over time. Examples are differential equations that define the relationships for the extent of change of some state variables according to the change of time. A discrete model characterizes a system where the state variables change instantaneously at discrete points in time. At these discrete points in time some event or events may occur, changing the state of the system. For instance, the arrival of a packet at a router at a certain time is an event that changes the state of the port buffer in the router.

In our discussion, we assume dynamic, stochastic, and discrete network models. We refer to these models as discrete-event simulation models.

Due to the complex nature of computer communications, network models tend to be complex as well. The development of special computer programs for a certain simulation problem is a possibility, but it may be very time consuming and inefficient. Recently, the application of simulation and modeling packages has become more customary, saving coding time and allowing the modeler to concentrate on the modeling problem in hand instead of the programming details. At first glance, the use of such network simulation and modeling packages, as COMNET, OPNET, etc., creates the risk that the modeler has to rely on modeling techniques and hidden procedures that may be proprietary and may not be available to the public. In the following sections we will discuss the simulation methodology on how to overcome the fear of this risk by using validation procedures to make sure that the real network system will perform the same way as it has been predicted by the simulation model.

## 5.2. The need for communications network modeling and simulation

### 5.2.1. Simulation versus emulation

In a world of more and more data, computers, storage systems, and networks, the design and management of systems are becoming an increasingly challenging task. As networks become faster, larger, and more complex, traditional static calculations are no longer reasonable approaches for validating the implementation of a new network design and multimillion dollar investments in new network technologies. Complex static calculations and spreadsheets are not appropriate tools any more due to the stochastic nature of network traffic and the complexity of the overall system.

Organizations depend more and more on new network technologies and network applications to support their critical business needs. As a result, poor network performance may have serious impacts on the successful operation of their businesses. In order to evaluate the various alternative solutions for a certain design goal, network designers increasingly rely on methods that help them evaluate several design proposals before the final decision is made and the actual systems is built. A widely accepted method is performance prediction through simulation. A simulation model can be used by a network designer to analyse de-

sign alternatives and study the behaviour of a new system or the modifications to an existing system without physically building it. A simulation model can also represent the network topology and tasks performed in a network in order to obtain statistical results about the network's performance.

It is important to understand the difference between simulation and emulation. The purpose of emulation is to mimic the original network and reproduce every event that happens in every network element and application. In simulation, the goal is to generate statistical results that represent the behaviour of certain network elements and their functions. In discrete event simulation, we want to observe events as they happen over time, and collect performance measures to draw conclusions on the performance of the network, such as link utilizations, response times, routers' buffer sizes, etc.

Simulation of large networks with many network elements can result in a large model that is difficult to analyse due to the large amount of statistics generated during simulation. Therefore, it is recommended to model only those parts of the network which are significant regarding the statistics we are going to obtain from the simulation. It is crucial to incorporate only those details that are significant for the objectives of the simulation. Network designers typically set the following objectives:

- *Performance modeling*: Obtain statistics for various performance parameters of links, routers, switches, buffers, response time, etc.
- *Failure analysis*: Analyse the impacts of network element failures.
- *Network design*: Compare statistics about alternative network designs to evaluate the requirements of alternative design proposals.
- *Network resource planning*: Measure the impact of changes on the network's performance, such as addition of new users, new applications, or new network elements.

Depending on the objectives, the same network might need different simulation models. For instance, if the modeler wants to determine the overhead of a new service or a protocol on the communication links, the model's links need to represent only the traffic generated by the new service. In another case, when the modeler wants to analyse the response time of an application under maximum offered traffic load, the model can ignore the traffic corresponding to the new service of the protocol analysed in the previous model.

Another important question is the granularity of the model, i.e., the level of details at which a network element is modeled. For instance, we need to decide whether we want to model the internal architecture of a router or we want to model an entire packet switched network. In the former case, we need to specify the internal components of a router, the number and speed of processors, types of buses, number of ports, amount of port buffers, and the interactions between the router's components. But if the objective is to analyse the application level end-to-end response time in the entire packet switched network, we would specify the types of applications and protocols, the topology of the network and link capacities, rather than the internal details of the routers. Although the low level operations of the routers affect the overall end-to-end response time, modeling the detailed operations do not significantly contribute to the simulation results when looking at an entire network. Modeling the details of the routers' internal operations in the order of magnitude of nanoseconds does not contribute significantly to the end-to-end delay analysis in the higher order of magnitude of microseconds or seconds. The additional accuracy gained from higher model granularity is far outweighed by the model's complexity and the time and effort required

by the inclusion of the routers' details.

Simplification can also be made by applying statistical functions. For instance, modeling cell errors in an ATM network does not have to be explicitly modeled by a communication link by changing a bit in the cell's header, generating a wrong CRC at the receiver. Rather, a statistical function can be used to decide when a cell has been damaged or lost. The details of a cell do not have to be specified in order to model cell errors.

These examples demonstrate that the goal of network simulation is to reproduce the functionality of a network pertinent to a certain analysis, not to emulate it.

### 5.3. Types of communications networks, modeling constructs

A communications network consists of network elements, nodes (senders and receivers) and connecting communications media. Among several criteria for classifying networks we use two: transmission technology and scale. The scale or distance also determines the technique used in a network: wireline or wireless. The connection of two or more networks is called **internetwork**. The most widely known internetwork is the Internet.

According to transmission technology we can broadly classify networks as broadcast and point-to-point networks:

- In *broadcast networks* a single communication channel is shared by every node. Nodes communicate by sending packets or frames received by all the other nodes. The address field of the frame specifies the recipient or recipients of the frame. Only the addressed recipient(s) will process the frame. Broadcast technologies also allow the addressing of a frame to all nodes by dedicating it as a broadcast frame processed by every node in the network. It is also possible to address a frame to be sent to all or any members of only a group of nodes. The operations are called multicasting and any casting respectively.
- *Point-to-point networks* consist of many connections between pairs of nodes. A packet or frame sent from a source to a destination may have to first traverse intermediate nodes where they are stored and forwarded until it reaches the final destination.

Regarding our other classification criterion, the scale of the network, we can classify networks by their physical area coverage:

- *Personal area networks (PANs)* support a person's needs. For instance, a wireless network of a keyboard, a mouse, and a *personal digital assistant (PDA)* can be considered as a PAN.
- *Local area networks (LANs)*, typically owned by a person, department, a smaller organization at home, on a single floor or in a building, cover a limited geographic area. LANs connect workstations, servers, and shared resources. LANs can be further classified based on the transmission technology, speed measured in bits per second, and topology. Transmission technologies range from traditional 10Mbps LANs to today's 10Gbps LANs. In terms of topology, there are bus and ring networks and switched LANs.
- *Metropolitan area networks (MANs)* span a larger area, such as a city or a suburb. A widely deployed MAN is the cable television network distributing not just one-way TV programs but two-way Internet services as well in the unused portion of the transmission spectrum. Other MAN technologies are the *Fiber Distributed Data Interface (FDDI)*

and IEEE wireless technologies as discussed below.

- *Wide area networks (WANs)* cover a large geographical area, a state, a country or even a continent. A WAN consists of hosts (clients and servers) connected by subnets owned by communications service providers. The subnets deliver messages from the source host to the destination host. A subnet may contain several transmission lines, each one connecting a pair of specialized hardware devices called routers. Transmission lines are made of various media; copper wire, optical fiber, wireless links, etc. When a message is to be sent to a destination host or hosts, the sending host divides the message into smaller chunks, called **packets**. When a packet arrives on an incoming transmission line, the router stores the packet before it selects an outgoing line and forwards the packet via that line. The selection of the outgoing line is based on a routing algorithm. The packets are delivered to the destination host(s) one-by-one where the packets are reassembled into the original message.

Wireless networks can be categorized as short-range radio networks, wireless LANs, and wireless WANs.

- In short range radio networks, for instance Bluetooth, various components, digital cameras, **Global Positioning System (GPS)** devices, headsets, computers, scanners, monitors, and keyboards are connected via short-range radio connections within 20- 30 feet. The components are in primary-secondary relation. The main system unit, the primary component, controls the operations of the secondary components. The primary component determines what addresses the secondary devices use, when and on what frequencies they can transmit.
- A wireless LAN consists of computers and access points equipped with a radio modem and an antenna for sending and receiving. Computers communicate with each other directly in a peer-to-peer configuration or via the access point that connects the computers to other networks. Typical coverage area is around 300 feet. The wireless LAN protocols are specified under the family of IEEE 802.11 standards for a range of speed from 11 Mbps to 108 Mbps.
- Wireless WANs comprise of low bandwidth and high bandwidth networks. The low bandwidth radio networks used for cellular telephones have evolved through three generations. The first generation was designed only for voice communications utilizing analog signaling. The second generation also transmitted only voice but based on digital transmission technology. The current third generation is digital and transmits both voice and data at most 2Mbps. Fourth and further generation cellular systems are under development. High-bandwidth WANs provides high-speed access from homes and businesses bypassing the telephone systems. The emerging IEEE 802.16 standard delivers services to buildings, not mobile stations, as the IEEE 802.11 standards, and operates in much higher 10-66 GHz frequency range. The distance between buildings can be several miles.
- Wired or wireless home networking is getting more and more popular connecting various devices together that can be accessible via the Internet. Home networks may consists of PCs, laptops, PDAs, TVs, DVDs, camcorders, MP3 players, microwaves, refrigerator, A/C, lights, alarms, utility meters, etc. Many homes are already equipped with high-speed Internet access (cable modem, DSL, etc.) through which people can download music and movies on demand.

The various components and types of communications networks correspond to the modeling constructs and the different steps of building a simulation model. Typically, a network topology is built first, followed by adding traffic sources, destinations, workload, and setting the parameters for network operation. The simulation control parameters determine the experiment and the running of the simulation. Prior to starting a simulation various statistics reports can be activated for analysis during or after the simulation. Statistical distributions are available to represent specific parameterizations of built-in analytic distributions. As the model is developed, the modeler creates new model libraries that can be reused in other models as well.

## 5.4. Performance targets for simulation purposes

In this section we discuss a non-exhausting list of *network attributes* that have a profound effect on the perceived network performance and are usual targets of network modeling. These attributes are the goals of the statistical analysis, design, and optimization of computer networks. Fundamentally, network models are constructed by defining the statistical distribution of the arrival and service rate in a queueing system that subsequently determines these attributes.

- *Link capacity*  
**Channel or link capacity** [3,4] is the number of messages per unit time handled by a link. It is usually measured in bits per second. One of the most famous of all results of information theory is Shannon's channel coding theorem: „For a given channel there exists a code that will permit the error-free transmission across the channel at a rate  $R$ , provided  $R \leq C$ , where  $C$  is the channel capacity.” Equality is achieved only when the Signal-to-noise Ratio (SNR) is infinite. See more details on Information and Coding Theory in [5].
- *Bandwidth*  
**Bandwidth** is the difference between the highest and lowest frequencies available for network signals. Bandwidth is also a loose term used to describe the throughput capacity of a specific link or protocol measured in Kilobits, Megabits, Gigabits, Terabits, etc., in a second.
- *Response time*  
The **response time** is the time it takes a network system to react to a certain source's input. The response time includes the transmission time to the destination, the processing time at both the source and destination and at the intermediate network elements along the path, and the transmission time back to the source. Average response time is an important measure of network performance. For users, the lower the response time the better. Response time statistics (mean and variation) should be stationary; it should not depend on the time of the day. Note that low average response time does not guarantee that there are no extremely long response times due to network congestions.
- *Latency*  
**Delay or latency** is the amount of time it takes for a unit of data to be transmitted across a network link. Latency and bandwidth are the two factors that determine the speed

of a link. It includes the propagation delay (the time taken for the electrical or optical signals to travel the distance between two points) and processing time. For instance, the latency, or round-time delay between a ground station of a satellite communication link and back to another ground station (over 34,000 km each way) is approximately 270 milliseconds. The round-time delay between the east and west coast of the US is around 100 ms, and transglobal is about 125 ms. The end-to-end delay of a data path between source and destination spanning multiple segments is affected not only by the media's signal speed, but also by the network devices, routers, switches along the route that buffer, process, route, switch, and encapsulate the data payload. Erroneous packets and cells, signal loss, accidental device and link failures and overloads can also contribute to the overall network delay. Bad cells and packets force retransmission from the initial source. These packets are typically dropped with the expectation of a later retransmission resulting in slowdowns that cause packets to overflow buffers.

- *Routing protocols*

The route is the path that network traffic takes from the source to the destination. The path in a LAN is not a critical issue because there is only one path from any source to any destination. When the network connects several enterprises and consists of several paths, routers, and links, finding the best route or routes becomes critical. A route may traverse through multiple links with different capacities, latencies, and reliabilities. Routes are established by routing protocols. The objective of the routing protocols is to find an optimal or near optimal route between source and destination avoiding congestions.

- *Traffic engineering*

A new breed of routing techniques is being developed using the concept of traffic engineering. Traffic engineering implies the use of mechanisms to avoid congestion by allocating network resources optimally, rather than continually increasing network capacities. Traffic engineering is accomplished by mapping traffic flows to the physical network topology along predetermined paths. The optimal allocation of the forwarding capacities of routers and switches are the main target of traffic engineering. It provides the ability to diverge traffic flows away from the optimal path calculated by the traditional routing protocols into a less congested area of the network. The purpose of traffic engineering is to balance the offered load on the links, routers, and switches in a way that none of these network elements is over or under utilized.

- *Protocol overhead*

Protocol messages and application data are embedded inside the protocol data units, such as frames, packets, and cells. A main interest of network designers is the overhead of protocols. Protocol overhead concerns the question: How fast can we really transmit using a given communication path and protocol stack, i.e., how much bandwidth is left for applications? Most protocols also introduce additional overhead associated with in-band protocol management functions. Keep-alive packets, network alerts, control and monitoring messages, poll, select, and various signaling messages are transmitted along with the data streams.

- *Burstiness*

The most dangerous cause of network congestion is the ***burstiness*** of the network traffic. Recent results make evident that high-speed Internet traffic is more bursty and its variability cannot be predicted as assumed previously. It has been shown that network

traffic has similar statistical properties on many time scales. Traffic that is bursty on many or all time scales can be described statistically using the notion of *long-range dependency* [6, 7, and 8]. Long-range dependent traffic has observable bursts on all time scales. One of the consequences is that combining the various flows of data, as it happens in the Internet, does not result in the smoothing of traffic. Measurements of local and wide area network traffic have proven that the widely used Markovian process models cannot be applied for today's network traffic. If the traffic were Markovian process, the traffic's burst length would be smoothed by averaging over a long time scale, contradicting the observations of today's traffic characteristics. The harmful consequences of bursty traffic will be analysed in a case study in Section 5.9.

- *Frame size*  
Network designers are usually worried about large frames because they can fill up routers' buffers much faster than smaller frames resulting in lost frames and retransmissions. Although the processing delay for larger frames is the same as for smaller ones, i.e., larger packets are seemingly more efficient, routers and switches can process internal queues with smaller packets faster. Larger frames are also target for fragmentation by dividing them into smaller units to fit in the **Maximum Transmission Unit (MTU)**. MTU is a parameter that determines the largest datagram than can be transmitted by an IP interface. On the other hand, smaller frames may create more collision in an Ethernet network or have lower utilization on a WAN link.
- *Dropped packet rate*  
Packets may be dropped by the data link and network layers of the OSI architecture. The transport layer maintains buffers for unacknowledged packets and retransmits them to establish an error-free connection between sender and receiver. The rate of dropping packets at the lower layers determines the rate of retransmitting packets at the transport layer. Routers and switches may also drop packets due to the lack of internal buffers. Buffers fill up quicker when WAN links get congested which causes timeouts and retransmissions at the transport layer. The TCP's *slow start algorithm* tries to avoid congestions by continually estimating the round-trip propagation time and adjusting the transmission rate according to the measured variations in the roundtrip time [11].

## 5.5. Traffic characterization

Communications networks transmit data with random properties. Measurements of network attributes are statistical samples taken from random processes, for instance, response time, link utilization, interarrival time of messages, etc. In this section we review basic statistics that are important in network modeling and performance prediction. After a family of statistical distributions has been selected that corresponds to a network attribute under analysis, the next step is to estimate the parameters of the distribution. In many cases the sample average or mean and the sample variance are used to estimate the parameters of a hypothesized distribution. Advanced software tools include the computations for these estimates. The mean is interpreted as the most likely value about which the samples cluster. The following equations can be used when discrete or continuous raw data are available. Let  $X_1, X_2, \dots, X_n$  are samples of size  $n$ . The mean of the sample is defined by



distribution	parameter(s)	estimator(s)
Poisson	$\alpha$	$\hat{\alpha} = \bar{X}$
exponential	$\lambda$	$\hat{\lambda} = 1/\bar{X}$
uniform	$b$	$\hat{b} = ((n + 1)/n)[\max(X)]$ (unbiased)
normal	$\mu, \sigma^2$	$\hat{\mu} = \bar{X}$ $\hat{\sigma}^2 = S^2$ (unbiased)

**Figure 5.1.** Estimation of the parameters of the most common distributions.

$$\bar{X} = \frac{\sum_{i=1}^n X_i}{n}.$$

The sample variance  $S^2$  is defined by

$$S^2 = \frac{\sum_{i=1}^n X_i^2 - n\bar{X}^2}{n - 1}.$$

If the data are discrete and grouped in a frequency distribution, the equations above are modified as

$$\bar{X} = \frac{\sum_{j=1}^k f_j X_j}{n},$$

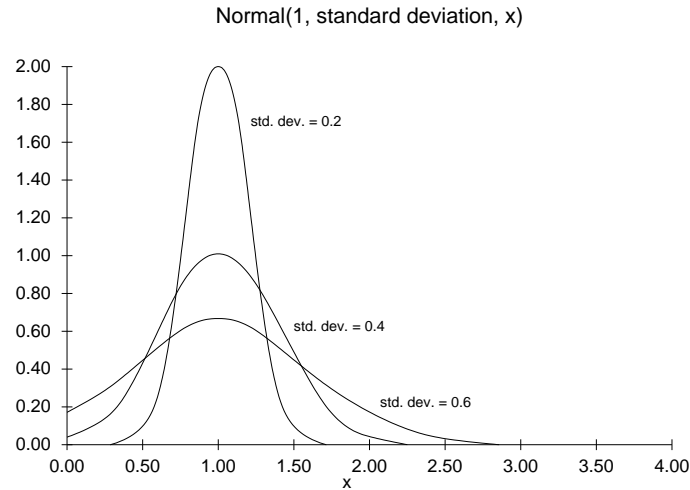
$$S^2 = \frac{\sum_{j=1}^k f_j X_j^2 - n\bar{X}^2}{n - 1},$$

where  $k$  is the number of different values of  $X$  and  $f_j$  is the frequency of the value  $X_j$  of  $X$ . The standard deviation  $S$  is the square root of the variance  $S^2$ .

The variance and standard deviation show the deviation of the samples around the mean value. Small deviation from the mean demonstrates a strong central tendency of the samples. Large deviation reveals little central tendency and shows large statistical randomness.

Numerical estimates of the distribution parameters are required to reduce the family of distributions to a single distribution and test the corresponding hypothesis. Table 5.1 from [12] describes estimators for the most common distributions occurring in network modeling. If  $\alpha$  denotes a parameter, the estimator is denoted by  $\hat{\alpha}$ . Except for an adjustment to remove bias in the estimates of  $\sigma^2$  for the normal distribution and in the estimate of  $b$  of the uniform distribution, these estimators are the maximum likelihood estimators based on the sample data.

Probability distributions describe the random variations that occur in the real world [13, 14]. Although we call the variations random, randomness has different degrees; the different distributions correspond to how the variations occur. Therefore, different distributions are used for different simulation purposes. Probability distributions are represented by probability density functions. Probability density functions show how likely a certain value is. Cumulative density functions give the probability of selecting a number at or below a certain value. For example, if the cumulative density function value at 1 was equal to 0.85, then 85% of the time, selecting from this distribution would give a number less than 1. The value



**Figure 5.2.** An example Normal distribution

of a cumulative density function at a point is the area under the corresponding probability density curve to the left of that value. Since the total area under the probability density function curve is equal to one, cumulative density functions converge to one as we move toward the positive direction. In most of the modeling cases, the modeler does not need to know all details to build a simulation model successfully. He or she has only to know which distribution is the most appropriate one for the case.

Below, we summarize the most common statistical distributions based on [12, 14, and 15]. We use the simulation modeling tool COMNET [14] to depict the respective probability density functions (PDF). From the practical point of view, a PDF can be approximated by a histogram with all the frequencies of occurrences converted into probabilities.

- *Normal distribution*

It typically models the distribution of a compound process that can be described as the sum of a number of component processes. For instance, the time to transfer a file (response time) sent over the network is the sum of times required to send the individual blocks making up the file. In modeling tools the Normal distribution function takes two positive, real numbers: mean and standard deviation. It returns a positive, real number. The stream parameter  $x$  specifies which random number stream will be used to provide the sample. It is also often used to model message sizes. For example, a message could be described with mean size of 20,000 bytes and a standard deviation of 5,000 bytes.

- *Poisson distribution*

It models the number of independent events occurring in a certain time interval; for instance, the number of packets of a packet flow received in a second or a minute by a destination. In modeling tools, the Poisson distribution function takes one positive, real number, the mean. The „number” parameter in Figure 5.3 specifies which random number stream will be used to provide the sample. This distribution, when provided with a time interval, returns an integer which is often used to represent the number of arrivals likely to occur in that time interval. Note that in simulation, it is more useful

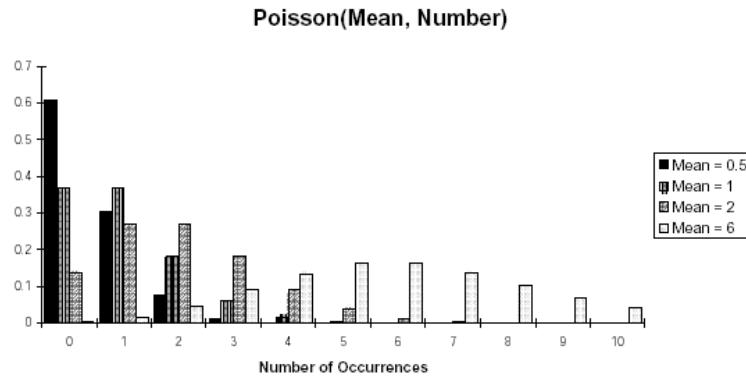


Figure 5.3. An example Poisson distribution

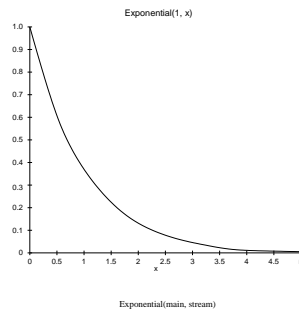


Figure 5.4. An example Exponential distribution

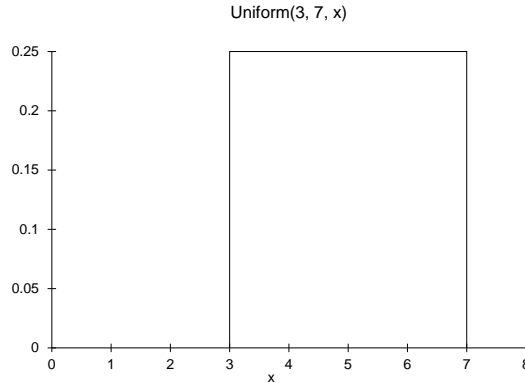
to have this information expressed as the time interval between successive arrivals. For this purpose, the Exponential distribution is used.

- *Exponential distribution*

It models the time between independent events, such as the interarrival time between packets sent by the source of a packet flow. Note, that the number of events is Poisson, if the time between events is exponentially distributed. In modeling tools, the Exponential distribution function takes one positive, real number, the mean and the stream parameter  $x$  that specifies which random number stream will be used to provide the sample. Other application areas include: Time between data base transactions, time between keystrokes, file access, emails, name lookup request, HTTP lookup, X-window protocol exchange, etc.

- *Uniform distribution*

Uniform distribution models data that range over an interval of values, each of which is equally likely. The distribution is completely determined by the smallest possible value min and the largest possible value max. For discrete data, there is a related discrete uniform distribution as well. Packet lengths are often modeled by uniform distribution. In modeling tools the Uniform distribution function takes three positive, real numbers:



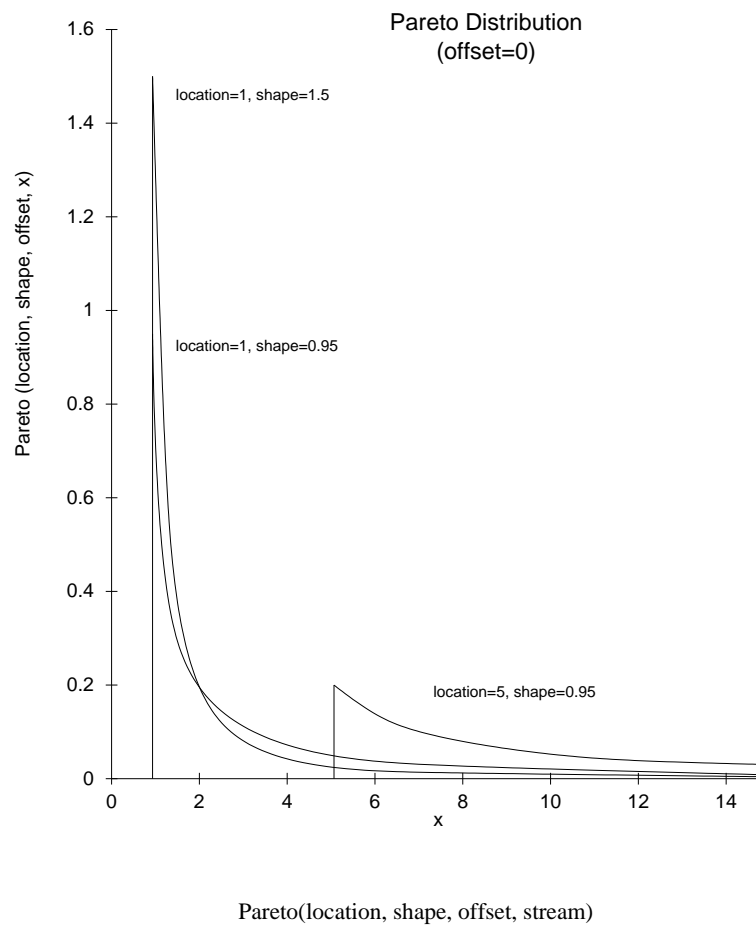
**Figure 5.5.** An example Uniform distribution

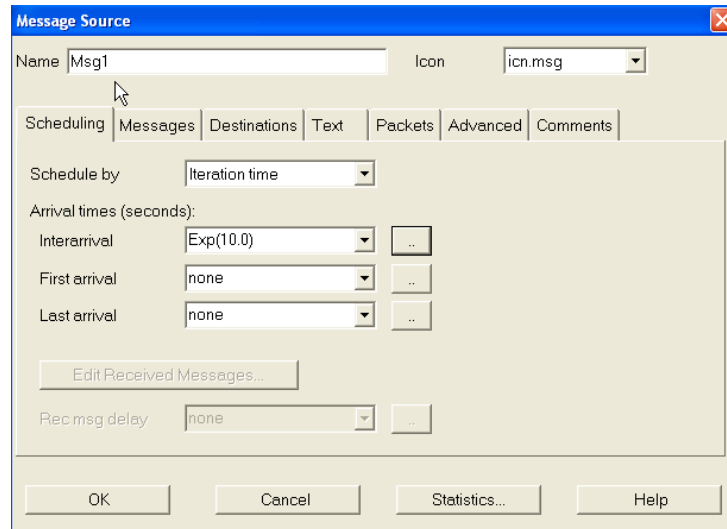
min, max, and stream. The stream parameter  $x$  specifies which random number stream will be used to provide the sample.

- *Pareto distribution*

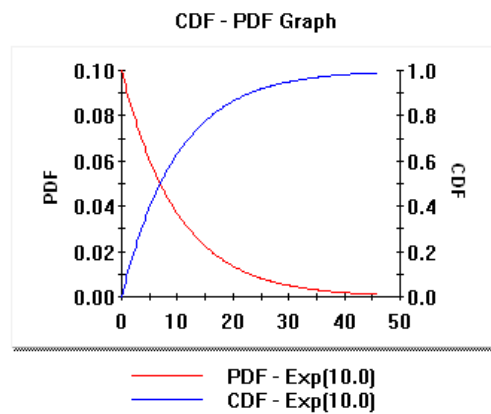
The Pareto distribution is a power-law type distribution for modeling bursty sources (not long-range dependent traffic). The distribution is heavily peaked but the tail falls off slowly. It takes three parameters: location, shape, and offset. The location specifies where the distribution starts, the shape specifies how quickly the tail falls off, and the offset shifts the distribution.

A common use of probability distribution functions is to define various network parameters. A typical network parameter for modeling purposes is the time between successive instances of messages when multiple messages are created. The specified time is from the start of one message to the start of the next message. As it is discussed above, the most frequent distribution to use for interarrival times is the exponential distribution. The parameters entered for the exponential distribution are the mean value and the random stream number to use. Network traffic is often described as a Poisson process. This generally means that the number of messages in successive time intervals has been observed and the distribution of the number of observations in an interval is Poisson distributed. In modeling tools, the number of messages per unit of time is not entered. Rather, the interarrival time between messages is required. It may be proven that if the number of messages per unit time interval is Poisson-distributed, then the interarrival time between successive messages is exponentially distributed. The interarrival distribution in the following dialog box for a message source in COMNET is defined by Exp (10.0). It means that the time from the start of one message to the start of the next message follows an exponential distribution with 10 seconds on the average. Figure 5.8 shows the corresponding probability density function.

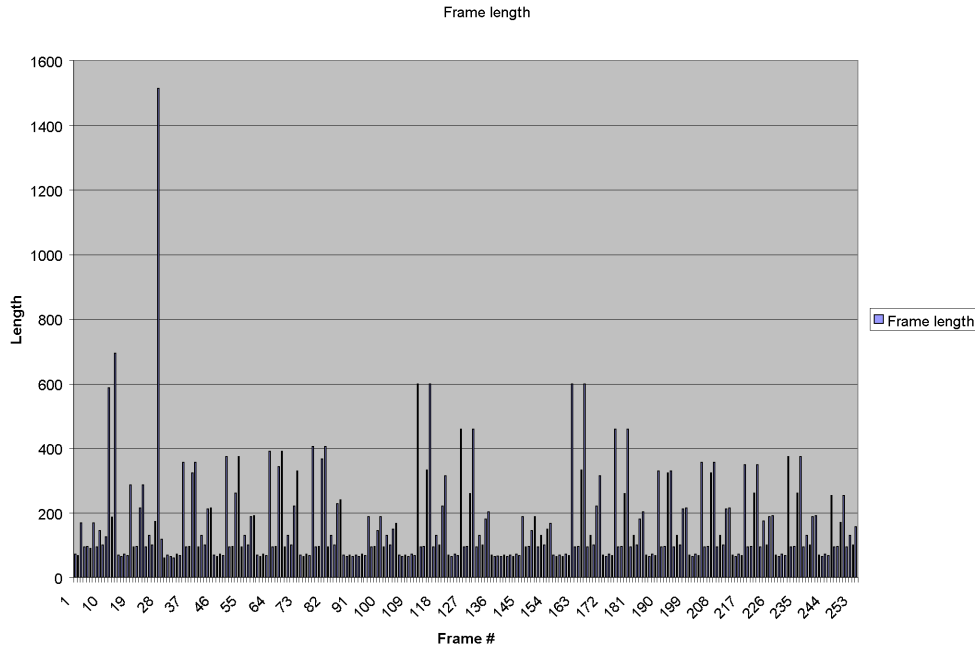
**Figure 5.6.** An example Pareto distribution



**Figure 5.7.** Exponential distribution of interarrival time with 10 sec on the average



**Figure 5.8.** Probability density function of the Exp (10.0) interarrival time



**Figure 5.9.** Visualization of anomalies in packet lengths

Many simulation models focus on the simulation of various traffic flows. Traffic flows can be simulated by either specifying the traffic characteristics as input to the model or by importing actual traffic traces that were captured during certain application transactions under study. The latter will be discussed in a subsequent section on Baselineing.

Network modelers usually start the modeling process by first analyzing the captured traffic traces to visualize network attributes. It helps the modeler understand the application level processes deep enough to map the corresponding network events to modeling constructs. Common tools can be used before building the model. After the preliminary analysis, the modeler may disregard processes, events that are not important for the study in question. For instance, the capture of traffic traces of a database transaction reveals a large variation in frame lengths. Figure 5.9 helps visualize the anomalies:

The analysis of the same trace (Figure 5.10) also discloses a large deviation of the interarrival times of the same frames (delta times):

Approximating the cumulative probability distribution function by a histogram of the frame lengths of the captured traffic trace (Figure 5.11) helps the modeler determine the family of the distribution:

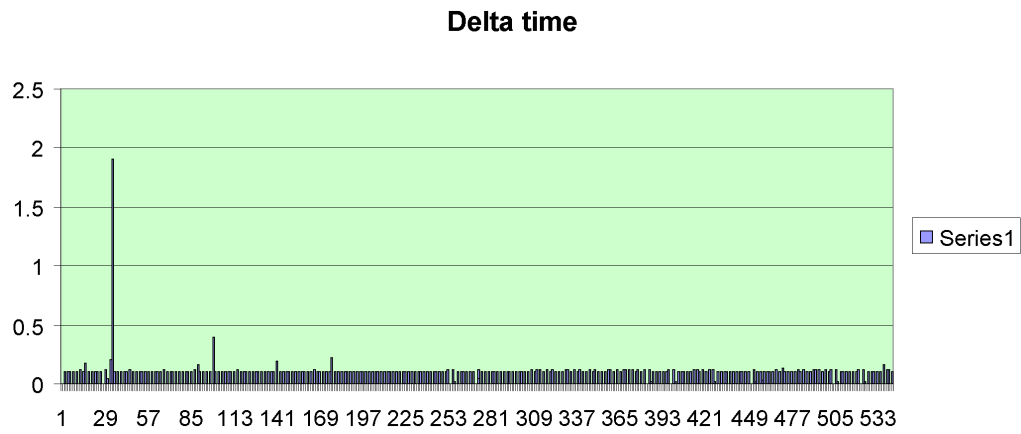


Figure 5.10. Large deviations between delta times

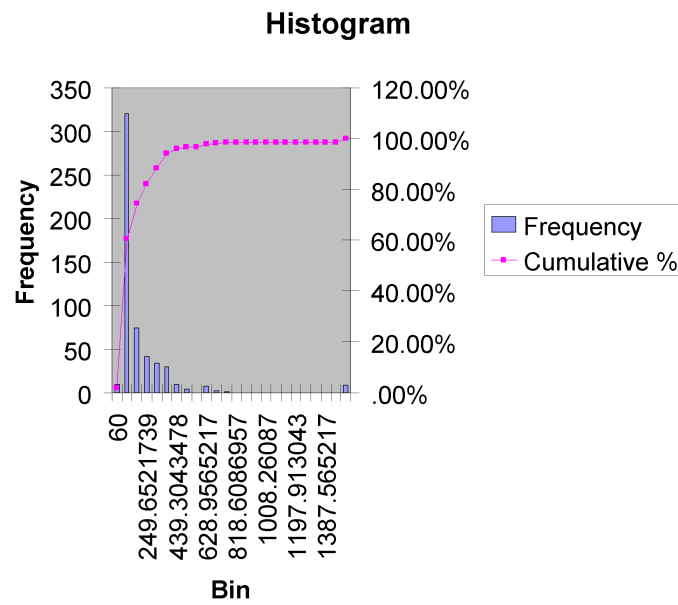


Figure 5.11. Histogram of frame lengths



## 5.6. Simulation modeling systems

### 5.6.1. Data collection tools and network analysers

The section summarizes the main features of the widely used discrete event simulation tools, OPNET and COMNET, and the supporting network analysers, Network Associates' Sniffer and OPNET's Application Characterization Environment.

*OPNET (OPTimized Network Engineering Tools)* is a comprehensive simulation system capable of modeling communication networks and distributed systems with detailed protocol modeling and performance analysis. (A free download version is available for academic purposes from [17].) OPNET consists of a number of tools that fall into three categories corresponding to the three main phases of modeling and simulation projects [17]: Model Specification, Data Collection and Simulation, and Analysis.

### 5.6.2. Model specification

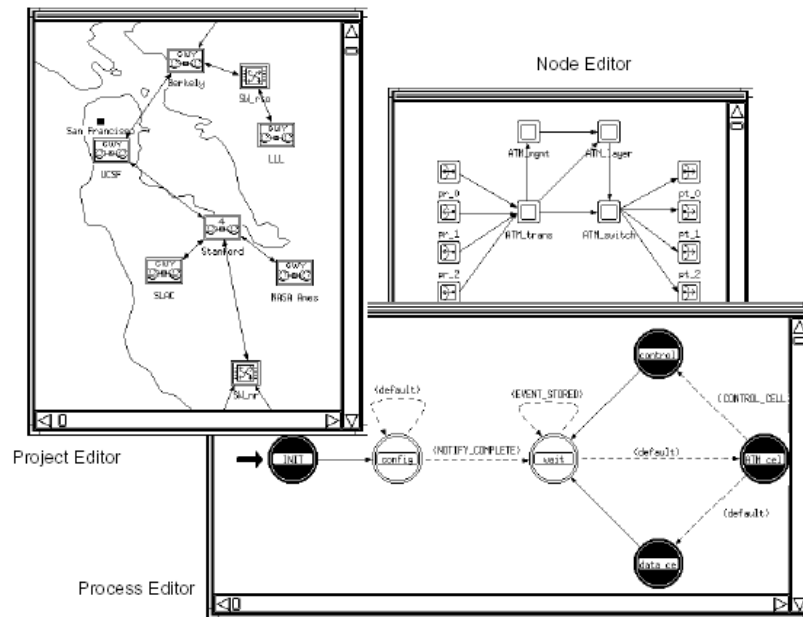
During model specification the network modeler develops a representation of the network system under study. OPNET implements the concept of model reuse, i.e., models are based on embedded models developed earlier and stored in model libraries. The model is specified at various levels of details using specification editors. These editors categorize the required modeling information corresponding to the hierarchical structure of an actual network system. The highest level editor, the *Project Editor* develops network models consisting of network topology, subnets, links, and node models specified in the *Node Editor*. The Node Editor describes nodes' internal architecture, functional elements and data flow between them. Node models in turn, consist of modules with process models specified by the *Process Editor*. The lowest level of the network hierarchy, the process models, describes the module's behaviour in terms of protocols, algorithms, and applications using finite state machines and a high-level language.

There are several other editors to define various data models referenced by process- or node-level models, e.g., packet formats and control information between processes. Additional editors create, edit, and view probability density functions (PDFs) to control certain events, such as the interarrival time of sending or receiving packets, etc. The model-specification editors provide a graphical interface for the user to manipulate objects representing the models and the corresponding processes. Each editor can specify objects and operations corresponding to the model's abstraction level. Therefore, the Project Editor specifies nodes and link objects of a network, the Node Editor specifies processors, queues, transmitters, and receivers in the network nodes, and the Process Editor specifies the states and transitions in the processes. Figure 5.12 depicts the abstraction level of each editor:

### 5.6.3. Data collection and simulation

OPNET can produce many types of output during simulation depending on how the modeler defined the types of output. In most cases, modelers use the built in types of data: output vectors, output scalars, and animation:

- Output vectors represent time-series simulation data consisting of list of entries, each of which is a time-value pair. The first value in the entries can be considered as the



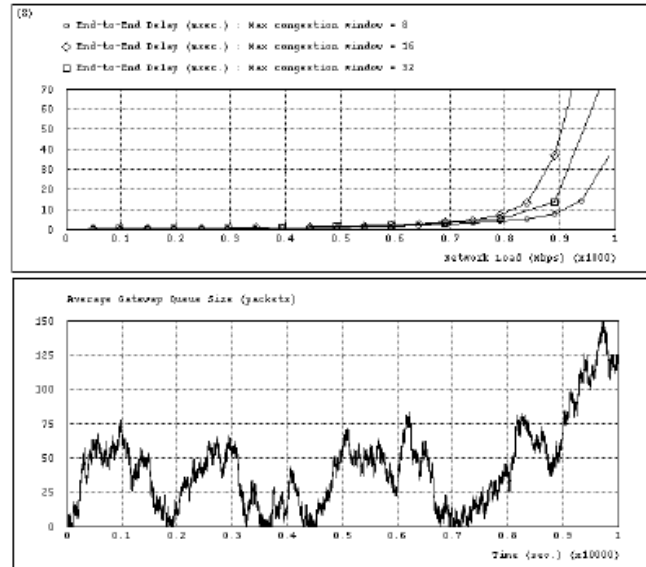
**Figure 5.12.** The three modeling abstraction levels specified by the Project, Node, and Process editors

independent variable and the second as the dependent variable.

- Scalar statistics are individual values derived from statistics collected during simulation, e.g., average transmission rate, peak number of dropped cells, mean response time, or other statistics.
- OPNET can also generate animations that are viewed during simulation or replay after simulation. The modeler can define several forms of animations, for instance, packet flows, state transitions, and statistics.

#### 5.6.4. Analysis

Typically, much of the data collected during simulations is stored in output scalar and output vector files. In order to analyse these data OPNET provides the *Analysis Tool* which is a collection of graphing and numerical processing functions. The Analysis Tool presents data in the form of graphs or traces. Each trace consists of a list of abscissa  $X$  and ordinate  $Y$  pairs. Traces are held and displayed in analysis panels. The Analysis Tool supports a variety of methods for processing simulation output data and computing new traces. Calculations, such as histograms, PDF, CDF, and confidence intervals are included. Analysis Tool also supports the use of mathematical filters to process vector or trace data. Mathematical filters are defined as hierarchical block diagrams based on a predefined set of calculus, statistical, and arithmetic operators. The example diagrams below (Figures 5.13 and 5.14) shows



**Figure 5.13.** Example for graphical representation of scalar data (upper graph) and vector data (lower graph)

graphs generated by the Analysis Tool:

Figure 5.14 Analysis Tool Showing Four Graphs.

COMNET is another popular discrete-event simulation system. We will discuss it briefly and demonstrate its features in Section 5.9.

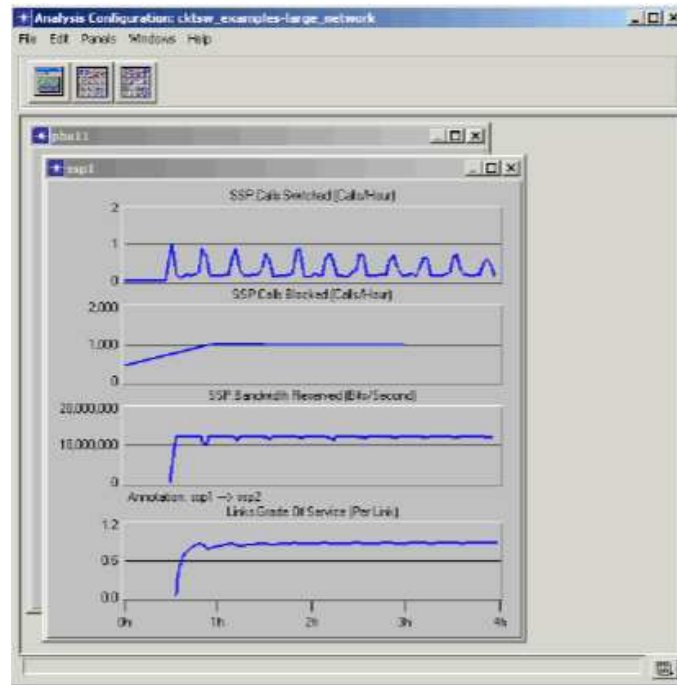


Figure 5.14. Figure ?? shows four graphs represented by the Analysis Tool:

### 5.6.5. Network Analysers – Application Characterization Environment (ACE)

There is an increasing interest in predicting, measuring, modeling, and diagnosing application performance across the application lifecycle from development through deployment to production. Characterizing the application's performance is extremely important in critical application areas, like in eCommerce. In the increasingly competitive eCommerce, the application's performance is critical, especially where the competition is just "one click" away. Application performance affects revenue. When an application performs poorly it is always the network that is blamed rather than the application. These performance problems may result from several areas including application design or slow database servers. Using tools, like ACE and Network Associates' Sniffer, network modelers can develop methodologies to identify the source of application slowdowns and resolve their causes. After analyzing the applications, modelers can make recommendations for performance optimization. The result is faster applications and better response times. The Application Characterization Environment (ACE) is a tool for visualizing, analyzing, and troubleshooting network applications. Network managers and application developers can use ACE to

- Locate network and application bottlenecks.
- Diagnose network and application problems.
- Analyse the affect of anticipated network changes on the response time of existing applications.

- Predict application performance under varying configurations and network conditions.

The performance of an application is determined by network attributes that are affected by the various components of a communication network. The following list contains some example for these attributes and the related network elements:

- *Network media*
  - Bandwidth (Congestion, Burstiness)
  - Latency (TCP window size, High latency devices, Chatty applications)
- *Nodes*
- *Clients*
  - User time
  - Processing time
  - Starved for data
- *Servers*
  - Processing time
  - Multi-tier waiting data
  - Starved for data
- *Application*
  - Application turns (Too many turns – Chatty applications)
  - Threading (Single vs. multi-threaded)
  - Data profile (Bursty, Too much data processing)

Analysis of an application requires two phases:

- Capture packet traces while an application is running to build a baseline for modeling an application. We can use the ACE's capturing tool or any other network analysers to capture packet traces. The packet traces can be captured by strategically deployed capture agents.
- Import the capture file to create a representation of the application's transactions called an application task for further analysis of the messages and protocol data units generated by the application.

After creating the application task, we can perform the following operations over the captured traffic traces:

- View and edit the captured packet traces on different levels of the network protocol stack in different windows. We can also use these windows to remove or delete sections of an application task. In this way, we focus on transactions of our interest.
- Perform application level analysis by identifying and diagnosing bottlenecks. We can measure the components of the total response time in terms of application level time, processing time, and network time and view detailed statistics on the network and application. We can also decode and analyse the network and application protocol data units from the contents of the packet traces.

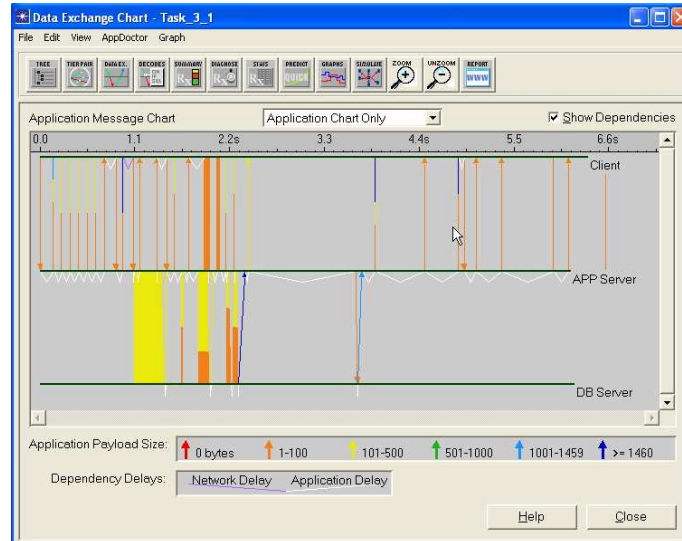


Figure 5.15. Data Exchange Chart.

- Predict application performance in „what-if” scenarios and for testing projected changes.

Without going into specific details we illustrate some of the features above through a simple three-tier application. We want to determine the reason or reasons of the slow response time from a Client that remotely accesses an Application Server (App Server) to retrieve information from a Database Server (DB Server). The connection is over an ADSL line between the client and the Internet, and a 100Mbps Ethernet connection between the App Server and the DB Server. We want to identify the cause of the slow response time and recommend solutions. We deployed capture agents at the network segments between the client and the App Server and between the servers. The agents captured traffic traces simultaneously during a transaction between the client and the App Server and the App Server and the DB Server respectively. Then, the traces were merged and synchronized to obtain the best possible analysis of delays at each tier and in the network.

After importing the trace into ACE, we can analyse the transaction in the *Data Exchange Chart*, which depicts the flow of application messages among tiers over time.

The Data Exchange Chart shows packets of various sizes being transmitted between the Client and the servers. The overall transaction response time is approximately 6 seconds. When the „Show Dependencies” checkbox is checked, the white dependency lines indicate large processing delays on the Application Server and Client tiers. For further analysis, we generate the „Summary of Delays” window showing how the total response time of the application is divided into four general categories: Application delay, Propagation delay, Transmission delay and Protocol/Congestion delay. Based on this chart we can see the relation between application and network related delays during the transaction between the client and the servers. The chart clearly shows that the application delay far outweighs the

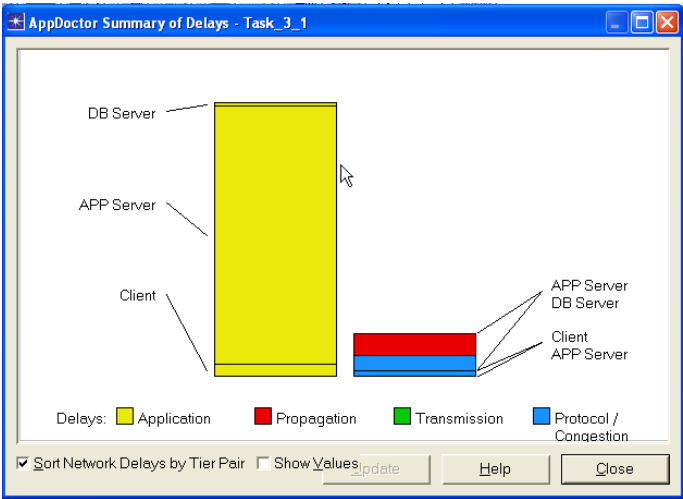


Figure 5.16. Summary of Delays

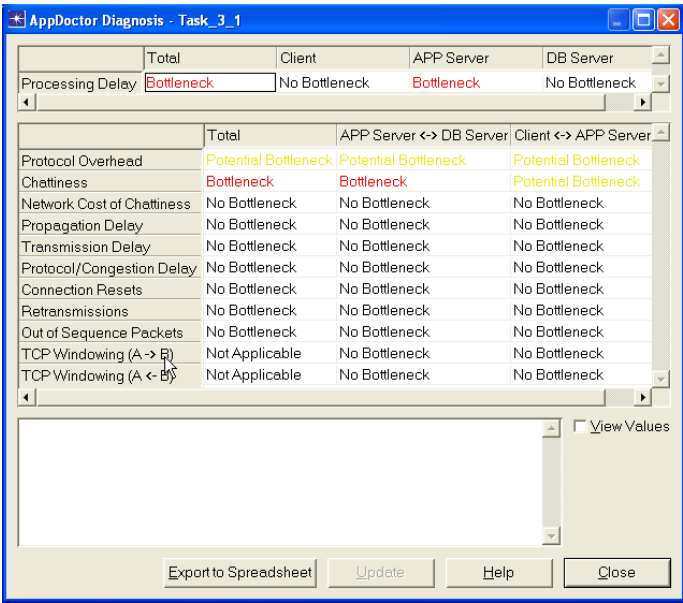


Figure 5.17. Diagnosis window.

Propagation, Transmission, and Protocol/Congestion delays slowing down the transaction.

The „Diagnosis” function (Figure 5.17) provides a more granular analysis of possible bottlenecks by analyzing factors that often cause performance problems in networked applications. Values over a specified threshold are marked as bottlenecks or potential bottlenecks.

	Total	Client	APP Server	DB Server
Busy Time (Seconds)	6.037165	0.600838	5.355270	0.081057
Processing Delay (Seconds)	5.664784	0.260976	5.322750	0.081057
Network Delay (Seconds)	0.893350	Not Applicable	Not Applicable	Not Applicable

	Total	APP Server <-> DB Server	Client <-> APP Server
Response Time (Seconds)	6.558134	2.642250	6.558134
Application Turns	78	39	39
Application Messages	99	40	59
Application Message Bytes	27,450	10,023	17,427
Average Application Message Size (Bytes)	277.27	250.57	295.37
Network Packets	130	47	83
Network Packet Bytes	34,554	12,561	21,993
Average Network Packet Payload Size (Bytes)	265.80	267.26	264.98
Propagation Delay (Seconds)	Not Applicable	0.011392	0.000000
Delay due to Propagation (Seconds)	0.455602	0.455602	0.000000
Transmission Speed (Bits/Second)	Not Applicable	100,000,000	100,000,000
Delay due to Transmission Speed (Seconds)	0.002439	0.000983	0.001455
Protocol/Congestion Delay (Seconds)	0.435325	0.313325	0.122000
Max Application Turn Bytes (A -> B)	Not Applicable	150	414
Max Application Turn Bytes (A <- B)	Not Applicable	4,160	8,833
Max Unacknowledged Data (A -> B) (Bytes)	Not Applicable	150	414
Max Unacknowledged Data (A <- B) (Bytes)	Not Applicable	4,160	6,621
Retransmissions	0	0	0
Out of Sequence Packets	0	0	0
Connection Resets	0	0	0

Figure 5.18. Statistics window.

The diagnosis of the transaction confirms that the primary bottleneck is due to Processing Delay on the Application Server. The processing delay is due to the file I/O, CPU processing, or memory access. It also reveals another bottleneck: the chattiness of the application that leads us to the next step. We investigate the application behaviour in terms of application turns that can be obtained from the transaction statistics. An application turn is a change in direction of the application-message flow.

The statistics of the transaction (Figure 5.18) disclose that the number of application turns is high, i.e., the data sent by the transaction at a time is small. This may cause significant application and network delays. Additionally, a significant portion of application processing time can be spent processing the many requests and responses. The Diagnosis window indicates a „Chattiness” bottleneck without a „Network Cost of Chattiness” bottleneck, which means the following:

- The application does not create significant network delays due to chattiness.
- The application creates significant processing delays due to overhead associated with handling many small application level requests and responses.
- The application’s „Network Cost of Chattiness” could dramatically increase in a high-latency network.

The recommendation is that the application should send fewer, larger application messages. This will utilize network and tier resources more efficiently. For example, a database application should avoid sending a set of records one record at a time.

Would the response time decrease significantly if we added more bandwidth to the link between the client and the APP Server (Figure 5.19)? Answering this question is important



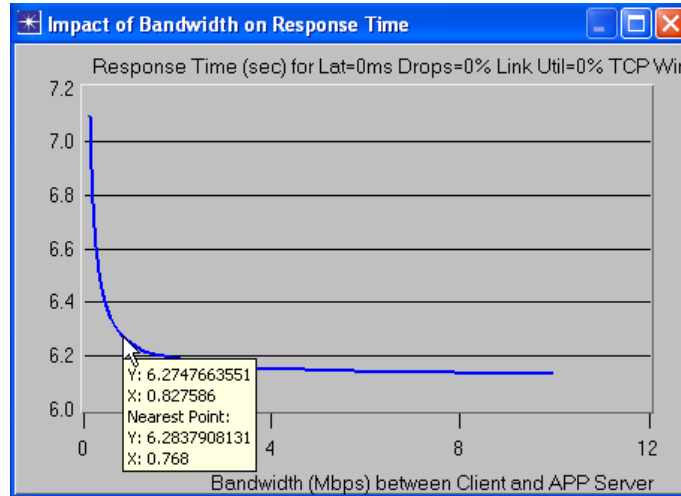


Figure 5.19. Impact of adding more bandwidth on the response time.

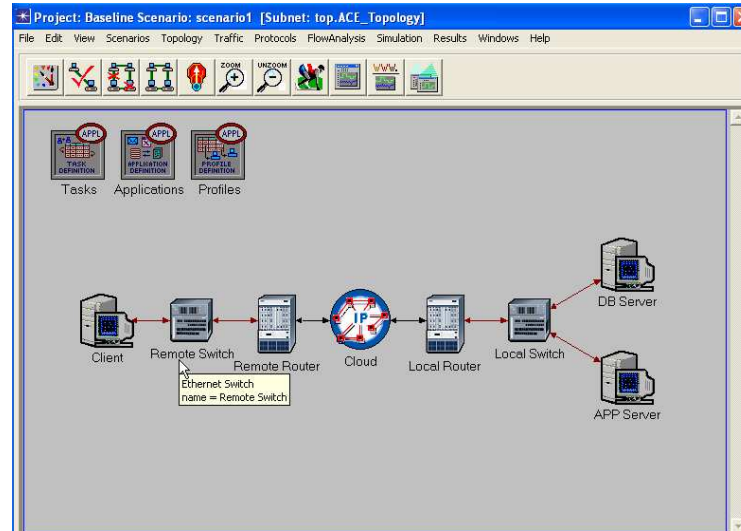
because adding more bandwidth is expensive. Using the prediction feature we can answer the question. In the following chart we selected the bandwidth from 128K to 10Mbps. The chart shows that beyond approximately 827Kbps there is no significant improvement in response time, i.e., for this application the recommended highest bandwidth is no more than 827Kbps, which can be provided by a higher speed DSL line.

After the analysis of the application's performance, we can immediately create the starting baseline model from the captured traffic traces for further simulation studies as illustrated in Figure 5.20.

#### 5.6.6. Sniffer

Another popular network analyser is Network Associates' *Sniffer*. (Network Associates has recently renamed it to Netasyst.) It is a powerful network visualization tool consisting of a set of functions to:

- Capture network traffic for detailed analysis.
- Diagnose problems using the *Expert Analyzer*.
- Monitor network activity in real time.
- Collect detailed utilization and error statistics for individual stations, conversations, or any portion of your network.
- Save historical utilization and error information for baseline analysis.
- Generate visible and audible real-time alarms and notify network administrators when troubles are detected.
- Probe the network with active tools to simulate traffic, measure response times, count hops, and troubleshoot problems.



**Figure 5.20.** Baseline model for further simulation studies.

For further details we refer the reader to the vendors' documentations on <http://www.nai.com>.

## 5.7. Model Development Life Cycle (MDLC)

There are several approaches for network modeling. One possible approach is the creation of a starting model that follows the network topology and approximates the assumed network traffic statistically. After some changes are made, the modeler can investigate the impact of the changes of some system parameters on the network or application performance. This is an approach when it is more important to investigate the performance difference between two scenarios rather than starting from a model based on real network traffic. For instance, assuming certain client/server transactions, we want to measure the change of the response time as the function of the link utilization 20%, 40%, 60%, etc. In this case it is not extremely important to start from a model based on actual network traffic. It is enough to specify certain amount of data transmission estimated by a frequent user or designer. We investigate, for this amount of data, how much the response time will increase as the link utilization increases relative to the starting scenario.

The most common approach for network modeling follows the methodologies of proactive network management. It implies the creation of a network model using actual network traffic as input to simulate current and future behaviour of the network and predict the impact of the addition of new applications on the network performance. By making use of modeling and simulation tools network managers can change the network model by adding new devices, workstations, servers, and applications. Or they can upgrade the links to

higher speed network connections and perform „what-if” scenarios before the implementation of the actual changes. We follow this approach in our further discussions because this approach has been widely accepted in the academia, corporate world, and the industry. In the subsequent paragraphs we elaborate a sequence of modeling steps, called the **Model Development Life Cycle – MDLC** that the author has applied in various real life scenarios of modeling large enterprise networks. The MDLC has the following steps:

- Identification of the topology and network components.
- Data collection.
- Construction and validation of the baseline model. Perform network simulation studies using the baseline.
- Creation of the application model using the details of the traffic generated by the applications.
- Integration of the application and baseline model and completion of simulation studies.
- Further data gathering as the network grows and changes and as we know more about the applications.
- Repeat the same sequence.

In the following, we expand the steps above:

#### **Identification of the topology and network components**

Topology data describes the physical network components (routers, circuits, and servers) and how they are connected. It includes the location and configuration description of each internetworking device, how those devices are connected (the circuit types and speeds), the type of LANs and WANs, the location of the servers, addressing schemes, a list of applications and protocols, etc.

#### **Data collection**

In order to build the baseline model we need to acquire topology and traffic data. Modelers can acquire topology data either by entering the data manually or by using network management tools and network devices’ configuration files. Several performance management tools use the **Simple Network Management Protocol – SNMP** to query the *Management Information Base (MIB)* maintained by SNMP agents running in the network’s routers and other internetworking devices. This process is known as an SNMP discovery. We can import topology data from routers’ configuration files to build a representation of the topology for the network in question. Some performance management tools can import data using the map file from a network management platform, such as HP OpenView or IBM NetView. Using the network management platform’s export function, the map file can be imported by modeling.

The network traffic input to the baseline model can be derived from various sources: Traffic descriptions from interviews and network documents, design or maintenance documents, MIB/SNMP reports and network analyser and **Remote Monitoring** – traffic traces. RMON is a network management protocol that allows network information to be gathered at a single node. RMON traces are collected by RMON probes that collect data at different levels of the network architecture depending on the probe’s standard. Table 5.21 includes the most widely used standards and the level of data collection:

	RMON1	RMON2	Enterprise RMON
Ethernet/Token Ring	X	X	X
MAC Layer Monitoring	X	X	X
Network Layer Monitoring		X	X
Application Layer Monitoring		X	X
Switched LAN, Frame Relay, ATM			X
VLAN Support			X
Application response time			X

**Figure 5.21.** Comparison of RMON Standards.

Network traffic can be categorized as usage-based data and application-based data. The primary difference between usage- and application-based data is the degree of details that the data provides and the conclusions that can be made based on the data. The division can be clearly specified by two adjacent OSI layers, the Transport layer and the Session layer: usage-based data is for investigating the performance issues through the transport layer; application-based data is for analyzing the rest of the network architecture above the Transport layer. (In Internet terminology this is equivalent to the cut between the TCP level and the applications above the TCP level.)

The goal of collecting usage-based data is to determine the total traffic volume before the applications are implemented on the network. Usage-based data can be gathered from SNMP agents in routers or other internetworking devices. SNMP queries sent to the routers or switches provide statistics about the exact number of bytes that have passed through each LAN interface, WAN circuit, or (Permanent Virtual Circuit – PVC) interfaces. We can use the data to calculate the percentage of utilization of the available bandwidth for each circuit.

The purpose of gathering application-based data is to determine the amount of data generated by an application and the type of demand the application makes. It allows the modeler to understand the behaviour of the application and to characterize the application level traffic. Data from traffic analysers or from RMON2-compatible probes, Sniffer, NETScout Manager, etc., provide specifics about the application traffic on the network. Strategically placed data collection devices can gather enough data to provide clear insight into the traffic behaviour and flow patterns of the network applications. Typical application level data collected by traffic analysers:

- The type of applications.
- Hosts communicating by network layer addresses (i.e., IP addresses).
- The duration of the network conversation between any two hosts (start time and end time).
- The number of bytes in both the forward and return directions for each network conversation.
- The average size of the packets in the forward and return directions for each network conversation.
- Traffic burstiness.
- Packet size distributions.
- Packet interarrival distributions.
- Packet transport protocols.
- Traffic profile, i.e., message and packet sizes, interarrival times, and processing delays.
- Frequency of executing application for a typical user.
- Major interactions of participating nodes and sequences of events.

#### **Construction and validation of the baseline model. Perform network simulation studies using the baseline**

The goal of building a baseline model is to create an accurate model of the network as it exists today. The baseline model reflects the current „as is” state of the network. All studies will assess changes to the baseline model. This model can most easily be validated since its predictions should be consistent with current network measurements. The baseline model generally only predicts basic performance measures such as resource utilization and response time.

The baseline model is a combination of the topology and usage-based traffic data that have been collected earlier. It has to be validated against the performance parameters of the current network, i.e., we have to prove that the model behaves similarly to the actual network activities. The baseline model can be used either for analysis of the current network or it can serve as the basis for further application and capacity planning. Using the import functions

of a modeling tool, the baseline can be constructed by importing first the topology data gathered in the data collection phase of the modeling life cycle. Topology data is typically stored in topology files (.top or .csv) created by Network Management Systems, for instance HP OpenView or Network Associate's Sniffer. Traffic files can be categorized as follows:

- Conversation pair traffic files that contain aggregated end-to-end network load information, host names, packet counts, and byte counts for each conversation pair. The data sets allow the modeling tool to preserve the bursty nature of the traffic. These files can be captured by various data collection tools.
- Event trace traffic files that contain network load information in the form of individual conversations on the network rather than summarized information. During simulation the file can replay the captured network activity on an event by event basis.

Before simulation the modeler has to decide on the following simulation parameters:

- *Run length:* **Runtime length** must exceed the longest message delay in the network. During this time the simulation should produce sufficient number of events to allow the model to generate enough samples of every event.
- *Warm-up period:* The simulation **warm-up period** is the time needed to initialize packets, buffers, message queues, circuits, and the various elements of the model. The warm-up period is equal to a typical message delay between hosts. Simulation warm-up is required to ensure that the simulation has reached steady-state before data collection begins.
- *Multiple replications:* There may be a need for multiple runs of the same model in cases when statistics are not sufficiently close to true values. We also need multiple runs prior to validation when we execute multiple replicates to determine variation of statistics between replications. A common cause of variation between replications is rare events.
- *Confidence interval:* A confidence interval is an interval used to estimate the likely size of a population parameter. It gives an estimated range of values that has a specified probability of containing the parameter being estimated. Most commonly used intervals are the 95% and 99% confidence intervals that have .95 and .99 probabilities respectively of containing the parameter. In simulation, confidence interval provides an indicator of the precision of the simulation results. Fewer replications result in a broader confidence interval and less precision.

In many modeling tools, after importing both the topology and traffic files, the baseline model is created automatically. It has to be checked for construction errors prior to any attempts at validation by performing the following steps:

- Execute a preliminary run to confirm that all source-destination pairs are present in the model.
- Execute a longer simulation with warm-up and measure the sent and received message counts and link utilization to confirm that correct traffic volume is being transmitted.

Validating the baseline model is the proof that the simulation produces the same performance parameters that are confirmed by actual measurements on the physical network. The network parameters below can usually be measured in both the model and in the physical network:

- Number of packets sent and received
- Buffer usage
- Packet delays
- Link utilization
- Node's CPU utilization

Confidence intervals and the number of independent samples affect how close a match between the model and the real network is to be expected. In most cases, the best that we can expect is an overlap of the confidence interval of predicted values from the simulation and the confidence interval of the measured data. A very close match may require too many samples of the network and too many replications of the simulation to make it practical.

#### **Creation of the application model using the details of the traffic generated by the applications**

Application models are studied whenever there is a need to evaluate the impact of a networked application on the network performance or to evaluate the application's performance affected by the network. Application models provide traffic details between network nodes generated during the execution of the application. The steps of building an application model are similar to the ones for baseline models.

- Gather data on application events and user profiles.
- Import application data into a simulation model manually or automatically.
- Identify and correct any modeling errors.
- Validate the model.

#### **Integration of the application and baseline models and completion of simulation studies**

The integration of the application model(s) and baseline model follows the following steps:

- Start with the baseline model created from usage-based data.
- Use the information from the application usage scenarios (locations of users, number of users, transaction frequencies) to determine where and how to load the application profiles onto the baseline model.
- Add the application profiles generated in the previous step to the baseline model to represent the additional traffic created by the applications under study.

Completion of Simulation studies consists of the following steps:

- Use a modeling tool to run the model or simulation to completion.
- Analyse the results: Look at the performance parameters of the target transactions in comparison to the goals established at the beginning of the simulation.
- Analyse the utilization and performance of various network elements, especially where the goals are not being met.

Typical simulation studies include the following cases:

- Capacity analysis

Capacity analysis studies the changes of network parameters, for instance:

- Changes in the number and location of users.
- Changes in network elements capacity.
- Changes in network technologies.

A modeler may be interested in the effect of the changes above on the following network parameters:

- Switches and routers' utilization
  - Communications link utilization
  - Buffer utilization
  - Retransmitted and lost packets
- Response time analysis
 

The scope of response time analysis is the study of message and packet transmission delay:

    - Application and network level packet end-to-end delay.
    - Packet round trip delay.
    - Message/packet delays.
    - Application response time.
  - Application Analysis
 

The scope of application studies is the ratio of the total application response time relative to the individual components of network and application delay. Application's analysis provides statistics of various measures of network and application performance in addition to the items discussed in a previous section.

#### **Further data gathering as the network grows and as we know more about the applications**

The goal of this phase is to analyse or predict how a network will perform both under current conditions and when changes to traffic load (new applications, users, or network structure) are introduced:

- Identify modifications to the network infrastructure that will alter capacity usage of the network's resources.
- A redesign can include increasing or decreasing capacity, relocating network elements among existing network sites, or changing communications technology.
- Modify the models to reflect these changes.
- Assess known application development or deployment plans in terms of projected network impact.
- Assess business conditions and plans in terms of their impact on the network from projected additional users, new sites, and other effects of the plans.
- Use ongoing baselining techniques to watch usage trends over time, especially related to Internet and intranet usage.



## 5.8. Methodology for modeling the impact of traffic burstiness on high-speed networks

Recent measurements of local area network traffic and wide-area network traffic have proved that the widely used Markovian process models cannot be applied for today's network traffic. If the traffic were a Markovian process, the traffic's burst length would be smoothed by averaging over a long time scale, contradicting the observations of today's traffic characteristics. Measurements of real traffic also prove that traffic burstiness is present on a wide range of time scales. Traffic that is bursty on many or all time scales can be characterized statistically using the concept of *self-similarity*. Self-similarity is often associated with objects in fractal geometry, objects that appear to look alike regardless of the scale at which they are viewed. In case of stochastic processes like time series, the term self-similarity refers to the process' distribution, which, when viewed at varying time scales, remains the same. Self-similar time series has noticeable bursts, which have long periods with extremely high values on all time scales. Characteristics of network traffic, such as packets/sec, bytes/sec, or length of frames, can be considered as stochastic time series. Therefore, measuring traffic burstiness is the same as characterizing the self-similarity of the corresponding time series.

The self-similarity of network traffic has also been observed in studies in numerous papers, such as [21], [22], and [23]. These and other papers show that packet loss, buffer utilization, and response time are totally different when simulations use either real traffic data or synthetic data that include self-similarity.

### Background

Let  $X = (X_t : t = 0, 1, 2, \dots)$  be a covariance stationary stochastic process. Such a process has a constant mean  $\mu = E[X_t]$ , finite variance  $\sigma^2 = E[(X_t - \mu)^2]$ , and an autocorrelation function  $r(k) = E[(X_t - \mu)(X_{t+k} - \mu)] / E[(X_t - \mu)^2]$  ( $k = 0, 1, 2, \dots$ ), that depends only on  $k$ . It is assumed that  $X$  has an autocorrelation function of the form:

$$r(k) \sim \alpha k^{-\beta}, \quad k \rightarrow \infty \quad (5.1)$$

where  $0 < \beta < 1$  and  $\alpha$  is a positive constant. Let  $X^{(m)} = (X_{(k)}^{(m)} : k = 1, 2, 3, m = 1, 2, 3, \dots)$  represent a new time series obtained by averaging the original series  $X$  over nonoverlapping blocks of size  $m$ . For each  $m = 1, 2, 3, \dots$ ,  $X^{(m)}$  is specified by  $X_k^{(m)} = (X_{km-m+1} + \dots + X_{km})/m$ , ( $k \geq 1$ ). Let  $r^{(m)}$  denote the autocorrelation function of the aggregated time series  $X^{(m)}$ .

### Definition of self-similarity

The process  $X$  called exactly *self-similar* with self-similarity parameter  $H = 1 - \beta/2$  if the corresponding aggregated processes  $X^{(m)}$  have the same correlation structure as  $X$ , i.e.  $r^{(m)}(k) = r(k)$  for all  $m = 1, 2, \dots$  ( $k = 1, 2, 3, \dots$ ).

A covariance stationary process  $X$  is called *asymptotically self-similar* with self-similarity parameter  $H = 1 - \beta/2$ , if for all  $k$  large enough  $r^{(m)}(k) \rightarrow r(k)$ , as  $m \rightarrow \infty$ ,  $0.5 \leq H \leq 1$ .

### Definition of long-range dependency

A stationary process is called *long-range dependent* if the sum of the autocorrelation values approaches infinity:  $\sum_k r(k) \rightarrow \infty$ . Otherwise, it is called *short-range dependent*. It can be

derived from the definitions that while short-range dependent processes have exponentially decaying autocorrelations, the autocorrelations of long-range dependent processes decay hyperbolically; i.e., the related distribution is heavy-tailed. In practical terms, a random variable with heavy-tail distribution generates extremely large values with high probability [26, 27]. The degree of self-similarity is expressed by the parameter  $H$  or **Hurst-parameter**. The parameter represents the speed of decay of a process' autocorrelation function. As  $H \rightarrow 1$  the extent of both self-similarity and long-range dependence increases. It can also be shown that for self-similar processes with long-range dependency  $H > 0.5$  [28].

### Traffic models

Traffic modeling originates in traditional voice networks. Most of the models have relied on the assumption that the underlying processes are Markovian [29] (or more general, short-range dependent). However, today's high-speed digital packet networks are more complex and bursty than traditional voice traffic due to the diversity of network services and technologies.

Several sophisticated stochastic models have been developed as a reaction to new developments, such as Markov-modulated Poisson processes, fluid flow models, Markovian arrival processes, batched Markovian arrival process models, packet train models, and *Transform-Expand-Sample models*. These models mainly focus on the related queuing problem analytically. They are usually not compared to real traffic patterns and not proven to match the statistical property of actual traffic data.

Another category of models attempts to characterize the statistical properties of actual traffic data. For a long time, the area of networking research has lacked adequate traffic measurements. However, during the past years, large quantities of network traffic measurements have become available and collected in the Web and high-speed networks. Some of these data sets consist of high-resolution traffic measurements over hours, days, or weeks, (e.g., [27, 43, 44, 45, 46, 47, 48, 49, 50]). Other data sets provide information over time periods ranging from weeks to months and years. See, for instance, in [51, 52], SS7 data sets in [53, 54], and WAN measurements [55, 56, 57]. Statistical analyses of these high time-resolution traffic measurements have proved that actual traffic data from packet networks reveal self-similarity. These results point out the difference between traditional models and measured traffic data. While the assumed processes in traditional packet traffic models are short-range dependent, measured packet traffic data show evidence of long-range dependency. Figure 5.22 illustrates the difference between Internet traffic and voice traffic for different numbers of aggregated users. As the number of voice flows increases, the traffic becomes more and more smoothed contrary to the Internet traffic.

Quite the opposite to the well developed field of short-range dependent queuing models, fewer theoretical results exist for queueing systems with long-range dependence. For some of the results, see [59, 60, and 61]. In terms of modeling, the two major groups of self-similar models are fractional Gaussian noises and fractional ARIMA processes. The Gaussian models accurately represent aggregation of many traffic streams as it has been proved in [62] and illustrated by simulation in [63]. The results in [21] demonstrated the effect of long-range dependency on queue length statistics of a single queue system through a controlled fractionally differenced ARIMA (1, d, 0) input process. Another well-known model, the M/Pareto model has been used in modeling network traffic that is not sufficiently aggregated for the Gaussian model to apply, for example [63, 64].

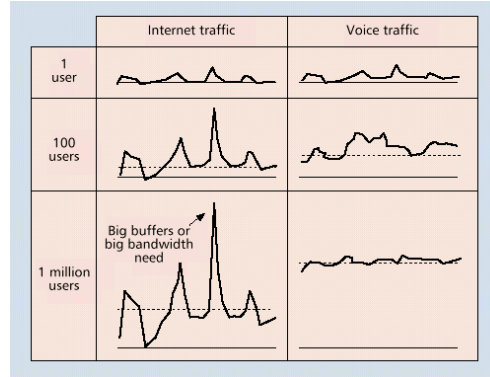


Figure 5.22. The self-similar nature of Internet network traffic

### Black box vs. structural models

We share the authors' opinion in [65, 66] calling the approach of traditional time series analysis as black box modeling as opposite to the structural modeling that concentrates on the environment in which the models' data was collected; i.e., the complex hierarchies of network components that make up today's communications systems. While the authors admit that black box models can be and are useful in other contexts, they argue that black box models are of no use for understanding the dynamic and complex nature of the traffic in modern packet networks. Black box models have not much use in designing, managing and controlling today's networks either. In order to provide physical explanations for empirically observed phenomena such as long-range dependency, we need to replace black box models with structural models. The attractive feature of structural traffic models is that they take into account the details of the layered architecture of today's networks and can analyse the interrelated network parameters that ultimately determine the performance and operation of a network. Time series models usually handle these details as black boxes [65]. Because actual networks are complex systems, in many cases, black box models assume numerous parameters to represent a real system accurately. For network designers, who are important users of traffic modeling, black box models are not very useful. It is rarely possible to measure or estimate the model's numerous parameters in a complex network environment. For a network designer, a model ought to be simple, meaningful in a particular network. It can relay on actual network measurements, and the result ought to be relevant to the performance and the operation of a real network.

For a long time, traffic models were developed independently of traffic data collected in real networks. These models could not be applied in practical network design. Today the availability of huge data sets of measured network traffic and the increasing complexity of the underlying network structure emphasize the application of the Ockham' Razer in network modeling. (Ockham's Razor is a principle of the mediaeval philosopher William Ockham. According to his principle, modelers should not make more assumptions than the minimum needed. This principle is also called the Principle of Parsimony and motivates all scientific modeling and theory building. It states that modelers should choose the simplest model among a set of otherwise equivalent models of a given phenomenon. In any given

model, Ockham's Razor helps modelers include only those variables that are really needed to explain the phenomenon. Following the principle, model development will become easier, reducing the possibilities for inconsistencies, ambiguities and redundancies [67].)

Structural models are presented, for instance in [66] and [68]. The papers demonstrate how the self-similar nature of aggregated network traffic of all conversations between hosts explains the details of the traffic dynamics at the level generated by the individual hosts. The papers introduce structural traffic models that have a physical meaning in the network context and underline the predominance of long-range dependence in the packet arrival patterns generated by the individual conversations between hosts. The models provide insight into how individual network connections behave in local and wide area networks. Although the models go beyond the black box modeling methodology by taking into account the physical structure of the aggregated traffic patterns, they do not include the physical structure of the intertwined structure of links, routers, switches, and their finite capacities along the traffic paths.

In [20], the authors demonstrated that World Wide Web traffic shows characteristics that are consistent with self-similarity. They show that transmission times may be heavytailed, due to the distribution of available file sizes in the Web. It is also shown that silent times may also be heavy-tailed; primarily due to the effect of user „think time”. Similarly to the structural models in [66, 68], this paper lacks of analyzing the impact of selfsimilar traffic on the parameters of the links and the routers' buffers that ultimately determine a network's performance.

The effect of self-similar traffic on queue length statistics for real ATM VBR traffic is studied in [69]. Making use of fractionally- differenced ARIMA (1, d, 0) for the input process, the authors study the queue length process for varying-level of mean utilization. The results of the simulation model demonstrate that the increased buffer size do not significantly decrease cell loss probabilities for long-range dependent processes. The models developed under the S+ package [70] focus on the queue length of a single-server queue. They do not simulate the combined, overall effect of the interrelated, individual components of a network, such as links, buffers, and applications' response time.

Our paper describes a traffic model that belongs to the structural model category above. We implement the M/Pareto model within the discrete event simulation package COMNET that allows the analysis of the negative impact of self-similar traffic on not just one single queue, but on the overall performance of various interrelated network components, such as link, buffers, response time, etc. The commercially available package does not readily provide tools for modeling self-similar, long-range dependent network traffic. The model-generated traffic is based on measurements collected from a real ATM network. The choice of the package emphasizes the need for integrated tools that could be useful not just for theoreticians, but also for network engineers and designers. Our paper intends to narrow the gap between existing, well-known theoretical results and their applicability in everyday, practical network analysis and modeling. It is highly desirable that appropriate traffic models should be accessible from measuring, monitoring, and controlling tools. Our model can help network designers and engineers, the ultimate users of traffic modeling, understand the dynamic nature of network traffic and assist them to design, measure, monitor, and control today's complex, high-speed networks in their everyday's practice.

### Implications of burstiness on high-speed networks

Various papers [33, 50, 71, 72, and 73] discuss the impact of burstiness on network congestion. Their conclusions are:

- Congested periods can be quite long with losses that are heavily concentrated.
- Linear increases in buffer size do not result in large decreases in packet drop rates.
- A slight increase in the number of active connections can result in a large increase in the packet loss rate.

Results show that packet traffic „spikes” (which cause actual losses) ride on longer-term „ripples”, which in turn ride on still longer-term „swells” [48].

Another area where burstiness can affect network performance is a link with priority scheduling between classes of traffic. In an environment, where the higher priority class has no enforced bandwidth limitations (other than the physical bandwidth), interactive traffic might be given priority over bulk-data traffic. If the higher priority class is bursty over long time scales, then the bursts from the higher priority traffic could obstruct the lower priority traffic for long periods of time.

The burstiness may also have an impact on networks where the admission control mechanism is based on measurements of recent traffic, rather than on policed traffic parameters of individual connections. Admission control that considers only recent traffic patterns can be misled following a long period of fairly low traffic rates.

#### 5.8.1. Model parameters

Each transaction between a client and a server consists of active periods followed by inactive periods. Transactions consist of groups of packets sent in each direction. Each group of packets is called a burst. The burstiness of the traffic can be characterized by the following time parameters:

- **Transaction Interarrival Time (TIAT):** The time between the first packet in a transaction and the first packet of the next immediate transaction.
- **Burst Interarrival Time,  $1/\lambda$ ,  $\lambda$  arrival rate of bursts:** The time between bursts.
- **Packet Interarrival Time,  $1/r$ ,  $r$ : arrival rate of packets:** The time between packets in a burst.

#### The Hurst parameter

It is anticipated that the rapid and ongoing aggregation of more and more traffic onto integrated multiservice networks will eventually result in traffic smoothing. Once the degree of aggregation is sufficient, the process can be modeled by Gaussian process [63]. Currently, network traffic does not show characteristics that close to Gaussian. In many networks the degree of aggregation is not enough to balance the negative impact of bursty traffic. However, before traffic becomes Gaussian, existing methods can still provide accurate measurement and prediction of bursty traffic.

Most of the methods are based on the estimate of the Hurst parameter  $H$  - the higher the value of  $H$ , the higher the burstiness, and consequently, the worse the queuing performance of switches and routers along the traffic path. Some are more reliable than others. The reliability depends on several factors; e.g., the estimation technique, sample size, time

scale, traffic shaping or policing, etc. Based on published measurements we investigated methods with the smallest estimation error\*. <sup>1</sup> Among those, we chose the *Rescaled Adjusted Range (R/S)* method because we found it implemented in the Benoit package [74]. The Hurst parameter calculated by the package is input to our method.

### The M/Pareto traffic model and the Hurst parameter

Recent results in [63, 75] have proven that the M/Pareto model is appropriate for modeling long-range dependent traffic flow characterized by long bursts. Originally, the model was introduced in [75] and applied in the analysis of ATM buffer levels. The M/Pareto model was also used in [76, 77, 78, and 79] to predict the queuing performance of Ethernet, VBR video, and IP packet streams in a single server queue. We apply the M/Pareto model not just for a single queue, but also for predicting the performance of an interconnected system of links, switches and routers affecting the individual network elements' performance. We make use of some of the calculations presented in [63, 75, 76, 77, and 78].

The M/Pareto model is a Poisson process of overlapping bursts with arrival rate  $\lambda$ . A burst generates packets with arrival rate  $r$ . Each burst, from the time of its interval, will continue for a Pareto-distributed time period. The use of Pareto distribution results in generating extremely long bursts that characterize long-range dependent traffic.

The probability that a Pareto-distributed random variable  $X$  exceeds threshold  $x$  is:

$$\Pr \{X > x\} = \begin{cases} \left(\frac{x}{\delta}\right)^{\gamma}, & x \geq \delta \\ 1, & \text{otherwise,} \end{cases} \quad (5.2)$$

$$1 < \gamma < 2, \delta > 0.$$

The mean of  $X$ , the mean duration of a burst  $\mu = \delta\gamma/(\gamma - 1)$  and its variance is infinite [76]. Assuming a  $t$  time interval, the mean number of packets  $M$  in the time interval  $t$  is:

$$M = \lambda t r \delta \gamma / (\gamma - 1), \quad (5.3)$$

where

$$\lambda = \frac{M(\gamma - 1)}{t r \delta \gamma}. \quad (5.4)$$

The M/Pareto model is described in [76, 77, 78, and 79] as asymptotically self-similar and it is shown that for the Hurst parameter the following equation holds:

$$H = \frac{3 - \gamma}{2}. \quad (5.5)$$

### 5.8.2. Implementation of the Hurst parameter in the COMNET modeling tool

We implemented the Hurst parameter and a modified version of the M/Pareto model in the discrete event simulation system COMNET [80, 81]. By using discrete event simulation methodology, we can get realistic results in measuring network parameters, such as utilization of links and the queuing performance of switches and routers. Our method can model and measure the harmful consequences of aggregated bursty traffic and predict its impact on the overall network's performance.

---

<sup>1</sup>Variance, Aggregated Variance, Higuchi, Variance of Residuals, Rescaled Adjusted Range (R/S), Whittle Estimator, Periodogram, Residuals of Regression [59].

Delta Time (sec)	Average Bandwidth Utilization %	Bytes Total/ sec	Bytes in/ sec	Bytes out/ sec
299	2.1	297.0	159.2	137.8
300	2.2	310.3	157.3	153.0
301	2.1	296.8	164.4	132.4
302	2.7	373.2	204.7	168.5
...	...	...	...	...

**Figure 5.23.** Traffic traces

Bytes átlagos száma	Message delay (ms)	Buffer level (byte)	Dropped packets száma	Links' Mean Bandwidth Utilization (%)		
				56 Kbps Frame Relay	ATM DS-3 segment	100 Mbps Ethernet
440.4279	78.687	0.04	0	3.14603	0.06	0.0031

**Figure 5.24.** Measured network parameters

### Traffic measurements

In order to build the baseline model, we collected traffic traces in a large corporate network by the Concord Network Health network analyser system. We took measurements from various broadband and narrowband links including 45Mbps ATM, 56Kbps, and 128 Kbps frame relay connections. The Concord Network Health system can measure the traffic in certain time intervals at network nodes, such as routers and switches. We set the time intervals to 6000 seconds and measured the number of bytes and packets sent and received per second, packet latency, dropped packets, discard eligible packets, etc. Concord Network Health cannot measure the number of packets in a burst and the duration of the bursts as it is assumed in the M/Pareto model above. Due to this limitation of our measuring tool, we slightly modify our traffic model according to the data available. We took snapshots of the traffic in every five minutes from a narrowband frame relay connection between a remote client workstation and a server at the corporate headquarters as traffic destination in the following format:

The mean number of bytes, the message delay from the client to server, the input buffer level at the client's local router, the number of blocked packets, the mean utilizations of the 56Kbps frame relay, the DS-3 segment of the ATM network, and the 100Mbps Ethernet link at the destination are summarized in Table 5.24:

COMNET represents a transaction by a message source, a destination, the size of the message, communication devices, and links along the path. The rate at which messages are sent is specified by an interarrival time distribution, the time between two consecutive packets. The Poisson distribution in the M/Pareto model generates bursts or messages with arrival rate  $\lambda$ , the number of arrivals, which are likely to occur in a certain time interval. In simulation, this information is expressed by the time interval between successive arrivals  $1/\lambda$ . For this purpose, we use the Exponential distribution. Using the Exponential distribution for interarrival time will result in an arrival pattern characterized by the Poisson distribution. In COMNET, we implemented the interarrival time with the function  $\text{Exp}(1/\lambda)$ . The interarrival time in the model is set to one second matching the sampling time interval set

in Concord Network Health and corresponding to an arrival rate  $\lambda = 1/\text{sec}$ .

In the M/Pareto model, each burst continues for a Pareto-distributed time period. The Concord Network Health cannot measure the duration of a burst; hence, we assume that a burst is characterized by the number of bytes in a message sent or received in a second. Since the ATM cell rate algorithm ensures that equal length messages are processed in equal time, then longer messages require longer processing time. So we can say that the distribution of the duration of bursts is the same as the distribution of the length of bursts. Hence, we can modify the M/Pareto model by substituting the Pareto-distributed duration of bursts with the Pareto-distributed length of bursts. We derive  $\delta$  of the Pareto distribution not from the mean duration of bursts, but from the mean length of bursts.

The Pareto distributed length of bursts is defined in COMNET by two parameters- the location and the shape. The location parameter corresponds to the  $\delta$ , the shape parameter corresponds to the  $\gamma$  parameter of the M/Pareto model in (1) and can be calculated from the relation (4) as

$$\gamma = 3 - 2H . \quad (5.6)$$

The Pareto distribution can have infinite mean and variance. If the shape parameter is greater than 2, both the mean and variance are finite. If the shape parameter is greater than 1, but less than or equal to 2, the mean is finite, but then the variance is infinite. If the shape parameter is less than or equal to 1, both the mean and variance are infinite.

From the mean of the Pareto distribution we get:

$$\delta = \frac{\mu \cdot (\gamma - 1)}{\gamma} . \quad (5.7)$$

The relations (5) and (6) allow us to model bursty traffic based on real traffic traces by performing the following steps:

- a. Collect traffic traces using the Concord Network Health network analyser.
- b. Compute the Hurst parameter  $H$  by making use of the Benoit package with the traffic trace as input.
- c. Use the Exponential and Pareto distributions in the COMNET modeling tool with the parameters calculated above to specify the distribution of the interarrival time and length of messages.
- d. Generate traffic according to the modified M/Pareto model and measure network performance parameters.

The traffic generated according to the steps above is bursty with parameter  $H$  calculated from real network traffic.

### 5.8.3. Validation of the baseline model

We validate our baseline model by comparing various model parameters of a 56Kbps frame relay and a 6Mbps ATM connection with the same parameters of a real network as the Concord Network Health network analyser traced it. For simplicity, we use only the „Bytes Total/sec” column of the trace, i.e., the total number of bytes in the „Bytes Total/sec” column is sent in one direction only from the client to the server. The Hurst parameter of the real



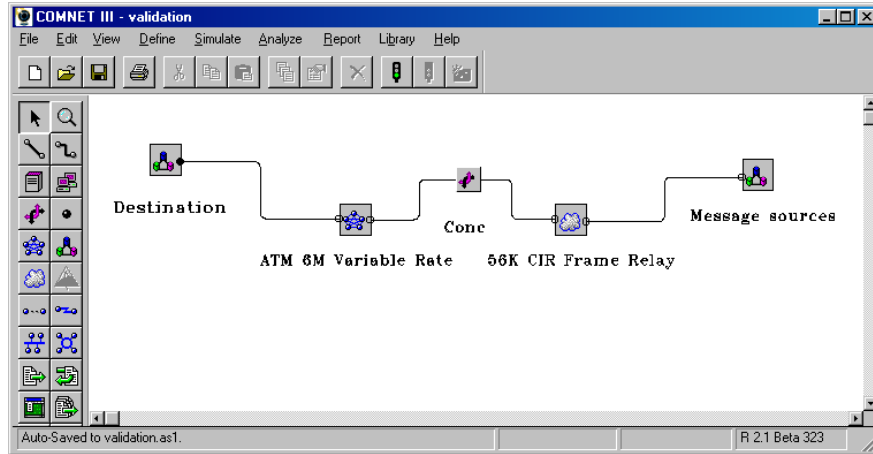


Figure 5.25. Part of the real network topology where the measurements were taken.

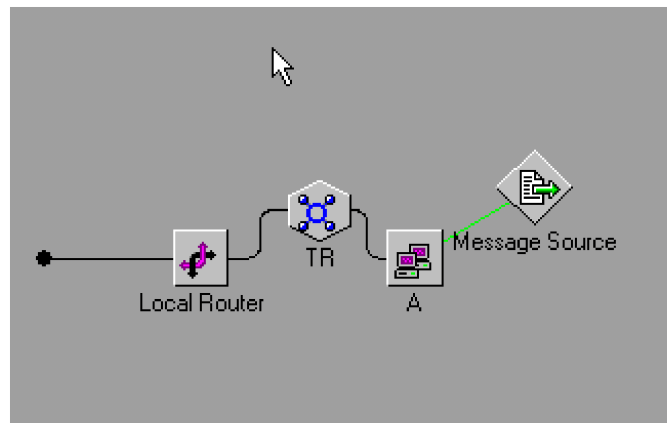


Figure 5.26. „Message Source” remote client

traffic trace is  $H = 0.55$  calculated by the Benoit package. The topology is as follows:

The „Message sources” icon is a subnetwork that represents a site with a token ring network, a local router, and a client *A* sending messages to the server *B* in the „Destination” subnetwork:

The interarrival time and the length of messages are defined by the Exponential and Pareto functions Exp (1) and Par (208.42, 1.9) respectively. The Pareto distribution’s location (208.42) and shape (1.9) are calculated from formulas (5) and (6) by substituting the mean length of bursts (440 bytes from Table 2.) and  $H = 0.55$ .

The corresponding heavy-tailed Pareto probability distribution and cumulative distribution functions are illustrated in Figure 5.28 (The *X* – axis represents the number of bytes):

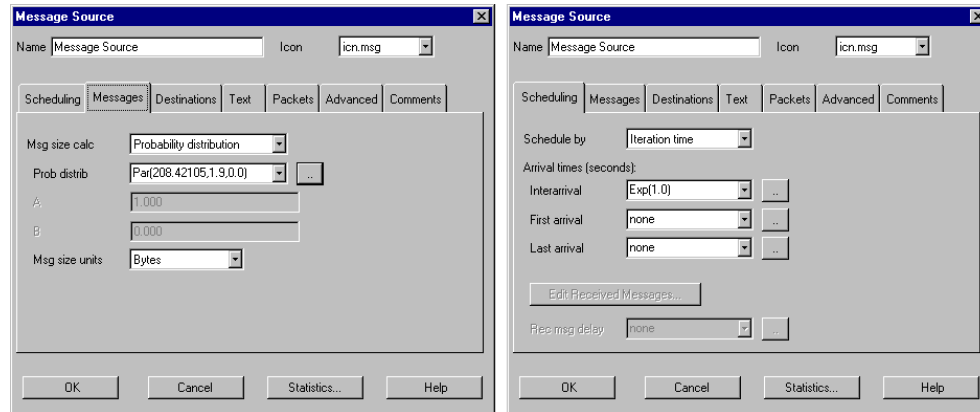


Figure 5.27. Interarrival time and length of messages sent by the remote client

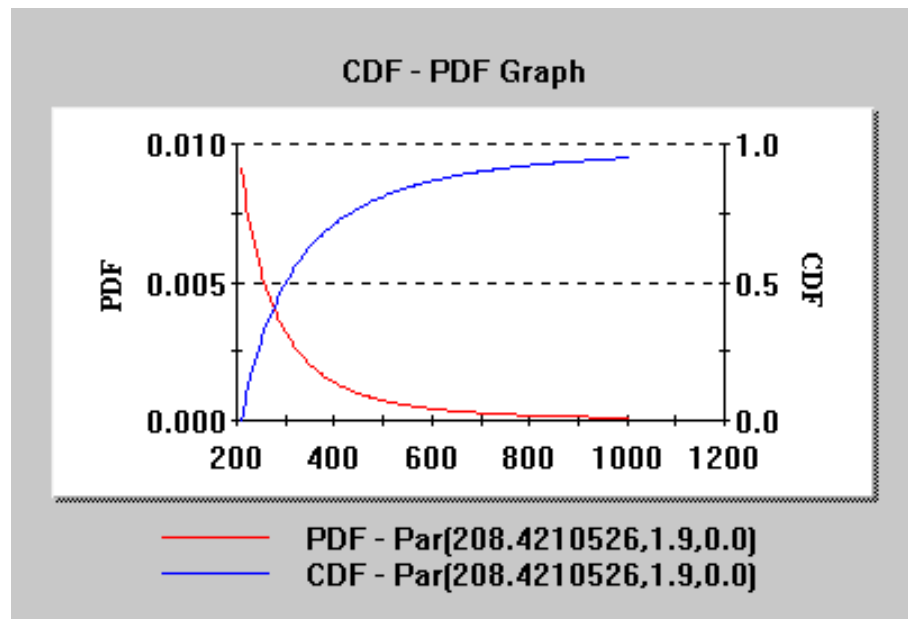


Figure 5.28. The Pareto probability distribution for mean 440 bytes and Hurst parameter  $H = 0.55$

The „Frame Relay” icon represents a frame relay cloud with 56K committed information rate (CIR). The „Conc” router connects the frame relay network to a 6Mbps ATM network with variable rate control (VBR) as shown in Figures 5.29 and 5.30:

The „Destination” icon denotes a subnetwork with server *B*:

The results of the model show almost identical average for the utilization of the frame relay link (0.035 ~3.5%) and the utilization of the real measurements (3.1%):

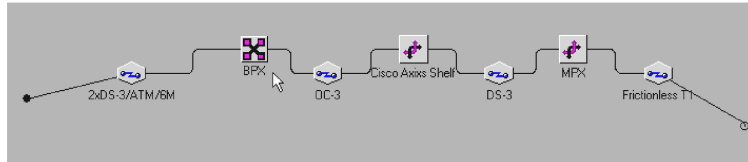


Figure 5.29. The internal links of the 6Mbps ATM network with variable rate control (VBR)

Figure 5.30. Parameters of the 6Mbps ATM connection

The message delay in the model is also very close to the measured delay between the client and the server (78 msec):

The input buffer level of the remote client's router in the model is almost identical with the measured buffer level of the corresponding router:

Similarly, the utilizations of the model's DS-3 link segment of the ATM network and

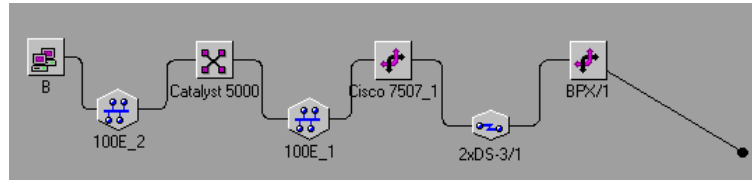


Figure 5.31. The „Destination” subnetwork

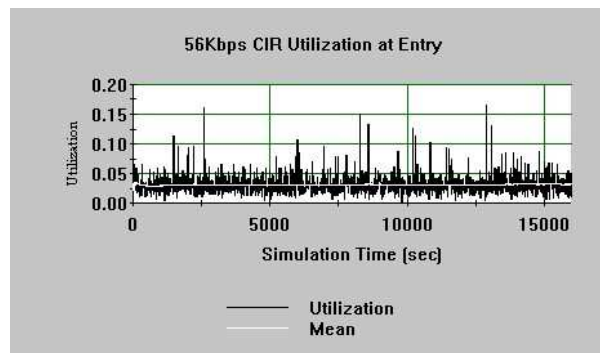


Figure 5.32. Utilization of the frame relay link in the baseline model

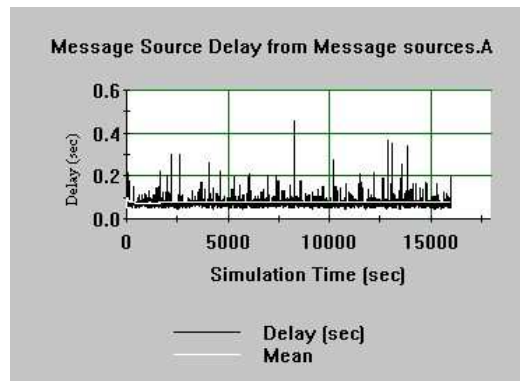


Figure 5.33. Baseline message delay between the remote client and the server

the Ethernet link in the destination network closely match with the measurements of the real network:

It can also be shown from the model's traffic trace that for the model generated messages the Hurst parameter  $H = 0.56$ , i.e., the model generates almost the same bursty traffic as the real network. Furthermore, the number of dropped packets in the model was zero similarly to the number of dropped packets in the real measurements. Therefore, we start from a

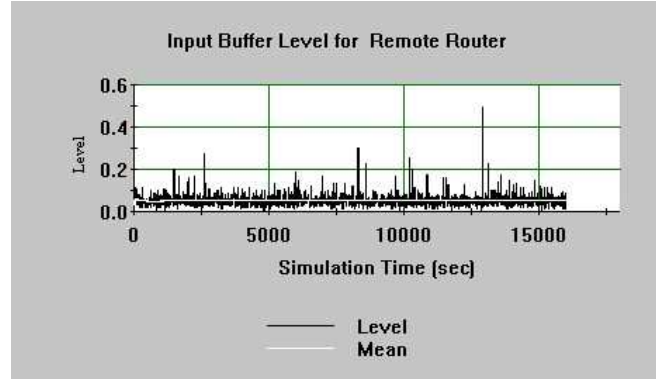


Figure 5.34. Input buffer level of remote router

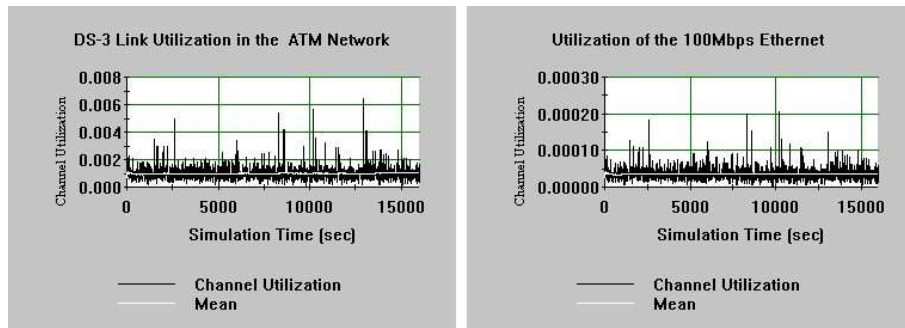


Figure 5.35. Baseline utilizations of the DS-3 link and Ethernet link in the Destination

model that closely represents the real network.

#### 5.8.4. Consequences of traffic burstiness

In order to illustrate our method, we developed a COMNET simulation model to measure the consequences of bursty traffic on network links, message delays, routers' input buffers, and the number of dropped packets due to the aggregated traffic of large number of users. The model implements the Hurst parameter as it has been described in Section 3. We repeated the simulation for 6000 sec, 16000 sec and 18000 sec to allow infrequent events to occur a reasonable number of times. We found that the results are very similar in each simulation.

##### Topology of bursty traffic sources

The „Message Source” subnetworks transmit messages as in the baseline model above, but with different burstiness:  $H = 0.95$ ,  $H = 0.75$ ,  $H = 0.55$ , and with fixed size. Initially, we simulate four subnetworks and four users per subnetwork each sending the same volume of data (mean 440 bytes per second) as in the validating model above:

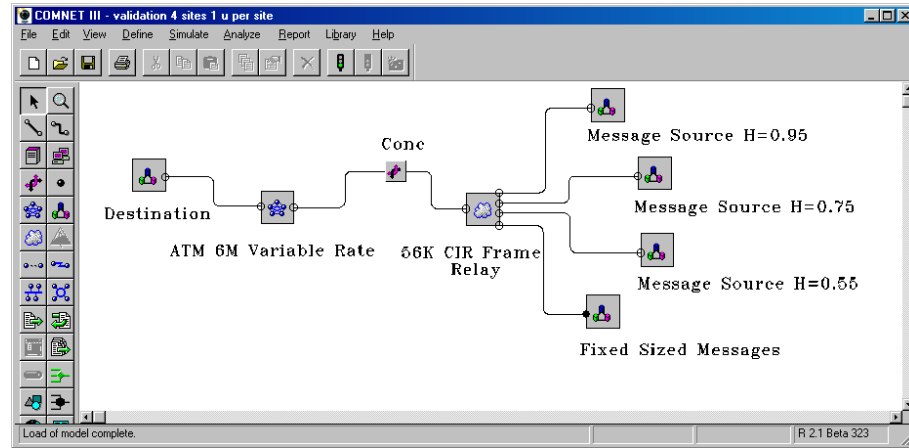


Figure 5.36. Network topology of bursty traffic sources with various Hurst parameters

	Fixed size messages	$H = 0.55$	$H = 0.75$	$H = 0.95$
Average Utilization	0.12	0.13	0.13	0.14
Peak Utilization	0.18	0.48	1	1

Figure 5.37. Simulated average and peak link utilization

	Fixed size messages	$H = 0.55$	$H = 0.75$	$H = 0.95$
Average response time (ms)	75.960	65.61	87.880	311.553
Peak response time (ms)	110.06	3510.9	32418.7	112458.08
Standard deviation	0.470	75.471	716.080	4341.24

Figure 5.38. Response time and burstiness

### Link utilization and message delay

First, we are going to measure and illustrate the extremely high peaks in frame relay link utilization and message delay. The model traffic is generated with message sizes determined by various Hurst parameters and fixed size messages for comparison. The COMNET modeling tool has a trace option to capture its own model generated traffic. It has been verified that for the model-generated traffic flows with various Hurst parameters the Benoit package computed similar Hurst parameters for the captured traces.

The following table shows the simulated average and peak link utilization of the different cases. The utilization is expressed in the  $[0, 1]$  scale not in percentages:

The enclosed charts in Appendix A clearly demonstrate that even though the average link utilization is almost identical, the frequency and the size of the peaks increase with the burstiness, causing cell drops in routers and switches. We received the following results for response time measurements:

The charts in the Appendix A graphically illustrate the relation between response times and various Hurst parameters.

	Fixed size message	$H = 0.55$	$H = 0.75$	$H = 0.95$
Packets accepted	13282	12038	12068	12622
Packets blocked	1687	3146	3369	7250
Average buffer use in bytes	56000858	61001835	62058222	763510495

**Figure 5.39.** Relation between the number of cells dropped and burstiness

#### Input buffer level for large number of users

We also measured the number of cells dropped at a router's input buffer in the ATM network due to surge of bursty cells. We simulated the aggregated traffic of approximately 600 users each sending the same number of bytes in a second as in the measured real network. The number of blocked packets is summarized in the following table:

#### 5.8.5. Conclusion

The paper presented a discrete event simulation methodology to measure various network performance parameters while transmitting bursty traffic. It has been proved in recent studies that combining bursty data streams will also produce bursty combined data flow. The studies imply that the methods and models used in traditional network design require modifications. We categorize our modeling methodology as a structural model contrary to a black box model. Structural models focus on the environment in which the models' data was collected; i.e., the complex hierarchies of network components that make up today's communications systems. Although black box models are useful in other contexts, they are not easy to use in designing, managing and controlling today's networks. We implemented a well-known model, the M/Pareto model within the discrete event simulation package COMNET that allows the analysis of the negative impact of self-similar traffic on not just one single queue, but on the overall performance of various interrelated network components as well. Using real network traces, we built and validated a model by which we could measure and graphically illustrate the impact of bursty traffic on link utilization, message delays, and buffer performance of Frame Relay and ATM networks. We illustrated that increasing burstiness results in extremely high link utilization, response time, and dropped packets, and measured the various performance parameters by simulation.

The choice of the package emphasizes the need for integrated tools that could be useful not just for theoreticians, but also for network engineers and designers. Our paper intends to narrow the gap between existing, well-known theoretical results and their applicability in everyday, practical network analysis and modeling. It is highly desirable that appropriate traffic models should be accessible from measuring, monitoring, and controlling tools. Our model can help network designers and engineers, the ultimate users of traffic modeling, understand the dynamic nature of network traffic and assist them in their everyday practice.

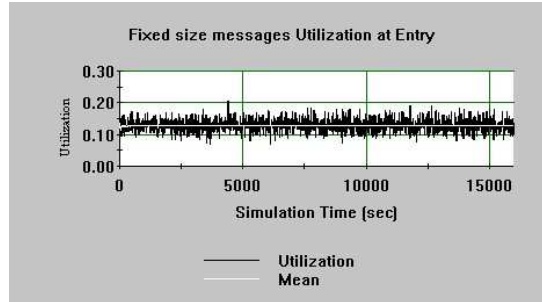


Figure 5.40. Utilization of the frame relay link for fixed size messages

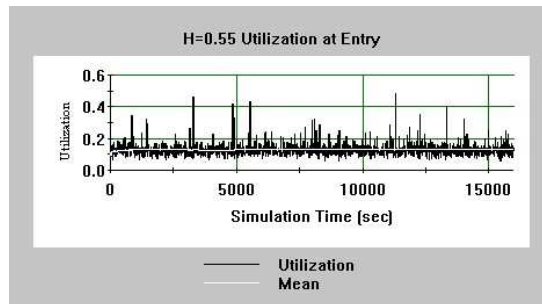


Figure 5.41. Utilization of the frame relay link for Hurst parameter  $H = 0.55$

## 5.9. Appendix A.

### 5.9.1. Measurements for link utilization

The following charts demonstrate that even though the average link utilization for the various Hurst parameters is almost identical, the frequency and the size of the peaks increase with the burstiness, causing cell drops in routers and switches. The utilization is expressed in the  $[0, 1]$  scale not in percentages:

### 5.9.2. Measurements for message delays

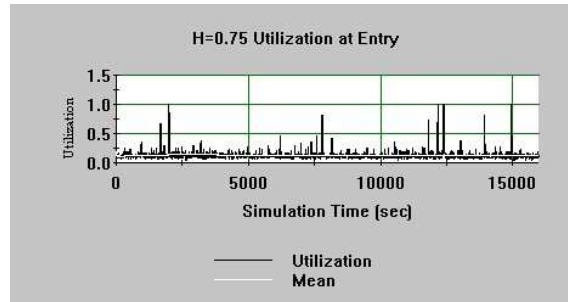
Charts 5.44–5.47 illustrate the relation between response time and various Hurst parameters:

#### Exercises

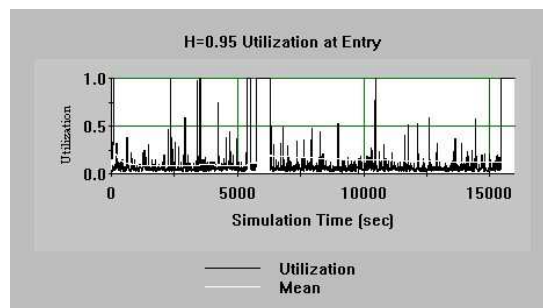
**5.9-1** Name some attributes, events, activities and state variables that belong to the following concepts:

- Server
- Client
- Ethernet

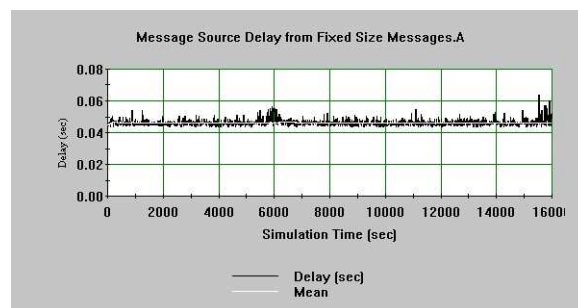




**Figure 5.42.** Utilization of the Frame relay link for Hurst parameter  $H = 0.75$  (Peaks are higher):



**Figure 5.43.** Utilization of the frame relay link for Hurst parameter  $H = 0.95$  (Many high peaks)



**Figure 5.44.** Message delay for fixed size message

- Packet switched network
- Call set up in cellular mobile network
- TCP *Slow start algorithm*

**5.9-2** Read the article about the application of the network simulation and write a report

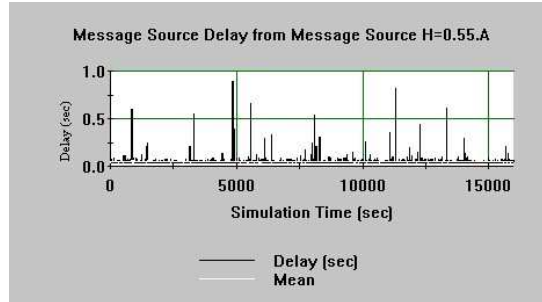


Figure 5.45. Message delay for  $H = 0.55$  (longer response time peaks)

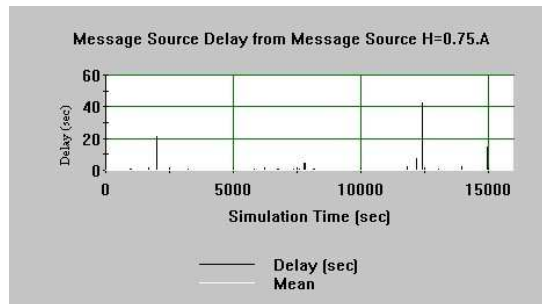


Figure 5.46. Message delay for  $H = 0.75$  (Long response time peaks)

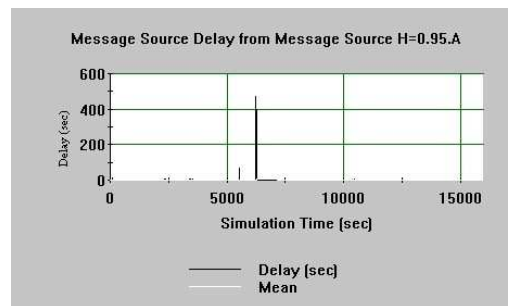


Figure 5.47. Message delay for  $H = 0.95$  (Extremely long response time peak)

about how the article approaches the validation of the model.

**5.9-3** For this exercise it is presupposed that there is a network analyser software (e.g., Lanalyzer for Windows or any similar) available to analyse the network traffic. We use the mentioned software thereafter.

- Let's begin to transfer a file between a client and a server on the LAN. Observe the detailed statistics of the utilization of the datalink and the number of packets per second

then save the diagram.

- Read the *Capturing and analyzing Packet* chapter in the Help of Lanalyzer.
- Examine the packets between the client and the server during the file transfer.
- Save the captured trace information about the packets in .csv format. Analyse this file using spreadsheet manager. Note if there are any too long time intervals between two packets, too many bad packets, etc. in the unusual protocol events.

**5.9-4** In this exercise we examine the network analyzing and baseline maker functions of the Sniffer. The baseline defines the activities that characterize the network. By being familiar with this we can recognize the non-normal operation. This can be caused by a problem or the growing of the network. Baseline data has to be collected in case of typical network operation. For statistics like bandwidth utilization and number of packets per second we need to make a chart that illustrates the information in a given time interval. This chart is needed because sampled data of a too short time interval can be false. After adding one or more network component a new baseline should be made, so that later the activities before and after the expansion can be compared. The collected data can be exported to be used in spreadsheet managers and modeling tools, that provides further analysing possibilities and is helpful in handling gathered data.

Sniffer is a very effective network analyzing tool. It has several integrated functions.

- Gathering traffic-trace information for detailed analysis.
- Problem diagnosis with Expert Analyzer.
- Real-time monitoring of the network activities.
- Collecting detailed error and utilization statistics of nodes, dialogues or any parts of the network.
- Storing the previous utilization and fault information for baseline analysis.
- When a problem occurs it creates visible or audible alert notifications for the administrators.
- For traffic simulation monitoring of the network with active devices, measuring the response time, hop counting and faults detection.
- The History Samples option of the Monitor menu allows us to record the network activities within a given time interval. This data can be applied for baseline creation that helps to set some thresholds. In case of non-normal operation by exceeding these thresholds some alerts are triggered. Furthermore this data is useful to determine the long-period alteration of the network load, therefore network expansions can be planned forward.
- Maximum 10 of network activities can be monitored simultaneously. Multiple statistics can be started for a given activity, accordingly short-period and long-period tendencies can be analysed concurrently. Network activities that are available for previous statistics depends on the adapter selected in the Adapter dialogue box. For example in case of a token ring network the samples of different token ring frame types (e.g, Beacon frames), in Frame Relay networks the samples of different Frame Relay frame types (e.g, LMI frames) can be observed. The available events depend on the adapter.

#### Practices:

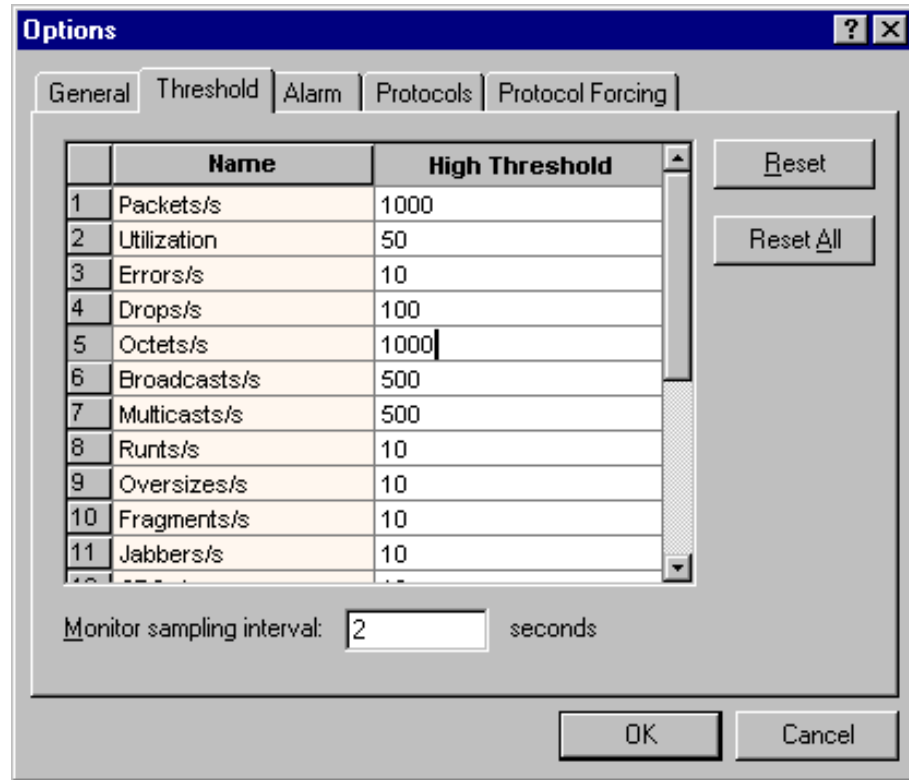
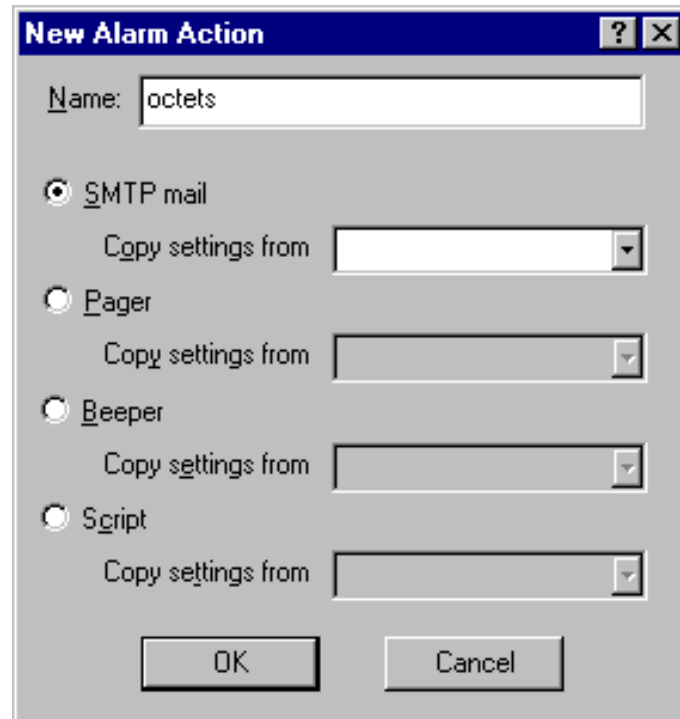


Figure 5.48. Settings

- Set up a filter (Capture/Define filter) between your PC and a remote Workstation to sample the IP traffic.
- Set up the following at the Monitor/History Samples/Multiple History: Octets/sec, Utilization, Packets/sec, Collisions/sec and Broadcasts/sec.
- Configure sample interval for 1 sec. (right click on the Multiple icon and Properties/Sample).
- Start network monitoring (right click on the Multiple icon and Start Sample).
- Simulate a typical network traffic, e.g, download a large file from a server.
- Record the „Multiple History” during this period of time. This can be considered as baseline.
- Set the value of the Octets/sec tenfold of the baseline value at the Tools/Options/MAC/Threshold. Define an alert for the Octets/sec: When this threshold exceeded, a message will be sent to our email address. On Figure 5.48 we suppose that this threshold is 1,000.
- Alerts can be defined as shown in Figure 5.49.
- Set the SMTP server to its own local mail server (Figure 5.50).



The "New Alarm Action" dialog box has a title bar with a question mark and a close button. It contains a "Name:" label followed by a text box containing "octets". Below this are four radio button options: "SMTP mail" (selected), "Pager", "Beeper", and "Script". Each option is followed by a "Copy settings from" label and a dropdown menu. At the bottom are "OK" and "Cancel" buttons.

**New Alarm Action**

Name: octets

☒ SMTP mail  
Copy settings from [dropdown]

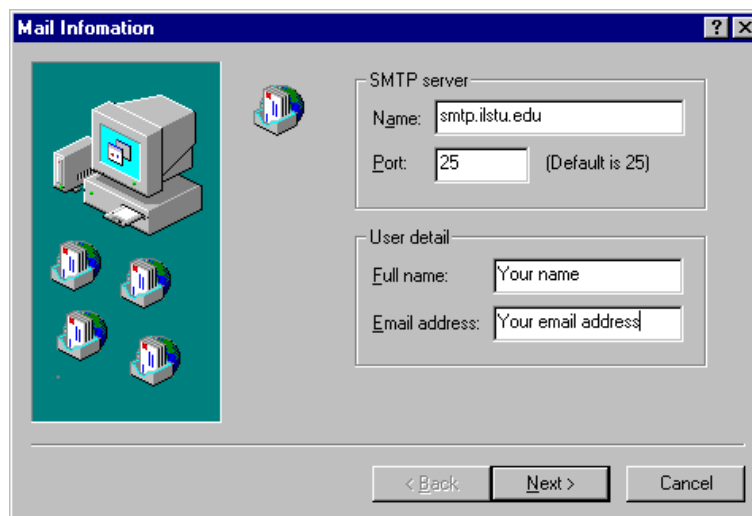
☐ Pager  
Copy settings from [dropdown]

☐ Beeper  
Copy settings from [dropdown]

☐ Script  
Copy settings from [dropdown]

OK Cancel

Figure 5.49. New alert action



The "Mail Information" dialog box has a title bar with a question mark and a close button. On the left is a graphic of a computer monitor and four small icons. On the right, there are two sections: "SMTP server" and "User detail". The "SMTP server" section has "Name:" (smtp.ilstu.edu) and "Port:" (25, with a note "(Default is 25)"). The "User detail" section has "Full name:" (Your name) and "Email address:" (Your email address). At the bottom are "< Back", "Next >", and "Cancel" buttons.

**Mail Information**

SMTP server

Name: smtp.ilstu.edu

Port: 25 (Default is 25)

User detail

Full name: Your name

Email address: Your email address

< Back Next > Cancel

Figure 5.50. Mailing information

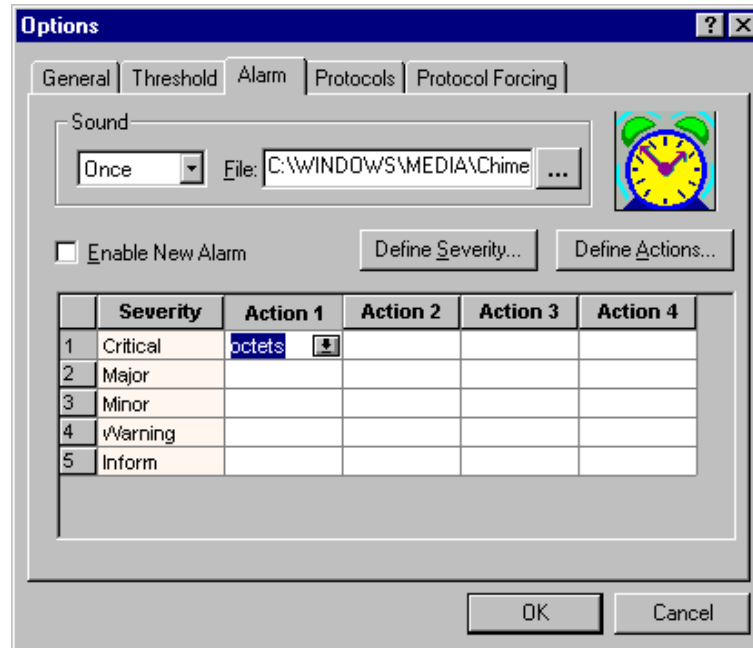


Figure 5.51. Settings

- Set the Severity of the problem to Critical (Figure 5.51).
- Collect tracing information (Capture/Start) about network traffic during file download.
- Stop capture after finished downloading (Capture/Stop then Display).
- Analyse the packets' TCP/IP layers with the Expert Decode option.
- Check the „Alert message” received from Sniffer Pro. Probably a similar message will be arrived that includes the octets/sec threshold exceeded:

From: ...  
 Subject: Octets/s: current value = 22086, High Threshold = 9000  
 To: ...

This event occurred on ...

Save the following files:

- The „Baseline screens”
- The Baseline Multiple History.csv file
- The „alarm e-mail”.

**5.9-5** The goal of this practice is to build and validate a baseline model using a network modeling tool. It's supposed that a modeling tool such as COMNET or OPNET is available for the modeler.

First collect response time statistics by pinging a remote computer. The ping command measures the time required for a packet to take a round trip between the client and the server. A possible format of the command is the following: `ping hostname -n x -l y -w z > filename` where „x” is the number of packet to be sent, „y” is the packet length in bytes, „z” is the time value and „filename” is the name of the file that includes the collected statistics.

For example the ping 138.87.169.13 -n 5 -l 64 > c: ping.txt command results the following file:

```
Pinging 138.87.169.13 with 64 bytes of data:
Reply from 138.87.169.13: bytes=64 time=178ms TTL=124
Reply from 138.87.169.13: bytes=64 time=133ms TTL=124
Reply from 138.87.169.13: bytes=64 time=130ms TTL=124
Reply from 138.87.169.13: bytes=64 time=127ms TTL=124
Reply from 138.87.169.13: bytes=64 time=127ms TTL=124
```

- Create a histogram for these time values and the sequence number of the packets by using a spreadsheet manager.
- Create a histogram about the number of responses and the response times.
- Create the cumulative density function of the response times indicating the details at the tail of the distribution.
- Create the baseline model of the transfers. Define the traffic attributes by the density function created in the previous step.
- Validate the model.
- How much is the link utilization in case of messages with length of 32 and 64 bytes?

**5.9-6** It is supposed that a modeling tool (e.g., COMNET, OPNET, etc.) is available for the modeler. In this practice we intend to determine the place of some frequently accessed image file in a lab. The prognosis says that the addition of clients next year will triple the usage of these image files. These files can be stored on the server or on the client workstation. We prefer storing them on a server for easier administration. We will create a baseline model of the current network, we measure the link-utilization caused by the file transfers. Furthermore we validate the model with the current traffic attributes. By scaling the traffic we can create a forecast about the link- utilization in case of trippled traffic after the addition of the new clients.

- Create the topology of the baseline model.
- Capture traffic trace information during the transfer and import them.
- Run and validate the model (The number of transferred messages in the model must be equal to the number in the trace file, the time of simulation must be equal to the sum of the Interpacket Times and the link utilization must be equal to the average utilization during capture).
- Print reports about the number of transferred messages, the message delays, the link utilization of the protocols and the total utilization of the link.
- Let's triple the traffic.

- Print reports about the number of transferred messages, the message delay, the link utilization of the protocols and the total utilization of the link.
- If the link-utilization is under the baseline threshold then we leave the images on the server otherwise we move them to the workstations.
- Question: Where should they be stored, on the cliens or on the server? Kérdés: Hol érdemesebb tárolni ezeket, a klienseken vagy a kiszolgálón?

**5.9-7** The aim of this practice to compare the performance of the shared and the switched Ethernet. It can be shown that transformation of the shared Ethernet to switched one is only reasonable if the number of collisions exceeds a given threshold.

a. Create the model of a client/server application that uses shared Ethernet LAN. The model includes 10Base5 Ethernet that connects one Web server and three group of workstations. Each group has three PCs, furthermore each group has a source that generates „Web Request” messages. The Web server application of the server responds to them. Each „Web Request” generates traffic toward the server. When the „Web Request” message is received by the server a „Web Response” message is generated and sent to the appropriate client.

- Each „Web Request” means a message with 10,000 bytes of length sent by the source to the Web Server every Exp(5) second. Set the text of the message to „Web Request”.
- The Web server sends back a message with the „Web Response” text. The size of the message varies between 10,000 and 100,000 bytes that determined by the Geo(10000, 100000) distribution. The server responds only to the received „Web Request” messages. Set the reply message to „Web Response”.
- For the rest of the parameters use the default values.
- Select the „Channel Utilization” and the („Collision Stats”) at the („Links Reports”).
- Select the „Message Delay” at the („Message + Response Source Report”).
- Run the simulation for 100 seconds. Animation option can be set. Futassuk a szimulációt 100 másodpercig. Megadhatjuk az animáció opciót is.
- Print the report that shows the „Link Utilization”, the „Collision Statistics” and the report about the message delays between the sources of the traffic.

b. In order to reduce the response time transform the shared LAN to switched LAN. By keeping the clien/server parameters unchanged, deploy an Ethernet switch between the cliens and the server. (The server is connected to the switch with full duplex 10Base5 connection.)

- Print the report of „Link Utilization” and „Collision Statistics”, furthermore the report about the message delays between the sources of the traffic.

c. For all of the two models change the 10Base5 connections to 10BaseT. Unlike the previous situations we will experience a non-equivalent improvement of the response times. We have to give explanation.

**5.9-8** A part of a corporate LAN consists of two subnets. Each of them serves a department. One operates according to IEEE 802.3 CSMA/CD 10BaseT Ethernet standard, while the other communicates with IEEE 802.5 16Mbps Token Ring standard. The two subnets are connected with a Cisco 2500 series router. The Ethernet LAN includes 10 PCs, one of them



functions as a dedicated mail server for all the two departments. The Token Ring LAN includes 10 PC's as well, one of them operates as a file server for the departments.

The corporation plans to engage employees for both departments. Although the current network configuration won't be able to serve the new employees, the corporation has no method to measure the network utilization and its latency. Before engaging the new employees the corporation would like to estimate these current baseline levels. Employees have already complained about the slowness of download from the file server.

According to a survey, most of the shared traffic flown through the LAN originates from the following sources: electronic mailing, file transfers of applications and voice based messaging systems (Leaders can send voice messages to their employees). The conversations with the employees and the estimate of the average size of the messages provides the base for the statistical description of the message parameters.

E-mailing is used by all employees in both departments. The interviews revealed that the time interval of the mail sending can be characterized with an Exponential distribution. The size of the mails can be described with an Uniform distribution accordingly the mail size is between 500 and 2,000 bytes. All of the emails are transferred to the email server located in the Ethernet LAN, where they are be stored in the appropriate user's mailbox.

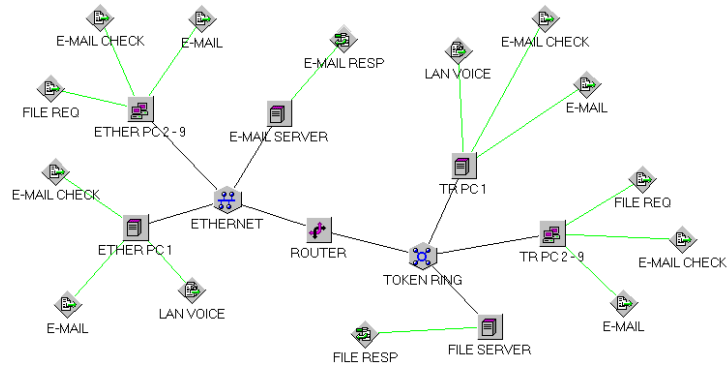
The users are able to read messages by requesting them from the email server. The checking of the mailbox can be characterized with a Poisson distribution whose mean value is 900 seconds. The size of the messages used for this transaction is 60 bytes. When a user wants to download an email, the server reads the mailbox file that belongs to the user and transfers the requested mail to the user's PC. The time required to read the files and to process the messages inside can be described with an Uniform distribution that gathers its value from the interval of 3 and 5 seconds. The size of the mails can be described with a Normal distribution whose mean value is 40,000 bytes and standard deviation is 10,000 bytes.

Both departments have 8 employees, each of them has their own computer, furthermore they download files from the file server. Arrival interval of these requests can be described as an Exponential distribution with a mean value of 900 ms. The requests' size follows Uniform distribution, with a minimum of 10 bytes minimum and a maximum of 20 bytes. The requests are only sent to the file server located in the Token Ring network. When a request arrives to the server, it read the requested file and send to the PC. This processing results in a very low latency. The size of the files can be described with a Normal distribution whose mean value is 20,000 bytes and standard deviation is 25,000 bytes.

Voice-based messaging used only by the heads of the two departments, sending such messages only to theirs employees located in the same department. The sender application makes connection to the employee's PC. After successful connection the message will be transferred. The size of these messages can be described by Normal distribution with a mean value of 50,000 bytes and a standard deviation of 1,200 bytes. Arrival interval can be described with a Normal distribution whose mean value is 1,000 seconds and standard deviation is 10 bytes.

TCP/IP is used by all message sources, and the estimated time of packet construction is 0.01 ms.

The topology of the network must be similar to the one in COMNET, Figure 5.52.



**Figure 5.52.** Network Topology

The following reports can be used for the simulation:

- Link Reports: Channel Utilization and Collision Statistics for each link.
- Node Reports: Number of incoming messages for the node.
- Message and Response Reports: The delay of the messages for each node.
- Session Source Reports: Message delay for each node.

By running the model, a much higher response time will be observed at the file server. What type of solution can be proposed to reduce the response time when the quality of service level requires a lower response time? Is it a good idea to set up a second file server on the LAN? What else can be modified?

## Chapter notes

Law and Kelton's monography [?] provides a good overview about the network systems. About the classification of computer networks we propose two monography released in hungarian as well, whose authors are Sima, Fountain és Kacsuk [?], and Tanenbaum [?]

Concerning the basis of probability the books of András, Prékopa [?] and Alfréd, Rényi [?] are recommended. We have summarized the most common statistical distribution by Banks' book [?]. The review of COMNET simulation modeling tool used to depict the density functions can be found in two publications of CACI [?, ?].

Concerning the background of mathematical simulation Katai's article [?], about spool theory the book of Kleinrock [?] and the online course material of János Sztrik [?] are recommended.

The definition of channel capacity can be found in the dictionaries that are available on the Internet [?, ?]. Information and code theory related details can be found in Jones and Jones' book [?].

Taqqu and Co. [?, ?] deal with long-range dependency.

Figure 5.1 that describes the estimations of the most common distributions in network modeling is taken from the book of Banks, Carson és Nelson könyvéből [?].

The OPNET software and its documentation can be downloaded from the address found in [?]. Each phase of simulation is discussed fully in this document.

The effect of traffic burstiness is analysed on the basis of Tibor Gyires's article cite-Gyires02.

Leland and Co. [?, ?], Crovella and Bestavros [4] report measurements about network traffic.

The self-similarity of networks is dealt by Erramilli, Narayan and Willinger [7], Willinger and Co. [?], and Beran [?]. Mandelbrot [?], Paxson és Floyd [?], furthermore the long-range dependent processes was studied by Mandelbrot and van Ness [?].

Traffic routing models can be found in the following publications: [2, 9, ?, ?, ?, ?, ?, ?].

Figure 5.22 is from the article of Listanti and Eramo [?]. Forgalmi adatokat tartalmaz [3, 6, 8, ?].

Long-range dependency was analysed by Addie, Zukerman and Neame [1], Duffield and O'Connell [5], and Narayan and Willinger [7].

The expression of *black box* modeling was introduced by Willinger and Paxson [?] in 1997.

Information about the principle of *Ockham's Razor* can be found on the web page of Francis Heylighen [?]. More information about Sniffer is on Network Associates' web site [?].

Willinger, Taqqu, Sherman and Wilson [?] analyse a structural model. Crovella and Bestavros [4] analysed the traffic of World Wide Web.

The effect of burstiness to network congestion is dealt by Neuts [?], and Molnár, Vidács and Nilsson [?].

The effect of the Hurst parameter is studied by Addie, Zukerman and Neame [1].

The Benoit-package can be downloaded from the Internet [?].

The Pareto-model is analysed by Addie, Zukerman and Neame [1].

# Bibliography

- [1] R. G. Addie, M. Zukerman, T. Neame. Broadband traffic modeling: Simple solutions to hard problems. *IEEE Communications Magazine*, (8):88–95, 1998. [219](#)
- [2] D. Anick, D. Mitra, M. Sondhi. Stochastic theory of a data handling system with multiple sources. *The Bell System Technical Journal*, 61:1871–1894, 1982. [219](#)
- [3] J. Beran, R. Sherman, M. Taqqu, W. Willinger. Long-range dependence in variable-bit-rate video traffic. *IEEE Transactions on Communications*, 43:1566–1579, 1995. [219](#)
- [4] M. Crovella, A. Bestavros. Self-similarity in world wide web traffic: Evidence and possible causes. *IEEE/ACM Transactions on Networking*, 5(6):835–846, 1997. [219](#)
- [5] N. Duffield, N. Oconnell. Large deviation and overflow probabilities for the general single-server queue, with applications. *Mathematical Proceedings of the Cambridge Philosophical Society*, 118:363–374, 1995. [219](#)
- [6] D. Duffy, A. McIntosh, M. Rosenstein, W. Willinger. Statistical analysis of ccsn/ss7 traffic data from working ccs subnetworks. *IEEE Journal on Selected Areas Communications*, 12:544–551, 1994. [219](#)
- [7] A. Erramilli, O. Narayan, W. Willinger. Experimental queueing analysis with long-range dependent packet-traffic. *IEEE/ACM Transactions on Networking*, 4(2):209–223, 1996. [219](#)
- [8] R. Gusella. A measurement study of diskless workstation traffic on an ethernet. *IEEE Transactions on Communications*, 38:1557–1568, 1990. [219](#)
- [9] H. Hefles, D. Lucantoni. A markov modulated characterization of packetized voice and data traffic and related statistical multiplexer performance. *IEEE Journal on Selected Areas in Communication*, 4:856–868, 1986. [219](#)

# Subject Index

## A, Á

Analysis Tool, [178](#)  
asymptotically self-similar process, [193](#)

## B

Bandwidth, [166](#)  
Bluetooth, [165](#)  
Burstiness, [167](#)  
Burst Interarrival Time, [197](#)

## C

COMNET, [170](#)

## D

Data Exchange Chart, [182](#)  
Dropped packet rate, [168](#)

## E, É

emulation, [163](#)

## F

Fiber Distributed Data Interface (FDDI), [165](#)  
Frame size, [168](#)

## G

Global Positioning System, [165](#)

## H

Hurst-parameter, [194](#)

## I, Í

Internet, [164](#)  
internetwork, [164](#)

## L

Latency, [166](#)  
Link capacity, [166](#)  
long-range dependency, [168](#)

long-range dependent process, [193](#)

## M

Management Information Base, [187](#)  
MDLC, [186](#)  
MIB, *see* Management Information Base  
model  
    continuous, [162](#)  
    deterministic, [161](#)  
    discrete, [162](#)  
    dynamic, [161](#)  
    static, [161](#)  
    stochastic, [161](#)  
Model Development Life Cycle, [187](#)  
modell  
    batched Markovian arrival, [194](#)  
    fluid-flow, [194](#)  
    Markovian arrival processes, [194](#)  
    Markov-modulált Poisson-processes, [194](#)

## N

Network Simulation, [161](#)  
Node Editor, [177](#)

## O, Ó

OPNET, [219](#)

## P

packet, [165](#)  
Packet Interarrival Time, [197](#)  
Performance Metrics, [166](#)  
permanent virtual circuit (PVC), [189](#)  
Process Editor, [177](#)  
Project Editor, [177](#)  
Protocol overhead, [167](#)  
PVC, *see* permanent virtual circuit

## R

Remote Monitoring, [187](#)  
Response time, [166](#)  
RMON, *see* Remote Monitoring  
Routing protocols, [167](#)

Run length, [190](#)

**S**

self-similarity, [193](#)

self-similar process, [193](#)

short-range dependent process, [193](#)

Simple Network Management Protocol, [187](#)

slow start algorithm, [168](#)

Sniffer, [185](#)

SNMP, *see* Simple Network Management Protocol

SNR, *see* signal to noise ratio

**T**

The Hurst parameter, [197](#)

Traffic engineering, [167](#)

Transaction Interarrival Time, [197](#)

Transform-Expand-Sample models, [194](#)

**W**

warm-up period, [190](#)

# Name index

## A, Á

Addie, R. G., [220](#)  
Anick, D., [220](#)

## B

Beran, J., [219](#), [220](#)  
Bestavros, Azer, [219](#), [220](#)

## C

Crovella, Mark E., [219](#), [220](#)

## D

Duffield, N., [219](#), [220](#)  
Duffy, D. E., [220](#)

## E, É

Erramilli, A., [219](#), [220](#)

## F

Floyd, S., [219](#)  
Fountain, Terence, [218](#)

## G

Gusella, Riccardo, [220](#)

## GY

Gyires, Tibor, [219](#)

## H

Hefles, H., [220](#)  
Heylighen, Francis, [219](#)  
Hurst, H. E., [194](#)

## J

Jones, Gareth A., [219](#)  
Jones, J. Mary, [219](#)

## K

Kacsuk, Péter, [218](#)

Káta, Imre, [218](#)

Kleinrock, Leonard, [218](#)

## L

Leland, W., [219](#)  
Lucantoni, D., [220](#)

## M

Mandelbrot, Benoit, [219](#)  
†Markov, Andrej Andrejevics, [193](#)  
McIntosh, A. A., [220](#)  
Mitra, D., [220](#)  
Molnár, Sándor, [219](#)

## N

Narayan, O., [219](#), [220](#)  
Neame, T., [220](#)  
Neuts, Marcel F., [219](#)  
Nilsson, Arne A., [219](#)

## O, Ó

OConnell, N., [219](#), [220](#)

## P

Pareto, Vilfredo, [172](#)  
Paxson, V., [219](#)  
Prékopa, András, [218](#)

## R

Rényi, Alfréd, [218](#)  
Rosenstein, M., [220](#)

## S

Sherman, R., [220](#)  
Sima, Dezső, [218](#)  
Sondhi, M. M., [220](#)

## SZ

Sztrik, János, [218](#)

**T**

Tanenbaum, Andrew S., [218](#)  
Taqq, Murad S., [219](#), [220](#)

**V**

Van Ness, John W., [219](#)  
Vidács, Attila, [219](#)

**W**

Willinger, W., [219](#), [220](#)  
Wilson, D., [219](#)

**Z**

Zukerman, M., [220](#)



# Contents

<b>5. Network Simulation (Tibor Gyires)</b>	<b>161</b>
5.1. Types of simulation	161
5.1.1. Systems, models, discrete-event simulation	161
5.2. The need for communications network modeling and simulation	162
5.2.1. Simulation versus emulation	162
5.3. Types of communications networks, modeling constructs	164
5.4. Performance targets for simulation purposes	166
5.5. Traffic characterization	168
5.6. Simulation modeling systems	177
5.6.1. Data collection tools and network analysers	177
5.6.2. Model specification	177
5.6.3. Data collection and simulation	177
5.6.4. Analysis	178
5.6.5. Network Analysers – Application Characterization Environment (ACE)	180
5.6.6. Sniffer	185
5.7. Model Development Life Cycle (MDLC)	186
Identification of the topology and network components	187
Data collection	187
Construction and validation of the baseline model. Perform network simulation studies using the baseline	189
Creation of the application model using the details of the traffic generated by the applications	191
Integration of the application and baseline models and completion of simulation studies	191
Further data gathering as the network grows and as we know more about the applications	192
5.8. Methodology for modeling the impact of traffic burstiness on high-speed networks	193
Background	193
Definition of self-similarity	193
Definition of long-range dependency	193
Traffic models	194

	Black box vs. structural models . . . . .	195
	Implications of burstiness on high-speed networks . . . . .	197
5.8.1.	Model parameters . . . . .	197
	The Hurst parameter . . . . .	197
	The M/Pareto traffic model and the Hurst parameter . . . . .	198
5.8.2.	Implementation of the Hurst parameter in the COMNET modeling tool . . . . .	198
	Traffic measurements . . . . .	199
5.8.3.	Validation of the baseline model . . . . .	200
5.8.4.	Consequences of traffic burstiness . . . . .	205
	Topology of bursty traffic sources . . . . .	205
	Link utilization and message delay . . . . .	206
	Input buffer level for large number of users . . . . .	207
5.8.5.	Conclusion . . . . .	207
5.9.	Appendix A. . . . .	208
5.9.1.	Measurements for link utilization . . . . .	208
5.9.2.	Measurements for message delays . . . . .	208
	<b>Bibliography . . . . .</b>	<b>220</b>
	<b>Subject Index . . . . .</b>	<b>221</b>
	<b>Name index . . . . .</b>	<b>223</b>