Modeling of Primary and Secondary Load in the Internet

B. E. Wolfinger*, M. Zaddach*, G. Bai**, K. D. Heidtmann*

*Dept. of Computer Science, Telecommunication and Computer Networks Division (TKRN), Hamburg University

**Institute for Media Communication, GMD - German National Research Center for Information Technology, Sankt Augustin

Report No. 227

November 2000

Abstract

A particular challenge, when trying to analyse and predict the behaviour of subnetworks of the global Internet, refers to the task of elaborating a sufficiently realistic workload characterization, e.g., by means of workload modeling. In particular, it is necessary to specify (work)load at different system interfaces.
This report presents a generalized proceeding for load modeling including a set of formal methods for load specification. At first the basic proceeding is applied by way of example to the modeling of primary load, i.e. load at an interface close to end-users, whereby we focus on video sources. We then tackle the challenging problem of characterizing secondary load, i.e. load as it is occuring at a lower layer interface within a protocol/service hierarchy, and for this purpose we suggest a new approach for analytical modeling of load transformations as they are typical for communication networks. Modeling load transformations again is exemplified by means of a comprehensive case study assuming video sources and considering some load transformation corresponding to the impact which a UDP/IP protocol hierarchy would have on some offered primary load. A set of detailed measurements proves that our approach to model load transformations can indeed be used to prognosticate a highly valid characterization of secondary load in case of Internet or Intranet configurations.

Kurzfassung

Eine der wesentlichen Herausforderungen, die sich bei der Analyse und Vorhersage des Verhaltens von Teilnetzen des Internets ergeben, stellt die Entwicklung einer hinreichend realistischen Lastcharakterisierung dar, z.B. mittels Lastmodellierung. Im speziellen ergibt sich die Notwendigkeit, die (Arbeits-)Last an unterschiedlichen Systemschnittstellen zu spezifizieren.
Dieser Bericht präsentiert eine allgemeine Vorgehensweise für die Lastmodellierung, einschließlich einer Reihe von formalen Methoden zur Lastspezifikation. Zunächst wird diese Vorgehensweise exemplarisch angewandt, um eine Primärlast, d.h. diejenige Last an einer benutzernahen Schnittstelle, zu modellieren, wobei sich die Studien auf Videoquellen beziehen. Daran anschließend beschäftigen wir uns mit dem Problem der Charakterisierung von Sekundärlasten, d.h. von Lasten, wie sie an einer in der Protokoll-/Diensthierarchie niedrigen Schnittstelle auftreten. Zu diesem Zwecke schlagen wir einen neuen Ansatz der analytischen Modellierung von Lasttransformationen vor, wie sie in Kommunikationsnetzen typisch sind. Auch die Modellierung dieser Lasttransformationen wird wiederum anhand umfangreicher Beispielstudien exemplarisch vorgestellt. Dazu werden Lasttransformationen herangezogen, welche den Einfluß einer UDP/IP-Protokollhierarchie auf eine benutzerseitig übergebene Primärlast widerspiegeln. Detaillierte Messungen untermauern, daß unser Ansatz der Modellierung von Lasttransformationen in der Tat eine sehr valide Prognose von Sekundärlasten für den Bereich des Internet sowie für Intranetumgebungen gestattet.

Contents

List of Figures

1 Workload Characterization and Load Modeling for Communication Systems and Computer Networks

When modeling service-systems, such as computers, communication networks, database systems etc., it is common practice to clearly separate the requests to be processed/served from the service-system proper which serves the requests. The set of requests to be served over time is denoted as load or workload of the (service-)system. If we apply this view to computer systems the requests to be served may correspond e.g. to user programs to be executed, and in database systems the transactions to be processed may represent the load. In communication or computer networks [34] files, E-mails, audio or video streams, WWW pages etc. could represent the load if we consider an application-oriented interface, whereas at an interface to a packet-switched network the packets to be transmitted would constitute the load which is offered to the network. It is well-known that a sufficiently realistic load characterization, in most cases, is an indispensable prerequisite in order to obtain valid results when predicting the expected system behaviour under a given load (e.g. by modeling or measurement studies) [8, 15].

In the case of communication systems and computer networks analyses, realistic load characterization is required, e.g., when applying analytical models or when executing simulation experiments. Moreover, load characterization is needed for the construction of artificial load generators [14], which e.g. may generate some synthetical load for an existing communication network. Using such load generators communication system behaviour, under various load conditions, can then be investigated by means of measurements.

As we will explain in section 2 in some more detail, load characterization on one hand has to specify the single requests which in their totality represent the overall load and on the other hand it has to specify the stream of requests (arrival process) at a well-defined interface of the service-system considered. Depending on the desired range of use of a load characterization very different degrees of freedom may exist for this characterization. To give an example, we observe that some analytical queueing models of packet-switched networks (such as e.g. Kleinrock's well known model for interactive traffic [20]) may only allow Poisson arrival streams of requests [30], i.e. the packets to be transmitted, in combination with exponentially distributed service-time requirements of requests corresponding to exponentially distributed packet lengths. Therefore, when using such a queueing model load characterization is limited to specifying mean arrival rate of packets (possibly dependent on the source-destination-pair considered) and to specifying mean packet-length. Evidently, distributions for interarrival times and lengths of packets are not part of load characterization in this case as these distributions are predefined by the class of model chosen. A much larger degree of freedom, however, may exist if some detailed load characterization is required in order to produce a highly realistic stream of requests in simulation experiments or as a basis for measurements in real communication networks.

As a consequence of the strongly varying degrees of freedom in load characterization we argue for the following basic requirements to be taken into account when characterizing the

load of computer or communication systems:

(R1) Load characterization should be based on comprehensive load measurements [28].

(R2) Results of load measurements should be coarsened in a direct dependency of the existing degree of freedom in load characterization, which, typically, is strongly dependent on the method of performance evaluation to be applied (e.g. analytical modeling, simulation or measurements).

(R3) One should clearly distinguish between characterization of the single requests which are part of the load and characterization of the timing aspects of the request arrival process.

(R4) A formal description technique should be applied for obtaining a precise specification of requests and request arrival streams.

In this paper we want to tackle the difficult problem of load characterization for the Internet, which by far represents the most important existing computer network (resp. network of networks). In particular, our contribution will introduce a flexible method for characterizing the load which exists at very different interfaces within an IP based protocol hierarchy. We also will apply our method of characterization by way of example.

Section 2 summarizes some of the features and properties of the Internet which make characterization of load for this network and its users an extremely hard problem. Moreover, we indicate the state-of-the-art in load measurements and load modeling of the Internet. In section 3 we are going to introduce a generalized proceeding for load modeling which satisfies above requirements (R1),...,(R4) and can be applied to computer systems as well as to networks. We apply our proceeding suggested during a first case study (section 4) to characterize load as it could occur at an application-oriented interface within an Internet host computer. In the study, by way of example, we will model video streams starting from detailed load measurements and assuming standards for video encoding, such as MPEG and H.261/H.263.

In contrast to TCP based applications, the streams induced by real time applications, typically based on UDP, are not self-regulating (by means of detecting and avoiding congestion). So it is important to characterize the nature of this *stiff* real time streams because of their inelastic and non-adaptive bandwidth requirements.

Our view is that load at an application-oriented interface, which we call *primary load*, is transformed by parts of the communication system into a different load, called *secondary load*, which we can observe at a lower layer interface within the given protocol hierarchy. The topic of load transformation will be discussed in the context of Internet (cf. section 5). In section 6 our second case study will illustrate our new approach to model load transformation processes. We will demonstrate these transformations for the UDP/IP stack because of its importance for real time applications. Thus, we will be able to characterize in a highly realistic manner the secondary load (IP traffic at an LLC interface) as it would be induced by a set of video traffic sources corresponding to the primary load, e.g. within a video server. Our load transformation approach will be successfully validated in section 7 by comparing measured secondary load (as observed in an Internet subsystem) with secondary load as it is prognosticated after transforming a given primary load in the modeling domain. Some of the problems still being unsolved or perhaps even being unsolvable in the long term in characterizing Internet load will be indicated in our conclusions.

2 Special Aspects of Modeling Internet Load

In this paper we are going to use the following definition of load, cf. [5, 19, 25, 36]:
The *(offered) load* or *workload* $L = L(E, S, IF, T)$ denotes the total sequence of requests which is offered by an environment E to a service-system S via a well-defined interface IF during the time-interval T. We call L the load generated by E for S at IF during T.
Let us shortly discuss the strong dependencies of L on E, S, IF and T for the case of a computer network:

- E: all the requests to be served by S are created within the environment as E, in particular, comprises the set of (human) network users as well as the (distributed) applications.
- S: as the service-system is responsible for serving the requests originated by E, the characterization of requests has to specify among others, the resource requirements of each request during its processing/service by S.
- IF: the interface chosen is extremely important as it reflects the decomposition of the computer network and its users into E and S; IF also directly determines the type of requests which can be part of the workload and, moreover, it limits the set of possible sequences of requests.
- T: evidently, the load observed in an existing network is highly dependent on the concrete choice of T, e.g. Sunday vs. Monday, January 1st vs. 2nd, 1-2 pm vs. 1-2 am.

In the following we want to focus on load characterization for the Internet. To begin we want to debate the question why load characterization for the Internet is so much more difficult than characterizing load in networks like closed corporate networks or conventional local-area networks.

So, what are the aspects of the Internet which complicate load characterization ? The following reasons can be identified:

- An enormous amount of users exist already at present (with still exponential growth), therefore traffic observable on IP layer or below typically represents the complex overlay of a large number of traffic streams generated by different users/applications. The traffic is also quite heterogeneous resulting e.g. from data, text, voice and video communications [22, 26].
- User behaviour is highly dynamic, new services are created and a large variety of application-oriented services may be used (e.g. more than 400 application protocols were observed in a 4 day measurement at the University of Saskatchewan [38]).
- User behaviour and therefore also the load generated by end-users is not always observable, e.g. the effort of observation may be too high (as just too many users and endsystems exist) or security mechanisms may restrict observability of system components (a lot of *black box*-subsystems exist in the Internet).
- The state of the network quite often does strongly influence user behaviour (especially if the state of an Internet-subnetwork used corresponds to the often-experienced phase *rien ne va plus*) [3].

Some of the reasons which simplify load characterization for local-area networks (LAN), as opposed to the Internet, are the following:

- typically, all endsystems of a LAN (about 10-1000) are observable in principle;
- less variation tends to exist in user behaviour (e.g. similar daily sequences of operation may exist in an enterprise);
- the number of users, active at any given instant of time is still relatively small and also the overall number of possible users is quite limited;
- last not least the traffic matrix tends to be simplified (e.g. client/server-relationships and a relatively low number of endsystems being addressable at all).

As a direct consequence of our definition of load, load characterization always assumes a well-defined interface. Unfortunately this is quite often not taken into account in existing publications. In the context of Internet we could, in particular, choose the following interfaces for load characterization:

- an application-oriented interface (e.g. interface to services/protocols such as FTP, Telnet, HTTP, SMTP, ...) [4, 21];
- interface to the transport services, based on TCP or UDP, within the endsystems [38];
- the packet interface to IP;
- the LLC interface, e.g. in an Ethernet-based Intranet (i.e. LAN with IP protocol hierarchy).

For most of the Internet interfaces mentioned, a large number of publications exist presenting load measurements for these interfaces. In particular, load measurements for application-oriented interfaces in the Internet have been presented in [4, 38] and comprehensive measurements at the transport layer interface have been published e.g. in [29] covering TCP and in [38] covering UDP. Load measurements referring to IP interface have been summarized in [10, 11, 17].

Load measurements for specific interfaces in communication networks can be used to look for stochastic processes which are able to reflect, with sufficiently good accuracy, the main characteristics of the arrival process observed. This approach to model arrival processes for streams of requests has been applied quite often and with good success in recent publications [12, 13, 35]. In order to supplement the existing approaches to approximate measured Internet load by means of stochastic processes, we argue for a more general proceeding which is not restricted to mathematical modeling but allows us some detailed load characterization and load modeling for simulation models and artificial load generators of IP based networks, too.

Our approach to load characterization and modeling will be presented in detail in the following sections and it will be applied in the context of the Internet. The approach comprises a generalized proceeding for load modeling directly based on load measurements. We suggest to tackle the problem of load modeling for communication networks starting with modeling of the primary load as it exists at an application-oriented interface. We start with

modeling the primary load because this allows us a straight-forward modeling of secondary load in a rather flexible way (e.g. for various choices of secondary load interfaces). Quite often primary load can be observed in a relatively simple and direct manner and load at such application-oriented interfaces typically is created quite independently of the communication network's state. Moreover, a complex primary load can be conceived as a (mutually independent) overlay of elementary single sources of primary load (e.g. single MPEG source in video communications, single file transfer using FTP, transmission of a sequence of PCM samples resulting of a single voice source, etc). Once we have solved the problem of characterizing the single sources of primary load (for various types of sources, cf. examples of source models in the context of ATM networks as introduced in [32]), by means of overlaying single sources we can produce any mix of complex primary load. Thereafter, we can apply our approach for load transformation (cf. section 5), in order to obtain a realistic characterization of the secondary load which exists at some arbitrarily chosen lower layer interface and which is induced by the complex mix of primary load.

For this innovative approach to characterize secondary load, of course, we have to assume some sufficiently detailed knowledge regarding the process of load transformation as it occurs in the communication network. One of the main advantages of our approach to secondary load characterization is, that it is not necessary to measure at the secondary load interface. Therefore, this method of load characterization can also be applied during the design of an innovative communication network (e.g. change of existing protocols) under the assumption that the kind of load transformation in the newly designed network is known sufficiently precisely and that the primary load will be created in the future network in the same way as in the present network. Another important advantage is, that it is not necessary to create a possibly very complex mix of primary load in an existing network - it is sufficient to consider this mix of primary load in the modeling domain.

3 A Generalized Proceeding for Modeling Computer Network Load

In this paper, we want to model traffic loads for the Internet, basing our approach on the generalized procedure proposed in [37] and [19]. The main purpose of this procedure is to present a unified description technique which allows us to formulate models of load (mainly for simulation experiments) for different degrees of detail in modeling and for various kinds of system interfaces. In particular, we want to cover load which reflects requirement of communication resources (however, this is no general restriction). The generalized proceeding for load modeling in the Internet which satisfies the requirements as stated in section 1 is based on the following main steps [36]:

* `STEP 1`: Decomposition of a communication network, a model of which has to be elaborated, into a system model SM and an environment model EM at a well-defined interface

First, we have to decompose the system and its embedding in some environment (comprising, e.g., the set of system users) in a way that we decide where to place the demarcation line between what we consider as system S on one hand and environment E on the other hand.

This decomposition directly provides us the interface IF between S and E. Evidently, IF may correspond to the union of several geographically distributed interfaces $IF_1, IF_2, \ldots, IF_k$ in the real communication system modeled. In the first step, the modeler also has to decide on which requests (passed from E to S) and which reactions (shown by S and observable by E) will be taken into account in load modeling. Moreover, we assume that the environment is mapped on a set of load generating users (possibly corresponding to human end-users or to some load generating application or system processes). For example, for load modeling of Internet: the TCP/IP protocol suite forms the basis for the Internet. And as is typical in the structuring of communication services, this protocol suite is developed in layers, each layer being responsible for a different facet of the communications. The Internet architecture is normally considered to be a 4-layer system, as shown in Figure 1.
Each layer respects one or more protocols for communicating with its peer at the same layer. Although the commonly used name for the Internet's entire protocol suite is TCP/IP, TCP and IP are only two of the many protocols used in combination. For example, Telnet is an application layer protocol, TCP is a transport layer protocol, IP is a network layer protocol, and maybe an Ethernet protocol would operate at the link layer. When an application sends data using TCP, the data is sent via the complete protocol stack and transmitted across the network. An SM/EM interface between Transport (e.g. TCP) Layer and Network (e.g. IP) Layer, at which TCP segments are offered by the Transport Layer to the Network Layer, could be selected. In this case, the Transport Layer could be considered as user (part of the environment E) and the system S would then consist of the Network and the Link Layer. We could draw a nearly identical picture for transmission of UDP data. The only change besides different protocol functionality is that the unit of information that UDP passes to IP is called a UDP datagram.

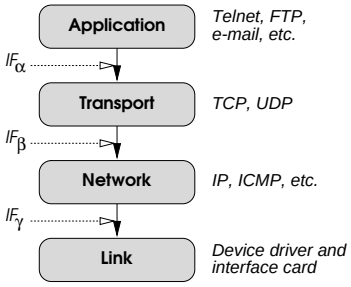


Figure 1: The four layers of the TCP/IP protocol suite

* `STEP 2`: Choice of the level of detail for modeling the system and the environment as well as for the interactions at the SM/EM interface

In order to fulfil our requirements to a generalized load modeling method we choose an object-oriented approach for modeling requests. On one hand, each request has a unique type, i.e. we build disjoint classes of requests; on the other hand, a type-specific set of attributes is associated to each type of requests (with predefined but possibly different domains for attribute values).

In `STEP 2` decisions are taken regarding request types and request attributes (considering, of course, the desired future applicability of the load model to be elaborated). In the Internet, a highly structured communication system, TCP provides a connection-oriented, reliable, byte stream service to the application layer. The term connection-oriented means the two applications using TCP (normally considered a client and a server) must establish a TCP connection with each other before they can exchange data. The TCP/IP traffic may be divided into three phases, that is, connection establishment phase, data transfer phase and connection termination phase. During the connection establishment phase, two types of requests will be generated by a client, i.e.

- CONNECTION_ESTABLISHMENT_request,
- CONNECTION_ACKNOWLEDGE_CLIENT_request,

that is, the client, first, sends a SYN (synchronize sequence numbers flag) segment specifying PN (port number) of the server (destination) that the client wants to connect to, and client (source)'s ISN (initial sequence number); second, the server responds with its own SYN and consumes one sequence number (system reaction). The server also acknowledges the client's SYN by ACKing the client's ISN plus one; third, the client acknowledges this SYN from the server by ACKing the server's ISN plus one. The attributes for the above connection establishment may be the PN of the server (destination address), client's ISN (source address), and so on. In fact, only a subset of all these attributes would typically be relevant for modeling. However, the modeler may also choose the option to select a new attribute, e.g. two attributes in the real world could be summarized as one in the model domain.

* STEP 3: Analysis and description of interactions being possible for a given interface between the communication network (modeled by SM) and its users (modeled by EM)

This step is concerned with specifying the possible sequences of interactions between S and E. This is quite similar to some service specification for a communication service, which also specifies the sequences of service primitives which are possible, in principle, over time; e.g., interactions for load modeling of TCP/IP traffic are resulting from the offering of data to be transmitted or from data arrival at different layers.

* STEP 4: Description of actual interactions between SM and EM (during the interval, within which model behaviour is observed)

This final step, based on the results of earlier steps, is now able to model the overall load generation process during an observation interval T. We assume that load is produced by a set $U_1, U_2, \ldots, U_n$ of load generating users (cf. STEP 1). Moreover, each user is modeled as an individual load generator, creating a stream SR of requests, where SR corresponds to a vector of (time, request)-tupels, i.e. $SR = ((t_1, R_1), (t_2, R_2), (t_3, R_3), \ldots, (t_m, R_m))$ for some positive integer m and t_i is denoting the instant of arrival of R_i at interface IF. We allow the sequences $(t_1, t_2, t_3, \ldots, t_m)$ to be defined by a trace (predetermined arrival process) or by a probability distribution (stochastic arrival process). In the later case, the distribution may have been determined as an approximation of load measurements. Also, types and attribute values for the requests R_i may directly correspond to load measurements.

In the following, we want to introduce a formal description technique to specify load models, which can be considered as some generalization of user-behaviour-graphs [7]. In particular, we wish to formally describe behaviour of single (load-generating) users resp. individual load generators. Our description technique allows to specify single requests and the load generation process over time.

According to STEP 2 of our load modeling method we specify each request by its unique request type and the set of request attributes (with well-defined domains for attribute values). In this paper, we would like to characterize the IP traffic in the Internet. Thus, as mentioned above, during the connection establishment phase, requests could be specified as:

```
CONNECTION_ESTABLISHMENT_request
Begin
 CLIENT_ISN: integer;
 SERVER_PN: integer
end
```

and

```
CONNECTION_ACKNOWLEDGE_CLIENT_request
Begin
 SERVER_SYN: integer
end
```

To reflect the dynamic behaviour of an individual load generator (cf. `STEP 4`), we assume that the load generator may be in one of four macro-states:

- φ_i : idle (initial state)
- φ_a : active (containing the only states, in which requests can be generated)
- φ_b : blocked (waiting for reactions of service-system S)
- φ_t : terminated (no further creation of requests possible).

Macro-states may be refined. They are composed of

- S-states, where a trigger event (external to the load generator) has to be waited for to leave this state. Trigger events are initialization and termination of the load generator or a reaction, indicated by S.
- R-states, where requests are generated in a non-time-consuming manner; these states are left immediately after request generation. R-states are present only as part of macro-state φ_a. Every R-state is responsible for generating requests of exactly one type and the algorithms of how to determine the values for the request attributes are associated to each R-state, too (e.g. by using a trace or a probability distribution). This state determines the type of request which will be generated.
- D-states, where delays are modeled as they may occur before generating the next request (e.g. corresponding to request interarrival times). Times associated to D-states, again, may be determined by traces or probability distributions.
- Transitions between states: they may be deterministic, determined by transition probabilities or by execution of some predefined procedure (the execution of which provides reference to the next state reached).

Figure 2 presents the description technique for specifying the behaviour of an individual load generator (corresponding to an elementary load generating user) by way of example.

In case of TCP/IP traffic, a request of connection establishment causes a state transition $\varphi_i \xrightarrow{\mathsf{R}} \varphi_a$. Another state transition $\varphi_a \xrightarrow{\mathsf{R}} \varphi_b$ will occur when TCP sends a segment. During staying at state φ_b, the automaton maintains a timer, waiting for the other end to acknowledge reception of the segment (reaction from the system). If an acknowledgement is not

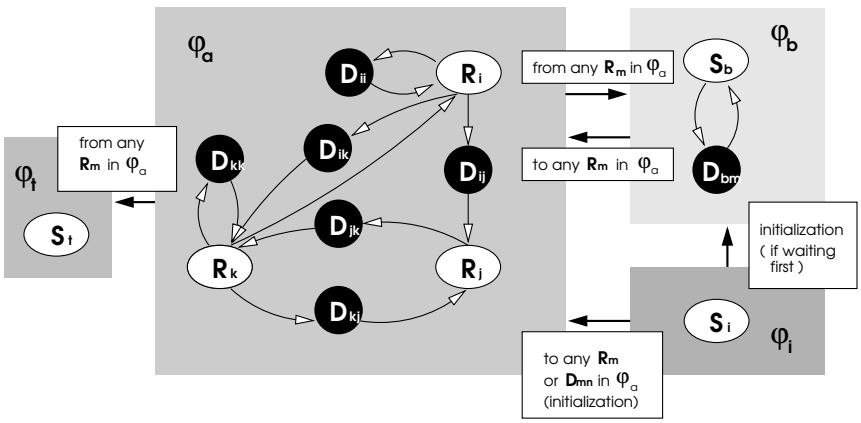


Figure 2: Automata based formal description of single-sources of load (simple example)

received in time, it returns to φ_a and the segment is retransmitted. (One should note that for the purpose of bulk data transfer in the data transfer phase, TCP uses a form of flow control called a sliding window protocol. It allows the sender to transmit multiple packets before it stops and waits for an acknowledgement. During this time, it remains at macro-state φ_a. This leads to faster data transfer.) When TCP receives data from the other end of the connection, it sends an acknowledgement. This acknowledgement is not sent immediately, but normally delayed for a fraction of a second. Therefore, a request generation by TCP typically depends on both client behaviour and status of the Internet.

For a more detailed presentation of our generalized load modeling method we refer the reader to [37]. Quite comprehensive experiences in applying the modeling method have been reported by [19, 5]. A load description language based on the description technique as presented in this section has been elaborated by J.J. Kim [19]. A specification technique with similar description power, however not based on extended finite automata but on extended Petri nets, has been published in [25].

4 Case Study I: Modeling of Internet Traffic at an Interface of Primary Load in Video Communications

In video communications via packet-switched networks, in particular, two classes of applications may be distinguished.

One of these refers to video conferences. Here, the video sequence as collected by a camera at discrete time instants leads to an isochronous stream of data units (uncompressed video frames) over time. The video frames are passed to a video encoder for compression at a frequency of $\omega \frac{\text{frames}}{\text{sec}}$. To achieve encoding in real-time the encoder has to execute compression of one frame in less than $\frac{1}{\omega} sec$. The compressed video frames are passed to a transport system for transmission again as an isochronous stream. The delay ξ (between provisioning of the uncompressed frame and transfer of the compressed frame) of the transport system is strongly determined by the speed of the video encoder.

The second class of important video applications corresponds to applications of the type Video-on-Demand (VoD), where the video sequence to be transmitted is already encoded, i.e. compressed. In this case, the compressed video frames are typically stored in a file and can be read there and be passed to the transport system as an isochronous stream, here too. The periodicity of the isochronous stream at the sender could be determined by the scheduled video display frequency at the receiver.

We will now tackle the problem of load characterization at an application-oriented interface. Real time network services typically distinguish two different coding options, constant bit rate (CBR) and variable bit rate (VBR). CBR defines a fixed rate, and uses coding control options such as quantization level and frame skip to realize the pre-defined rate. This leads to a very smooth data rate in the specified time interval, but for shorter intervals such as ones including just one frame arrival, the nature of the load induced by CBR is still highly dependable on the encoding algorithm used. Furthermore, CBR is not an adequate option for encoding VoD streams because of the fluctuating video quality. This makes it very difficult in deriving very precise models and characterizations for CBR video load on frame level.

VBR uses fixed coding options (as opposed to CBR), e.g. fixed quanization level and frame skip, so the induced data rate is dependent on the motion intensity and entropy of the coded sequence. Because of these fluctuations and their effects on network congestion it is very important to derive adequate models for this type of traffic. So, our following case studies will refer to VBR encoded video traffic.

4.1 Load Characterization of the Frame Length Generating Process

In the following we want to model primary load as it is generated in video communications at an interface close to the video source. In particular, we want to observe the compressed video stream at the interface (IF_P) between application-oriented services and the transport system. As we are going to strictly base load modeling on load measurements we have to collect measurements at IF_P. The arrival process of the isochronous video streams at IF_P being very regular and, in principle, known in advance, we can restrict ourselves to measurements characterizing the attributes of interest of the requests (i.e. the compressed video frames to be transmitted). As is usual in modeling video load [23] in the following we assume that length of frames is the only attribute of interest for the requests observed at IF_P. Therefore, we have to measure the length x_i (in Byte) of the i-th video frame being passed via IF_P at time $t_i = t_0 + \frac{i}{\omega}$sec, if t_0 denotes the start of the observation interval.

Thus, the collected trace of frame lengths $X = \{x_i \mid i = 1, 2, \ldots, n\}$ with n observed frames describes the load. This trace leads to the empirical distribution function

$$\mathcal{H}(s) = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}_{\{x_i \leq s\}}(s), \quad s \in \mathbb{R}, \tag{1}$$

where $\mathbf{1}_\Omega$ denotes the indicator function for the set $\Omega \subseteq \mathbb{R}$.

$\mathcal{H}(s)$ can be considered as load characterization concerning the marginal distribution function of frame lengths. Evidently, the distributions of lengths of data units have a strong impact in case of static resource reservations during data transmission as well as in case of an adaptive model-based quality-of-service management.

We carried out comprehensive load measurements based on well-established standards for video encoding, such as H.261, H.263 and MPEG [9], in order to obtain results of general interest. The series of experiments referred to in this section cover 52 different video sequences, varying the quantization levels from 1 to 18. We exemplify the results by discussing three series of experiments in some more detail, in particular choosing H.261 encoding of the sequences

- Claire, a news announcer in a sequence with very low motion intensity.
- Carphone, a video-recording taken from within a driving car and representing a sequence with periods of rather high motion intensity.
- Foreman, a video sequence of a building-site with permanently very high motion intensity.

Figure 3 depicts the trace of the video frame lengths for the sequence Claire; the figure also includes the empirical frame length distribution calculated according to equation 1 as well as its (astonishingly good) approximation by a Gaussian distribution.

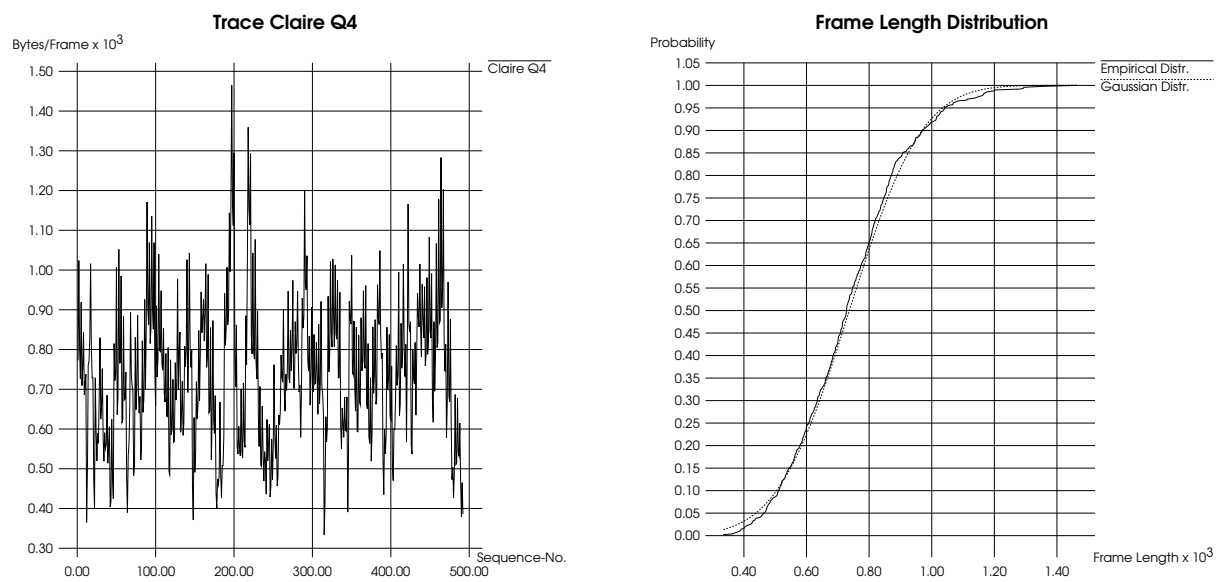


Figure 3: Trace and frame length distribution of the sequence Claire, quantization level 4 (Q4), H.261 encoding

Video sequence Claire leads to the hypothesis that lengths of frames (as produced as a result of H.261 encoding) can be closely approximated by a normal distribution. In order to investigate the validity of this hypothesis we repeated approximation of observed empirical lengths distributions by Gaussian distribution for a variety of other video sequences. The level of accuracy achievable by the approximation was very satisfactory in all the examples considered. As a further graphical illustration of typical deviations observed we refer to Fig. 4 related to the video sequences Carphone and Foreman.
Table 1 summarizes empirical mean $\hat{a} = \frac{1}{n}\sum_{i=1}^{n} x_i$ and estimated variance $\hat{\sigma}^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{a})^2$ determined by maximum likelihood method.

In order to quantitatively judge the accuracy of the maximum likelihood estimates, by means of a χ^2-test, we tested the empirical distribution for normal distribution according to

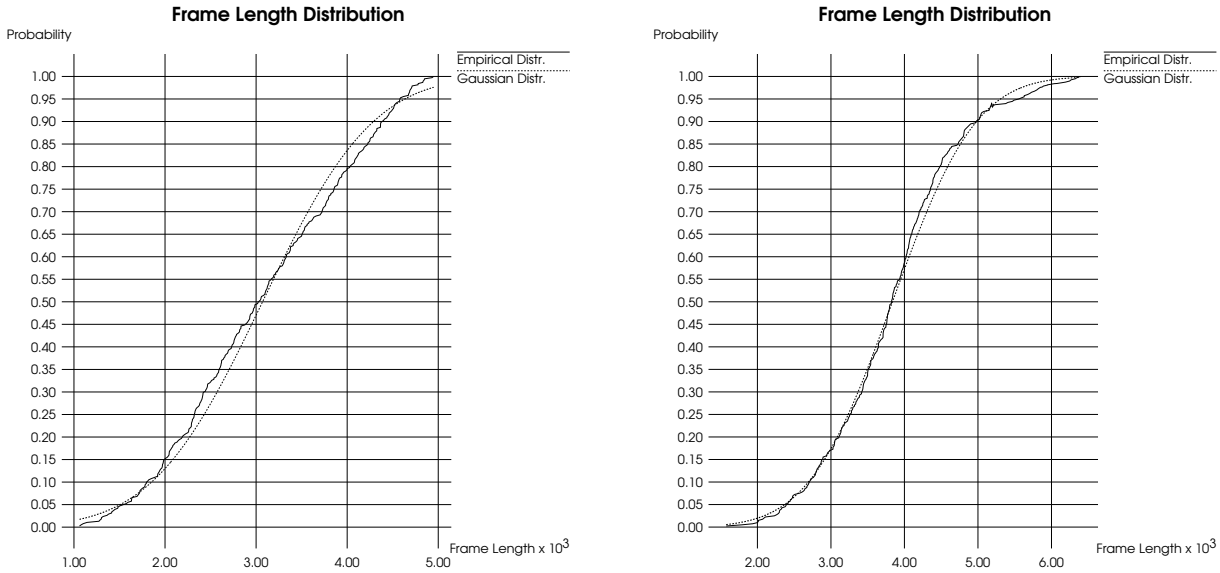


Figure 4: Frame length distributions of the sequences Carphone (left) and Foreman (right), quantization level 4, H.261 encoding

Standard	Quantization	Sequence		
		Claire	Foreman	Carphone
H.261	Q1	6256.17, 509.61	14309.19, 2424.25	12843.40, 2156.95
	Q4	737.70, 181.41	3824.02, 898.64	3071.75, 950.58
	Q10	301.52, 51.34	1154.70, 306.39	967.27, 347.09
H.263	Q1	3229.60, 527.24	10455.31, 1914.63	8987.55, 2141.08
	Q4	334.50, 187.16	1920.45, 558.26	1656.41, 705.45
	Q10	98.89, 93.66	528.91, 172.91	478.58, 257.00

Table 1: Empirical mean $\hat{a}$ and standard deviation $\hat{\sigma}$ of the frame lengths for selected video sequences

$\mathcal{N}(\hat{a}, \hat{\sigma}^2; x) = \Phi(\frac{x-\hat{a}}{\hat{\sigma}})$. Here, classification into 13 partitions (10 degrees of freedom with 2 estimated parameters) has been carried out. The significance level has been chosen to be $\alpha = 0.01$ leading to a significance size of $\chi^2_{0.01,10} = 23.209$ which is considerably higher than the values reached by any of the video sequences observed (cf. Table 2). So we can accept the hypothesis of normal distribution and even some more restrictive values of the significance size would not directly lead to rejection of the hypothesis.

Thus, it seems acceptable to characterize the marginal distribution function of video frame lengths by approximate normal distributions. An important advantage of this approach results from the fact that the normal distribution is determined by only two parameters and it allows a straight-forward derivation of quantiles and other statistical quantities.

Standard	Quantization	Sequence		
		Claire	Foreman	Carphone
H.261	Q1	4.2502	13.876	12.9953
	Q4	6.5248	14.0763	13.4218
	Q10	9.7671	13.1631	14.0233
H.263	Q1	4.2887	16.876	14.1479
	Q4	6.5352	18.1099	14.2638
	Q10	9.9816	14.4627	14.5210

Table 2: Results of the χ^2-test for normal distribution of frame lengths in video sequences with different levels of quantization

4.2 Autocorrelations Caused by Temporal Dependencies

Based on the results of modeling the one-dimensional marginal distribution of the frame length we now want to take a closer look on their autocorrelations. Although the marginal distribution of lengths is a basic and very important measure to characterize the load of video applications, it is necessary to evaluate the autocorrelation coefficients to get an adequate characterization. This is a result of the fact, that independence assumptions induce a smoother traffic than highly correlated processes such as fractal traffic processes, and therefore independence assumptions can be misleading in many cases.

So we have to take a closer look on the autocorrelation function of the traces $X = \{x_i \mid i = 1, 2, \ldots, n\}$, given by

$$\hat{\rho}(\tau) = \frac{1}{\hat{\sigma}_F \, \hat{\sigma}_L \, (n-\tau)} \sum_{j=1}^{n-\tau} (x_j - \hat{\mu}_F)(x_{j+\tau} - \hat{\mu}_L), \tag{2}$$

whereas $\hat{\mu}_F$ and $\hat{\mu}_L$ are the means of the first $n-\tau$ respectively the last $n-\tau$ values induced by the frame length traces. Likely, $\hat{\sigma}_F$ and $\hat{\sigma}_L$ denote the corresponding empirical standard deviations,

$$\hat{\mu}_F = \frac{1}{n-\tau} \sum_{j=1}^{n-\tau} x_j, \qquad \hat{\sigma}_F^2 = \frac{1}{n-\tau} \sum_{j=1}^{n-\tau} (x_j - \hat{\mu}_F)^2, \tag{3}$$

$$\hat{\mu}_L = \frac{1}{n-\tau} \sum_{j=\tau+1}^{n} x_j, \qquad \hat{\sigma}_L^2 = \frac{1}{n-\tau} \sum_{j=\tau+1}^{n} (x_j - \hat{\mu}_L)^2 \tag{4}$$

The measurements of the lag-τ autocorrelation coefficients (autocorrelation function) are shown in Fig. 5.

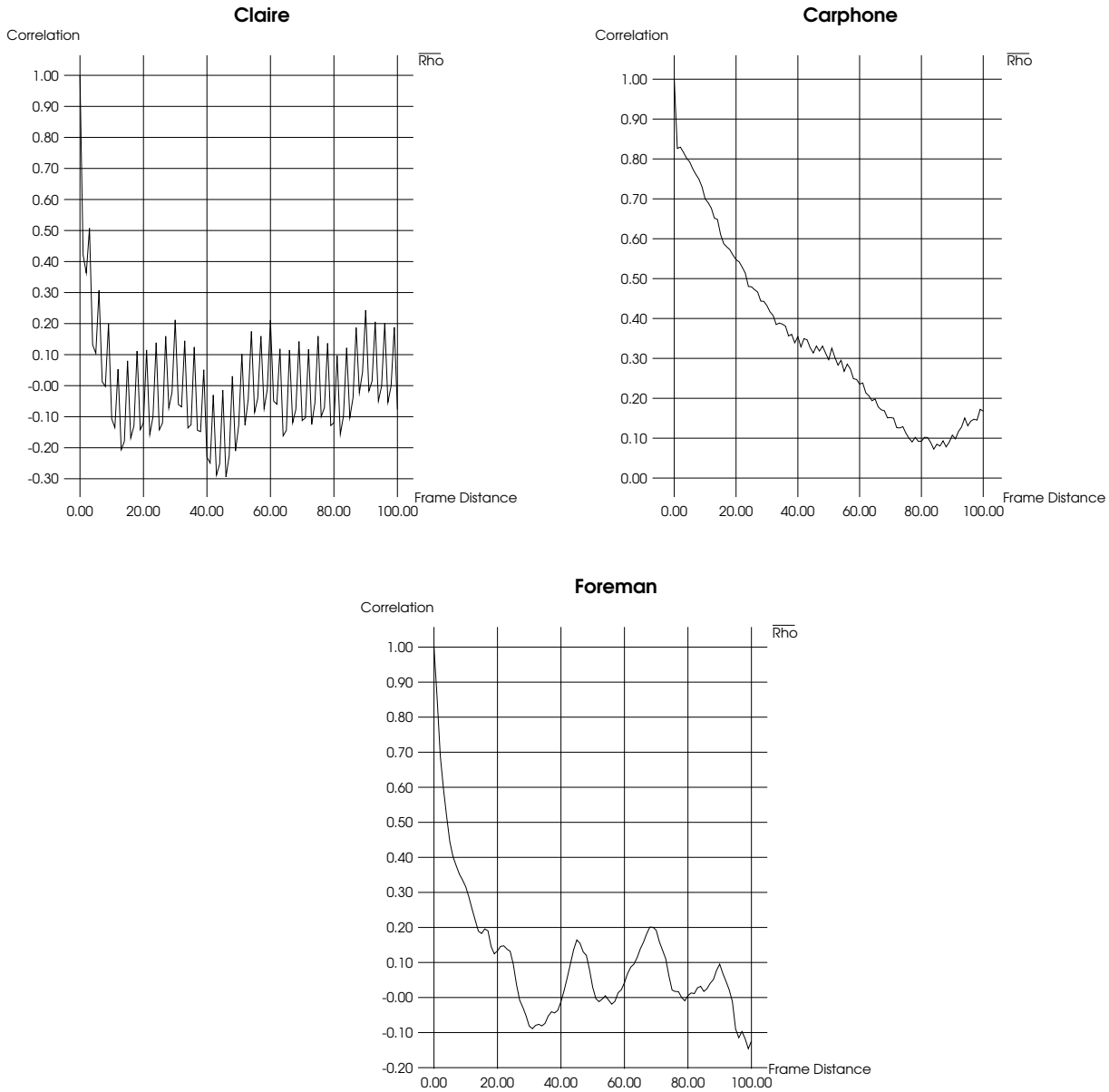


Figure 5: The empirical autocorrelation functions $\hat{\rho}(\tau)$ of the frame length traces for sequences Claire, Carphone and Foreman at quantization level 4

Analyzing the measurement results in figure 5 leads to the conclusion, that a non neglectable autocorrelation up to lag $\tau = 30$ does exist, especially in the traces of Carphone and Foreman sequences, while the sequence of Claire, whose picture contents show the strongest correlation as consequence of low motions, shows the lowest autocorrelation. The oscillating behaviour of the autocorrelation function can be explained by low motion, so that the coding decisions of producing predictive difference pictures alternate periodically.

A closer look on the more strongly correlated frame length traces of Carphone and Foreman reveal changes in the level of required bandwidth of these streams, corresponding to the level of motion within the sequences. This leads to the assumption, that correlation is based on changes in bandwidth requirements and finally on alternation of motion and picure contents.

In most video sequences the major fraction of a picture content persists for a longer while

and the motion intensity is maintained over several frames, so that in these subintervals no rapid changes in bandwidth requirements can be observed.

We now want to investigate in which way the autocorrelation function can be influenced by choice of the interval length of the frame length trace. Therefore, we define subtraces $X_{\mathbb{T}}$ of the frame length trace $X = (x_i : 1 \leq i \leq n)$, with $\mathbb{T} := \{t_1, \ldots, t_2\}$ and $1 \leq t_1 < t_2 \leq n$..

This leads to the following problem: If the amount of the events is too small, e.g. $|\mathbb{T}| \leq 100$, the evidence of empirical correlation function is limited, as a result of increasing the amount of random influences. In order to be independent of the size of subsequences, we compute the means of the correlation coefficient over all subsequent subsequences of a given trace. Let $\{1, \ldots, n\}$ denote the indices of the frame length trace, then we can separate it into m equidistant partitions of subtraces

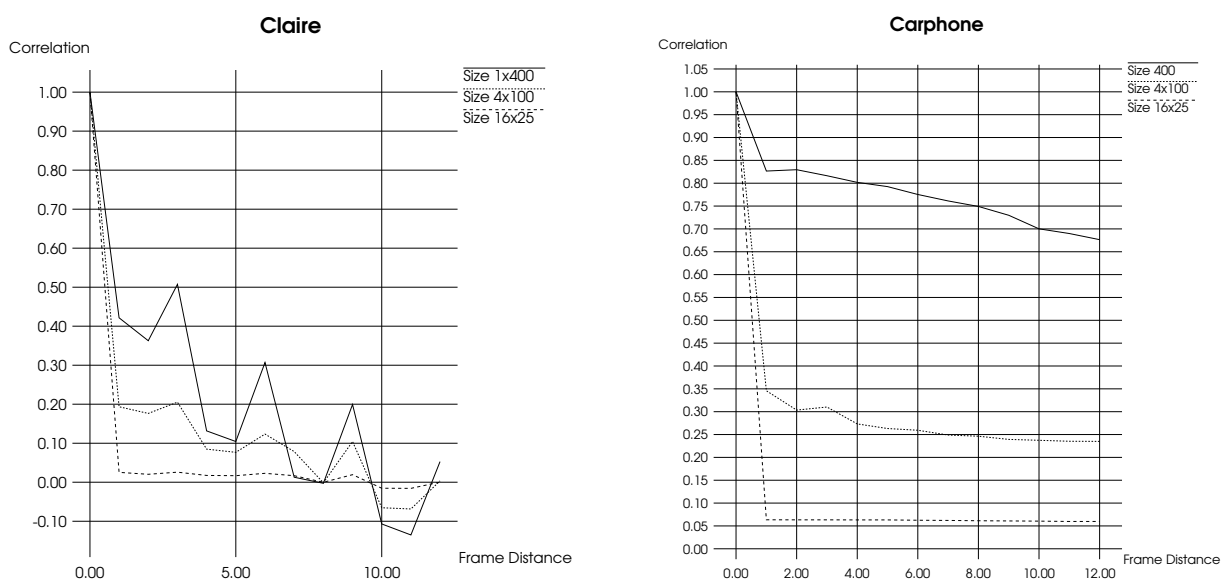
$$\mathbb{T}^m := \{ \{ \lfloor \frac{n(i-1)}{m} + 1 \rfloor, \ldots, \lfloor \frac{n\,i}{m} \rfloor \} \big| i = 1, \ldots, m \}. \tag{5}$$

Given the number of partitions m, i.e. each partition has a mean size of $\frac{n}{m}$, we can get the arithmetically computed correlation structure by evaluating

$$\overline{\rho}(\tau) = \frac{1}{m} \sum_{\mathbb{T} \in \mathbb{T}^m} \hat{\rho}_{\mathbb{T}}(\tau), \tag{6}$$

where $\hat{\rho}_{\mathbb{T}}(\tau)$ denotes the lag τ coefficient of correlation evaluating the subtrace $\mathbb{T}$ according to equation 2. These investigations have been executed for many sequences including Claire, Carphone and Foreman compressed by H.261- und H.263-codecs at different levels of quantization covering a range of 1 to 20. We will take a closer look on the results of H.261-coded video sequences at quantization level 4, according to our investigation above.

All traces were limited to a size of 400 frames and they are separated into 1, 4 and 16 partitions. This choice of 16 partitions limits the subtraces to a size of 25, so we investigate the autocorrelation functions up to lag 12. This will be sufficient to demonstrate the main results when neglecting the long-term autocorrelations. The results are presented in figure 6.



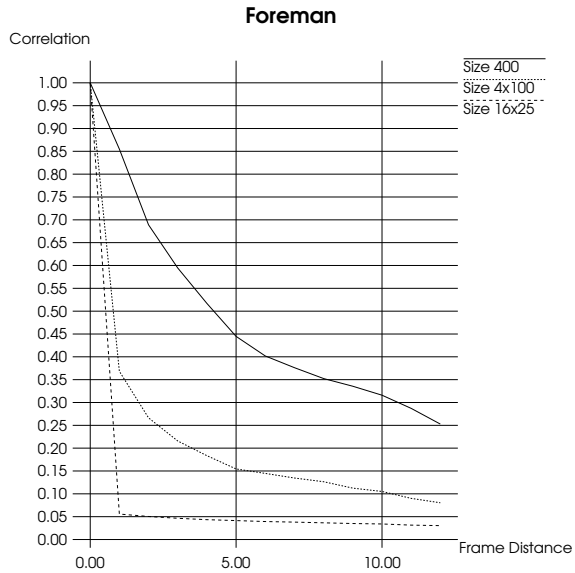


Figure 6: The empirical mean autocorrelation functions $\bar{\rho}(\tau)$ of the frame length traces in dependence of the subsequence size

In all sequences a collapsing of the autocorrelation structure can be observed if the subsequence size is reduced to values of 50 or lower; at sizes of 25 the autocorrelation vanishes completely. These results are confirmed by stochastic independence tests. It should be evident by now that the size of the observed sequences has a significant influence on the correlation structure. This leads to the conclusion, that the correlation structure is a result of long term correlations.

As a result of our measurements we achieve independence, if we compute the model parameter based only on short sequences, e.g. the last 25 frame sizes. This proceeding offers two major advantages. On the one hand, we can calculate the model parameters by implementation of simple ring buffers, and determine mean $\hat{\mu}$ and standard deviation $\hat{\sigma}$ by computing maximum likelihood estimators. On the other hand, this estimator is an adequate estimator of the current load situation induced by the video encoder at IF_P. We obtain a stochastic model for the current and the short-term future situations, namely $\mathcal{N}(\hat{\mu}, \hat{\sigma}^2)$-distributed frame sizes. Based on these independency assumptions we can elaborate more complex models such as models for MPEG-encoded video streams in a similar way.

So, we can identify a trade-off in model selection. On the one hand we can model the frame length process over a long time period. In this case, the autocorrelation structure is not neglectable and has to be modelled by adequate process classes. Normal distributed one-dimensional frame length processes controlled by *Transform Expand Sample*-Processes [15] would be a good choice here. On the other hand, we can use the results above and refer to short-term temporal dependencies, which can be neglected. In this case, we can only judge the current load situation by evaluating the last few events of the frame length generating process. This leads to an accurate characterization, but the parameters $(\hat{a}(t), \hat{\sigma}^2(t))$ are varying smoothly over time. The autocorrelation is reflected by the dependencies of the floating parameter set resulting from evaluation of the ring buffer.

Our measurements covering video sequences, encoded according to MPEG-1 standard,

show that the resulting frame lengths distributions can no longer be sufficiently well approximated by normal distributions, in case that higher levels of quantization (≥ 3) are used. The different types of frames in MPEG (i.e. the I-, P- and B-frames) lead to significantly differing results for the empirical distributions of frame lengths. However, by collecting dedicated measurement data for every single type of frame we still could apply the load model presented in this section to cover MPEG video streams, too. A complete MPEG video source would just have to be modeled as an overlay of three single streams (one stream with only I-, one with P- and one with B-frames), cf. overlay of streams as it is discussed in sections 6 and 7.

5 Load Transformation and its Modeling for the Purpose of Secondary Load Characterization

In the preceding section, by way of example we have investigated primary load as generated by video encoders. Video frames are passed to the communication system for the purpose of transmission. To prepare such a transmission the communication system has to process the video frames (being considered as so-called user data). Processing takes place according to the given communication protocols which, together, constitute the protocol hierarchy. Typical steps of the processing of data units within the protocol layers concern the fragmentation of user data into segments (e.g. packets, cells), the adding of protocol control information to user data (e.g. in the form of headers or trailers of protocol data units) or the creation of new data units without including any user data (e.g. creation of acknowledgements) [34].

We can interprete the processing of data units within protocol layers as a process of transformation effective on the primary load and producing the so-called secondary load. Let us denote components which transform load as *(load) transformers*. Load transformers change the properties of the load e.g. in such a way that, on one hand, data units corresponding to the secondary load may become larger or smaller than those of the primary load or that, on the other hand, the interarrival times of requests may be changed. Evidently, requests arriving at a primary load interface IF_P are not only modified but also always arrive at the secondary load interface IF_S after some delay, which may even vary from request to request and thus lead to the typically different request interarrival times at IF_S (as compared to the interarrival times at IF_P).

5.1 Real Load Transformation versus Load Transformation in a Modeling Domain

In communication systems a primary load at some interface IF_P within the protocol hierarchy induces a secondary load at some lower layer interface IF_S. Characterization of secondary load in many cases is as important as or even more important than characterization of primary load.

In characterizing the secondary load, as it would be induced by some given primary load, the following two approaches can be distinguished :

- direct measurement of the (real) load at interface IF_S in an existing communication system,

- modeling of the secondary load.

In case of the first approach we would have to generate the primary load of interest in the real network and measure the secondary load which arrives at IF_S after having passed the real transformation process (e.g. the protocol processing). This approach is not feasible if the interface IF_S is not accessible for measurements in an existing network or during design or early development of a new communication system, when IF_S would not yet be implemented. Moreover, it could be necessary to investigate a secondary load as it would be induced by a very special mix of single sources on the level of primary load and it could be impossible to generate this mix in the existing network.

In the second approach based on modeling, the influence of parts of a communication system on a given primary load is reflected by a model. Here, the real transformation process is replaced by a so-called *artificial transformer* (cf. Fig. 7). The purpose of an artificial transformer is to convert the attributes (and their values) of the primary load into those of the secondary load as well as to transform the arrival process of primary load requests into the one for secondary load requests. If the artificial transformer used is a sufficiently valid model of reality we can obtain a realistic prognosis of secondary load to be expected. Besides validity, an important requirement towards artificial transformers is their broad applicability, e.g. thanks to some flexible parametrization.

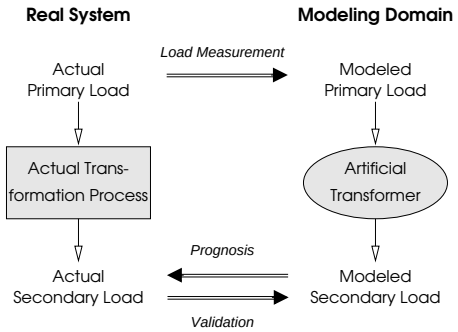


Figure 7: Load transformation in a real system versus transformation in a modeling domain

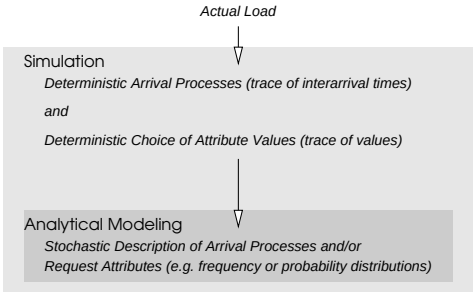


Figure 8: Levels of abstractions in load modeling

5.2 Modeling Load Transformation to Characterize Secondary Loads as Induced by Given Primary Loads

Load characterization has to cover the specification of the arrival process of requests as well as the specification of the values of request attributes. The characterization can be deterministic if we use, e.g., some trace or it may be probabilistic if we use, e.g., some probability distribution to reflect the interarrival times and the attribute values of the requests generated over time. This implies the levels of abstractions for the load as depicted by Fig. 8, namely

- the actual load,
- its deterministic description as a trace, or

- its probabilistic characterization by means of distributions.

Fig. 8 indicates that some measured load can be approximated by a distribution (e.g. to characterize lengths of data units) which may be directly used as a model of primary load by an artificial transformer. The artificial transformer may then reflect the transformation process just by changing (recalculating) the given distribution into a new one to approximate the induced secondary load (cf. section 6 for examples). This is an example where load transformation is expressed by an analytical model as here the artificial transformer just corresponds to mathematical calculations. Again, the validity of the predictions of the artificial transformer - in terms of a probability distribution to characterize secondary load - can be determined by means of comparisons with measured secondary load (cf. Fig. 9 for some graphical illustration of the proceeding).

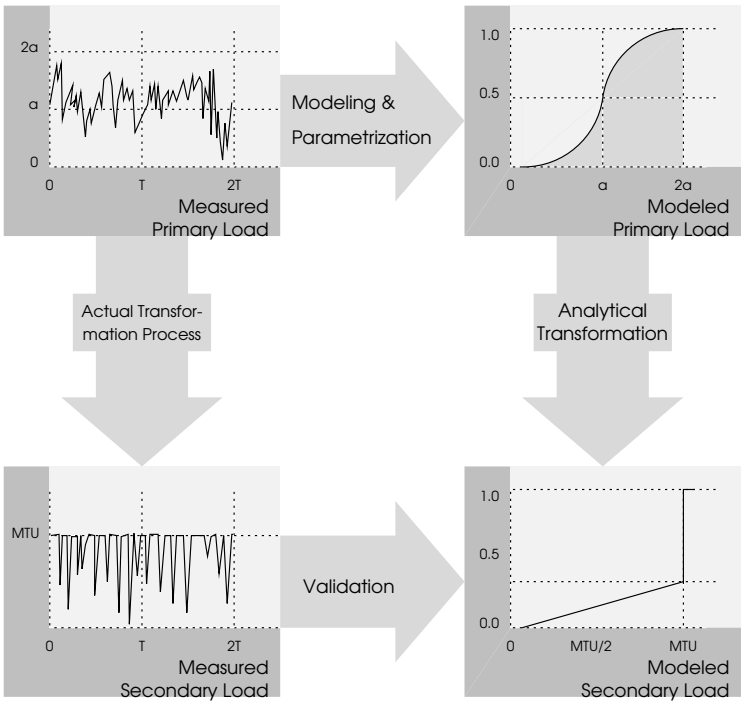


Figure 9: Analytical modeling of load transformations

In the model(ing) domain load transformations can be simulated or be the result of model calculations corresponding to the analytical evaluation of a mathematical model. As is common in modeling, in the special case of modeling load transformations, too, analytical modeling offers the important advantages of only minor programming and calculation effort in evaluating the required formulas as well as leading to more comprehensive possibilities of model evaluations and result interpretations.
When simulating load transformation processes we can distinguish two ways of proceeding. In the first approach, for any given transformation process we can elaborate a dedicated new simulation program the functionality of which is sufficiently close to the given real transformation process [19].

In the second, more advanced approach we can try to consider a (complex) transformation process as a combination of separate elementary transformations, each of these elementary transformations being modeled as an elementary building block (by means of a relatively simple simulation program) and then simulate the complex transformation by means of combining an appropriate set of elementary building blocks. The latter approach has already been successfully applied by us [5, 6] and we could show that the expenditure in modeling required could be reduced significantly as compared to the first approach of using dedicated simulation programs. As elementary building blocks we applied, in particular, one class of simple transformations having only an impact on the arrival process of requests and another class in which the transformation only modifies the set of request attributes or their values.

5.3 Examples of Generalized Transformers and Load Transformations in the Internet

A first class of simple load transformations only allows the modification of the interarrival times between requests. Here the departure process of requests after transformation differs from the arrival process, whereas request types and attributes remain completely unchanged. Examples of such transformations are algorithms for smoothing traffic such as *Leaky Bucket*- or *Token Bucket*-Algorithms [27]. A second class of simple transformations modifies request types and attributes but keeps constant the request interarrival times. To give an example for a transformation of the second class, generation of packet headers typically does not significantly change the packet interarrival times whereas it modifies the attribute *packet length*.

Some transformations within communication systems are quite complicated and therefore hard to be modeled, such as in the case that manipulation of requests in a packet-switching network with complex protocol software and hardware is even dependent on the network's state. Other transformations can be modeled in a quite straight-forward manner, such as fragmentation. Fragmentation is a typical example for a load transformation within the Internet, but it is also very common in other networks. As fragmentation we denote the separation of a data unit (e.g. a message or the data corresponding to an encoded video frame) into a sequence of data packets, each of which has been associated with a dedicated packet header. Two types of fragmentation can be distinguished: fragmentation into data units with a fixed maximum length (as e.g. in the Internet protocol stack, cf. section 6) and fragmentation into constant lengths data units (e.g. cells in ATM, cf. [27]).

Adding protocol control information to data units (cf. header generation as introduced in section 6) is also a simple transformation having an impact, in particular, on the attribute length. This transformation takes place in each layer of the protocol hierarchy whereas fragmentation is only occurring if the maximum packet length (cf. MTU as in section 6, in the Internet at least 576 Byte) of the next lower protocol layer resp. of the subsequent subnetwork is smaller than the one of the actual layer.

During a transmission according to the TCP protocol, data packets which do not yet transport user data are created for establishing and terminating a connection. Moreover, as a consequence of flow control, transformations can result, which modify the timing of the original arrival process in a way which depends on the state of the network [18]. As an example for the effect of flow control between adjacent protocol layers, the LLC layer (i.e.

the network access layer in the IP architectural model) may only accept packets of the IP layer if buffers on LLC layer are empty. Thus, outstanding acknowledgements may produce a back-pressure and therefore strongly influence the interarrival times of data units (packets) on the next lower layer.

6 Case Study II: Modeling of IP Traffic at an LLC-Interface as an Example of Secondary Load Characterization in a Video Server

6.1 Load Transformation within an IP Protocol Stack

The general concept of load transformation (e.g. by protocol layers) as introduced in the last section will now be exemplified in looking at load transformations being typical for the Internet. Protocols which have been most stable within the overall IP protocol stack are related to Network and Transport Layer:

- functionality of the Network Layer is largely determined by the Internet Protocol (IP [33] in its versions 4 and 6), providing an unreliable, connectionless packet-switching service (resp. datagram service for short);
- on Transport Layer there exists on one hand a reliable connection-oriented transport service based on TCP [33]; on the other hand an unreliable, connectionless transport service based on UDP [33] is available; moreover, there exist additional transport protocols which have been elaborated more recently such as RTP [31], e.g. to support real-time communications.

As we want to make use of our results for primary load modeling as presented in section 4, in the following we assume video communication directly based on UDP and IP protocols (cf. Fig. 10).

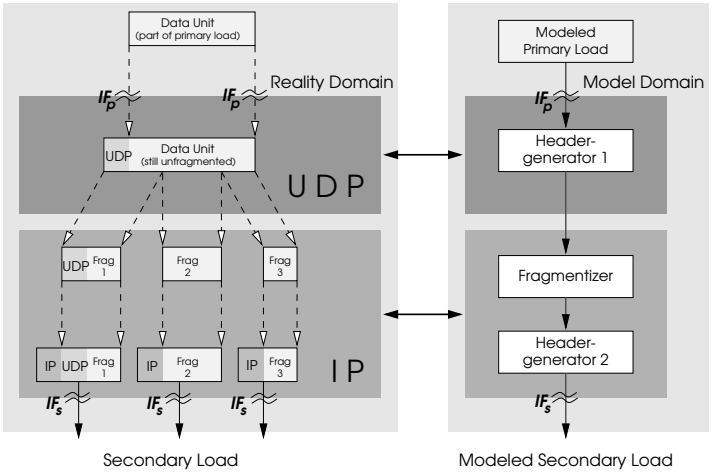


Figure 10: Modeling secondary load using load transformers

The interface IF_P which we choose to observe primary load, as it is generated by the video sources, corresponds to the UDP transport service-access-point (TSAP). The interface IF_S chosen by us for secondary load observation and modeling corresponds to the interface between IP layer and the network adapter (i.e. Ethernet adapter in our case). The interface IF_S chosen is still sufficiently *high level* in order to make results of load modeling not too much dependent of the network technology used and of the adapter's implementation (e.g. its buffer management). Let us shortly discuss the load transformation now, as it is effective between interface IF_P and IF_S. To keep the case study sufficiently simple let us assume that the primary load characterization at IF_P refers to

- the arrival process of requests at IF_P,
- only one existing type of requests, namely data units (e.g. the encoded video frames) to be transmitted via UDP,
- the data unit length as the only attribute of requests.

So, the load transformation resulting from UDP/IP functionality implies a modified arrival process of requests (now IP packets) at IF_S. At interface IF_S we still observe exactly one type of requests, again with (packet) length as the only attribute. Therefore to better understand load transformation as resulting from UDP/IP, we have to take a closer look at how data units, as offered to UDP at IF_P, are manipulated within UDP/IP Layers. As packet length at IF_S is the only attribute of interest, it is sufficient to focus on manipulations of data units within UDP/IP layers which have an impact on length. Length of data units is changed by UDP when adding the headers with UDP specific control information to data units. On Network Layer, fragmentation by IP changes not only the length but also the number of data units. Maximum data unit length used for fragmentation results from an agreement between IP and network adapter, being kept fix during network operation. Evidently, after fragmentation IP, too, adds IP specific control information to packets. Fig. 10 summarizes transformations which have an impact on lengths of data units being processed by UDP/IP. The figure also suggests the modeling of UDP/IP transformations by mapping these transformations (in the model domain) onto three elementary transformers, namely two Header-Generators and one Fragmentizer, which are placed in series.

6.2 Load Transformation for Single Sources of Load

In section 4 we concluded that a single source of video traffic can be adequately characterized by the distribution of lengths of video frames which are generated according to the video display frequency used. Let $\mathcal{L}(x)$ be the probability distribution of data unit lengths which resulted during modeling of primary load, observed at IF_P. In the following, we want to derive mathematically the impact which transformers of type *Header-Generator* and *Fragmentizer* have, in particular leading to new distributions of lengths.

Transformation of single requests by UDP implies that a header of $v = 8$ Byte length is added to data units. Here also a checksum is calculated for the user data leading to a delay being proportional to data unit length. According to measurements (for a Pentium-166 PC under Linux) this delay varies between 50 to about 250 μsec and thus is rather small compared to the interarrival times of video frames at IF_P of, e.g., 33 msec. So, in secondary

Type of network	MTU (Bytes)
Hyperchannel	65535
16 Mbit/s Tokenring (IBM)	17914
4 Mbit/s Tokenring (IEEE 802.5)	4464
FDDI	4352
Ethernet	1500
X.25	576

Table 3: Recommended Maximum Transmission Unit (MTU) lengths

load modeling, we will neglect the delay resulting of UDP processing.

The new distribution of data unit lengths $\mathcal{U}(x)$ which is a consequence of header generation by UDP is reflected by the following equation

$$\mathcal{U}(x) = \int_{-\infty}^{x} d\mathcal{L}(s - v) = \mathcal{L}(x - v). \tag{7}$$

UDP datagrams are passed to IP which, by means of fragmentation, has to make sure that the Maximum Transmission Unit (MTU)-length of the next lower layer LLC (Logical Link Control) is respected. Therefore, if a data unit handed over to IP has a length larger than the value MTU-length minus IP-header-length it will be fragmented. Typical MTU-lengths are summarized in Table 3.

In modern operating systems, such as UNIX, Linux or Windows NT, fragmentation takes place in a Shared Memory Interface and demands only a negligible amount of processing time. If Θ denotes MTU-length in Byte (e.g. $\Theta = 1500$ Byte for Ethernet) and ψ denotes IP-header-length in Byte (e.g. $\psi = 20$ Byte in case of IPv4 or $\psi = 40$ Byte for IP-version 6) the maximum length of fragments ϑ can be calculated as $\vartheta = \Theta - \psi$.

Thus, we can directly calculate the expected number of fragments β, a value which is of great significance in characterizing the burstiness of a traffic source. Let $\beta(n)$, $n = 1, 2, \ldots$ denote the probability that a UDP data unit is fragmented into exactly n segments, then

$$\beta(n) = \int_{(n-1)\,\vartheta}^{n\,\vartheta} d\mathcal{U}(x) = \mathcal{U}(n\,\vartheta) - \mathcal{U}((n-1)\,\vartheta), \tag{8}$$

which allows straight-forward calculation of β, namely

$$\beta = \sum_{i=1}^{\infty} i\,\beta(i) = \sum_{i=1}^{\infty} i\,\big(\mathcal{U}(i\,\vartheta) - \mathcal{U}((i-1)\,\vartheta)\big). \tag{9}$$

As for every distribution F we have $\lim_{x\to\infty} F(x) = 1$, it is evident that for every error bound $\epsilon > 0$ we find an $l \in \mathbb{N}$ with

$$l = \min_{k\in\mathbb{N}} \big\{\mathcal{U}(k\vartheta) \geq 1 - \epsilon\big\}, \tag{10}$$

so that the approximation $\mathcal{U}(k\vartheta) \to 1$ is valid for $k \geq l$. Thus, we obtain the following

approximation for β:

$$\begin{aligned}\beta \approx \sum_{i=1}^{l} i\left(\mathcal{U}(i\vartheta) - \mathcal{U}((i-1)\vartheta)\right) &= \sum_{i=1}^{l} i\mathcal{U}(i\vartheta) - \sum_{i=0}^{l-1}(i+1)\mathcal{U}(i\vartheta) = \\ = l\mathcal{U}(l\vartheta) - \sum_{i=0}^{l-1}\mathcal{U}(i\vartheta) &\approx \sum_{i=0}^{l-1} 1 - \mathcal{U}(i\vartheta).\end{aligned} \tag{11}$$

Equation 11 now allows us to calculate the distribution $\mathcal{F}(x)$ of lengths of fragments generated by IP:

$$\begin{aligned}\mathcal{F}(x) &= \begin{cases} 0, & x \leq 0, \\ \frac{1}{\beta}\sum_{i=0}^{l-1}\int_{i\vartheta}^{i\vartheta+x} d\mathcal{L}(s-\upsilon), & 0 < x < \vartheta, \\ 1, & x \geq \vartheta. \end{cases} \\ &= \begin{cases} 0, & x \leq 0, \\ \frac{1}{\beta}\sum_{i=0}^{l-1}\mathcal{L}(x+i\vartheta-\upsilon) - \mathcal{L}(i\vartheta-\upsilon), & 0 < x < \vartheta, . \\ 1, & x \geq \vartheta. \end{cases}\end{aligned} \tag{12}$$

After fragmentation the IP header is created and added to the corresponding fragment. According to measurements (again for a Pentium-166 PC under Linux) the CPU processing time required for header creation by IP is about 30μsec. This value is of interest as it strongly influences the packet interarrival times within bursts of packets, resulting from IP fragmentation, at the interface between IP and LLC layer, i.e. at interface IF_S in our load modeling example.

Concerning lengths of IP packets at interface IF_S we obtain the corresponding distribution $\mathcal{I}(x)$ of lengths directly as $\mathcal{I}(x) = \mathcal{F}(x-\psi)$, or using equation 12

$$\mathcal{I}(x) = \begin{cases} 0, & x \leq \psi, \\ \frac{1}{\beta}\sum_{i=0}^{l-1}\mathcal{L}(x+i\vartheta-\upsilon-\psi) - \mathcal{L}(i\vartheta-\upsilon-\psi), & \psi < x < \vartheta+\psi, \\ 1, & x \geq \vartheta+\psi. \end{cases} \tag{13}$$

Thus, equation 13 successfully completes our search for the distribution of data unit lengths at the secondary load interface IF_S. Moreover, our results also cover characterization of the mutual dependencies between fragments, in particular, the probability that a fragment is followed by another one referring to the same UDP datagram is determined by the array $\vec{\beta} = (\beta_1, \beta_2, \ldots)$, cf. eq. 8, as well as its expectation β. Our solution for calculating β directly (for a given distribution of data unit lengths at a primary load interface) is of important practical relevance: among others dimensioning of resources, such as appropriate choice of buffer sizes, and model based quality-of-service (QoS) management [2] may be considerably supported by knowledge of β.

6.3 Load Transformation for Complex Primary Load

In the following we want to generalize our discussion of load transformation to the case that primary load results from an overlay of m single sources. In video communication this

situation could correspond to a video server with load produced by m independent video sources S_i. Referring to section 4 we could characterize the single sources by m load models $\mathcal{L}_i(x) = \Phi(\frac{x-\hat{a}_i}{\hat{\sigma}_i})$, $\forall i = 1, \ldots, m$, and assume requests (video frames) of source S_i being generated with periodicity T_i beginning at starting instant τ_i, i.e. generation of requests at instants τ_i, $\tau_i + T_i$, $\tau_i + 2T_i$, ... Let be $\mathbb{T}_i = \{\tau_i + nT_i \,|\, n \in \mathbb{N},\, n < N_{\max}\}$ the set of all observed arrival times of stream i. Thus, the relative proportion α_i of arrivals concerning the i-th stream to the overall arrival process can be determined i.e. $\alpha_i = |\mathbb{T}_i| \left(\sum_{j=1}^{m} |\mathbb{T}_j|\right)^{-1}$, the relative proportion α_i^* of the departure process of the complex secondary load is given by $\alpha_i^* = \alpha_i \beta_i \left(\sum_{j=1}^{m} \beta_j\right)^{-1}$, where β_j are determined by equation 9 for all streams $i = 1, \ldots, m$. So, equation 12 and 13 imply

$$\mathcal{I}(x) = \begin{cases} 0, & x \leq \psi, \\ \sum_{i=0}^{l-1} \sum_{j=1}^{m} \alpha_j^* \big(\mathcal{L}_j(x + i\,\vartheta - \upsilon - \psi) - \mathcal{L}_j(i\,\vartheta - \upsilon - \psi)\big), & \psi < x < \vartheta + \psi, \\ 1, & x \geq \vartheta + \psi. \end{cases} \quad (14)$$

Equation 14 now allows us to characterize the secondary load (in terms of distribution of packet lengths) which is induced by a complex primary load representing e.g. the overlay of single video sources in a video server.

Among others, our results would allow us to directly use and easily parametrize a packet train model [16] (with deterministic intertrain- and intercar-times) as a realistic description of the secondary load to be expected.

7 Validation of Our Transformer Approach in Secondary Load Characterization by Means of Analytical Modeling

We now want to validate the accuracy of our method for secondary load prediction based on application of a load transformer. To prepare the validation we start with measuring both, the primary load (PL) as generated by single video sources, as well as the secondary load which is induced by PL and observed at the interface (IF_S) between IP and LLC layer. Measurements have been carried out for Pentium PCs (166MHz, under Linux) on one hand assuming an MTU size of 1500 Byte (which corresponds to the MTU size used in Ethernets as presently dominating LAN network infrastructure) and on the other hand supposing an MTU size of 576 Byte (used in the context of X.25 and also minimum MTU size in the Internet) [33].

The single sources of primary load active during the measurements have to be modeled in order to allow load transformation in the modeling domain. Each single source is mapped onto a load generator creating a sequence of requests (video frames to be transmitted) with a single attribute *length*. As suggested in section 4 the empirical distribution of length for PL is approximated by a Gaussian distribution $\mathcal{N}(\hat{a}, \hat{\sigma}^2)$, which can be mathematically transformed into the distribution to characterize the lengths of data units at the secondary load interface IF_S. To perform the transformation we just have to apply equation 13 of section 6 to the given normal distribution. A χ^2-test is again used in order to validate the prognosticated distribution for secondary load with respect to the actual, measured lengths at IF_S.

7.1 Validation of the Secondary Load Model in Case of Single Sources

As single sources of primary load in the following validation experiments we choose, by way of example, the video sequences Carphone and Foreman (cf. section 4). Both sequences were H.261 encoded selecting a quantization level of 4.

Carphone as single source: Table 1 showed that the Carphone sequence can be approximated by a Gaussian distribution with parameters $\hat{a} \approx 3071.75$ Byte and $\hat{\sigma} \approx 950.58$. The analytical transformation according to eq. 13 leads to the following values β, characterizing burstiness of load at interface IF_S, namely $\beta \approx 2.581$ (assuming an MTU size of 1500 Byte) and $\beta \approx 6.039$ (assuming an MTU size of 576 Byte).

Figure 11 depicts the distribution functions which characterize secondary load. The empirical distribution is based on an observation interval of 900 seconds duration.

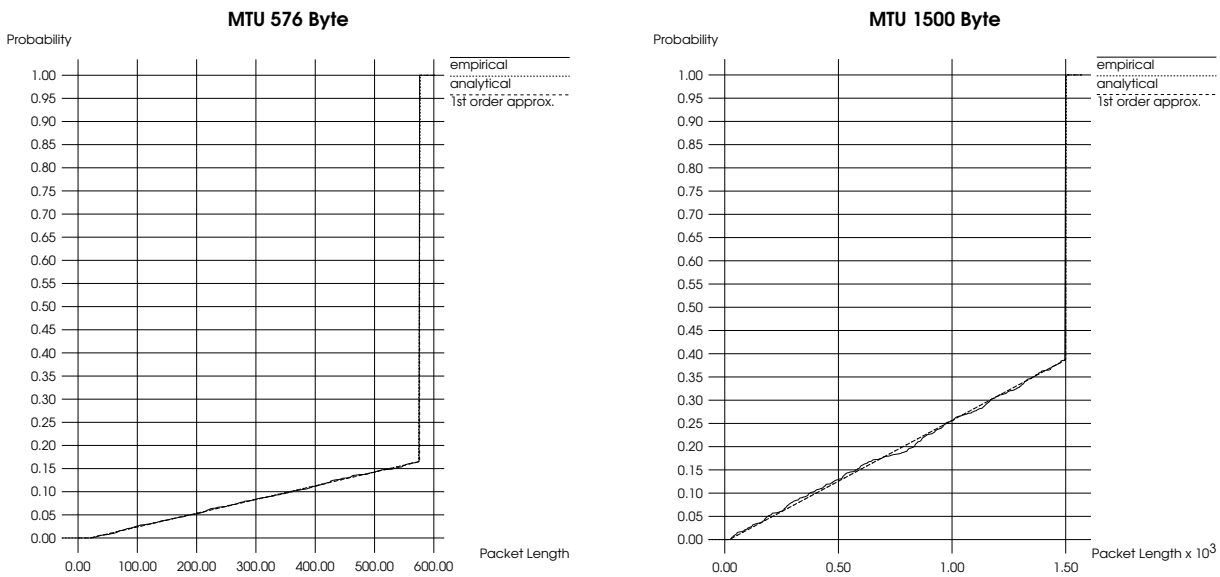


Figure 11: Comparison between measurement results and analytical modeling of secondary load (sequence Carphone, H.261, quantization level 4)

Besides the excellent conformity between empirical and analytical distribution it is remarkable that the distribution function characterizing the pure fragments (i.e. packets with lengths smaller than MTU size) is nearly linear. Would linearity be exactly fulfilled we could obtain a first approximation of the complete distribution function just by linear interpolation between the 3 points $\{(\psi, 0.0), (\Theta, \frac{1}{\beta}), (\Theta, 1.0)\}$. This approximation has also been included into Fig. 11 though it can not be distinguished from the analytically calculated function as the deviations between both curves are so small. This implies that, at least for this video sequence, calculation of β is completely sufficient to come up with a good model for approximation of the marginal distribution of secondary load.

A χ^2-test was executed with 13 partitions and, as 2 parameters were estimated, the reference value $\chi^2_{\alpha,10}$ was used. For a confidence level of $\alpha = 0.01$ this yields to a reference value of 23.209. Table 4 shows that this value is far from being reached by all of the video

sequences and quantization levels considered and it is evident that this would be true even for much higher confidence levels. Thus the results don't argue for rejecting the hypothesis that the empirical distribution is conform to the analytically determined one. Besides, this claim has also been confirmed by the results of a Kolmogoroff-Smirnoff-test not presented here.

MTU = 1500 Byte	Quantization	Sequence		
		Claire	Foreman	Carphone
H.261	Q1	1.6452	2.0647	1.8524
	Q4	1.7026	2.4532	1.8853
	Q10	1.8947	2.9362	1.9042
H.263	Q1	1.4804	1.9532	1.8529
	Q4	1.6732	2.0064	1.8763
	Q10	1.8773	2.3473	1.8963
MTU = 576 Byte				
H.261	Q1	0.2301	1.9542	1.6532
	Q4	0.6230	1.9978	1.6952
	Q10	0.8290	2.1522	1.7832
H.263	Q1	0.2241	1.8933	1.6032
	Q4	0.5342	1.9523	1.6239
	Q10	0.7685	1.9932	1.7050

Table 4: Results of a χ^2-test using 13 classes and 10 degrees of freedom respectively

Foreman **as single source:** As second example of a single source of primary load we investigated the video sequence Foreman. The results obtained are reflected by Figure 12. Here, Foreman sequence was approximated by a Gaussian distribution with parameters $\hat{a} \approx 3824.02$ Byte and $\hat{\sigma} \approx 898.64$ (cf. Table 1). As in the first example (Carphone), MTU sizes of 576 Byte and 1500 Byte were considered, leading to values for $\beta \approx 7.392$ in the first case and $\beta \approx 3.089$ in the second.

Further validation results covering also the video sequence Claire as well as H.263 (besides H.261) as an additional video encoding algorithm and choice of different levels of quantization are summarized in Table 4. Again, all results of validations are highly satisfying.

7.2 Validation in Case of Overlay of Multiple Sources

To stress our analytical modeling approach for secondary load characterization, in the following we want to drop the restriction of a single source of primary load. Instead, we assume a complex primary load resulting from an overlay of m single sources (in particular: $m \in \{3, 30\}$).

In the first series of experiments we consider a video server communicating with 3 video clients. As video sequences we choose Carphone, Claire and Foreman. The server is sending sequence Claire on a quantization level of 10 ($\hat{a} \approx 301.52, \hat{\sigma} \approx 51.34$) with a video frame frequency of 12 frames/sec, Carphone on a quantization level of 4 ($\hat{a} \approx 3071.75, \hat{\sigma} \approx 950.58$) with 15 frames/sec and Foreman on a quantization level of 1 ($\hat{a} \approx 14309.19, \hat{\sigma} \approx 2424.25$) with

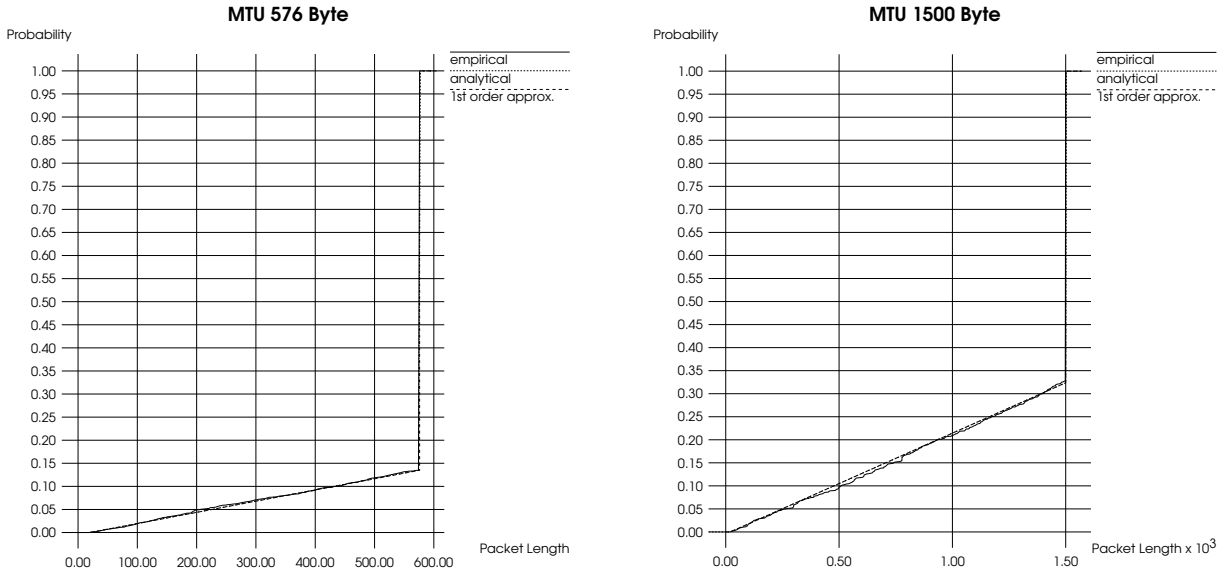


Figure 12: Comparison between measurement results and analytical modeling of secondary load (sequence Foreman, H.261, quantization level 4)

30 frames/sec.

We calculated $\beta = \sum_{i=1}^{m} \alpha_i \beta_i$ according to our analytical model for mixed traffic load transformation and found $\beta \approx 3.0891$ (for MTU size of 1500 Byte) and $\beta \approx 7.3922$ (for MTU size of 576 Byte). Comparisons between empirical and analytically determined distributions are presented in Figure 13.

The strong non-linearity in the empirical distribution function (for an MTU size of 1500 Byte), cf. Fig 13, for a value of the IP packet size of about 300 Byte is a consequence of the videosubstream Claire which is part of the primary load and remains completely unfragmented because of the already very small original video frames, as observed at IF_P. Even in this case, the analytically determined distribution still predicts highly precise results whereas the 1st order approximation leads to small deviations. Nevertheless for practical purpose even the accuracy of the 1st order approximation should be acceptable in most cases, especially as the error in calculating β (according to our analytical model) is neglectable here again. Results of the χ^2-test lead to a value of 1.4767 (with MTU size of 1500 Byte) and 0.9763 (with MTU size of 576 Byte) for the distribution determined by analytical load transformation. For the 1st order approximation, χ^2-test provides values of 9.0654 (MTU size: 1500) and 3.8722 (MTU size: 576). As in the case of single sources of load these values again don't argue for rejection of the hypothesis that the empirical packet length distribution is adequately approximated by both distributions suggested (analytical and 1st order approximation).

Typically a video server simultaneously serves a large number of clients. Therefore, in the following, we model a server serving 30 clients at the same time. All single sources of primary load are assumed to have different characteristics, in particular, we have chosen the following experimental boundary conditions:

- 18 different video sequences being part of the overall 30 sequences,

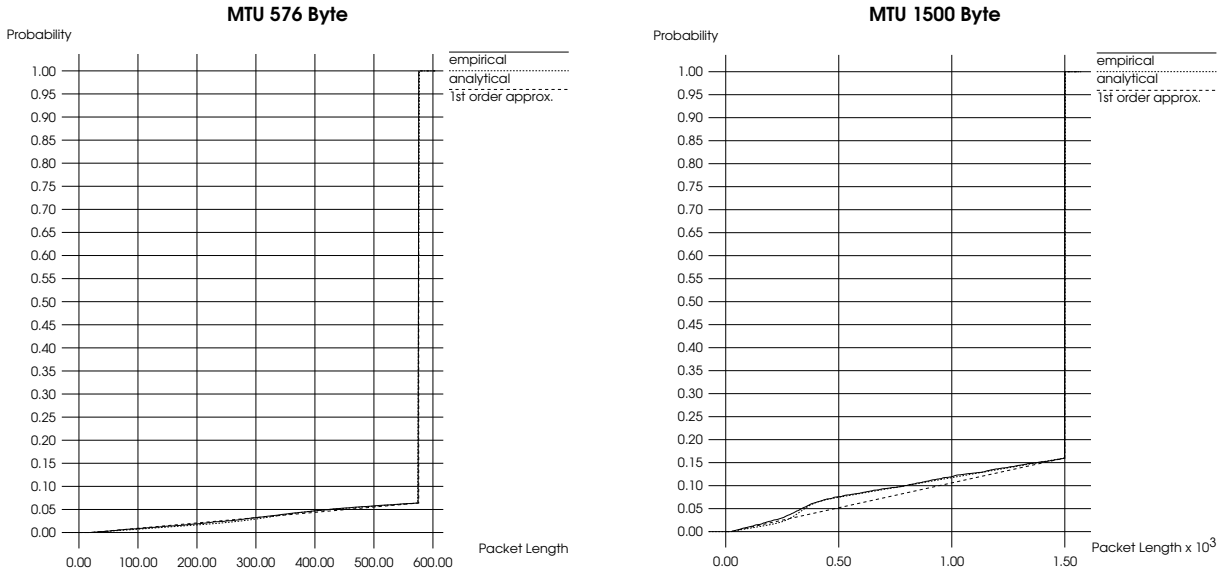


Figure 13: Comparison between measurement results and analytical modeling of secondary load (overlay of 3 sequences, H.261 encoding)

- levels of quantization between 1 and 10,
- frame frequencies between 20 and 30 frames/sec,
- 20 sequences are H.261 encoded, the rest (10) is H.263 encoded.

The length of the observation interval during measurements was taken to be 180 sec because of the large amount of resulting measurement data. Figure 14 demonstrates the validation results for both MTU sizes investigated.

The overlaying of a large number of single sources intensifies the effect of linearity in the distribution function of IP packet lengths up to the MTU size. We not only obtain highly accurate results with the exact analytical model but also get only very small deviations for the 1st order approximation. Quantitatively, this means that χ^2-test leads to calculated deviation χ_0^2 of 0.4325 (MTU 1500 Byte) and 0.0631 (MTU 576 Byte) for the exact analytical model and a value for χ_0^2 of 1.3241 (1500 Byte) and 0.2123 (576 Byte) in case of the 1st order approximation.

As a general conclusion of our numerous validation experiments, in all comparisons between the empirical packet length distribution (based on the actual measurements) and the mathematically predicted length distribution (based on our analytical load transformation model) we observed excellent agreement between both distributions. Predicted secondary load characterizations were found to remain valid even when substantially increasing the error introduced in primary load modeling. This results from the fact that, quite often, already the precise prediction of the expected number of fragments allows one to produce a sufficiently realistic characterization of secondary load (cf. 1st order approximation). The quality of such a simplified prediction is still improved with an increasing value of β and an enlargement of the number of overlayed single sources of primary load (inducing the secondary load). Therefore, at least for $\beta > 2.5$, already our 1st order approximation (as presented in this section),

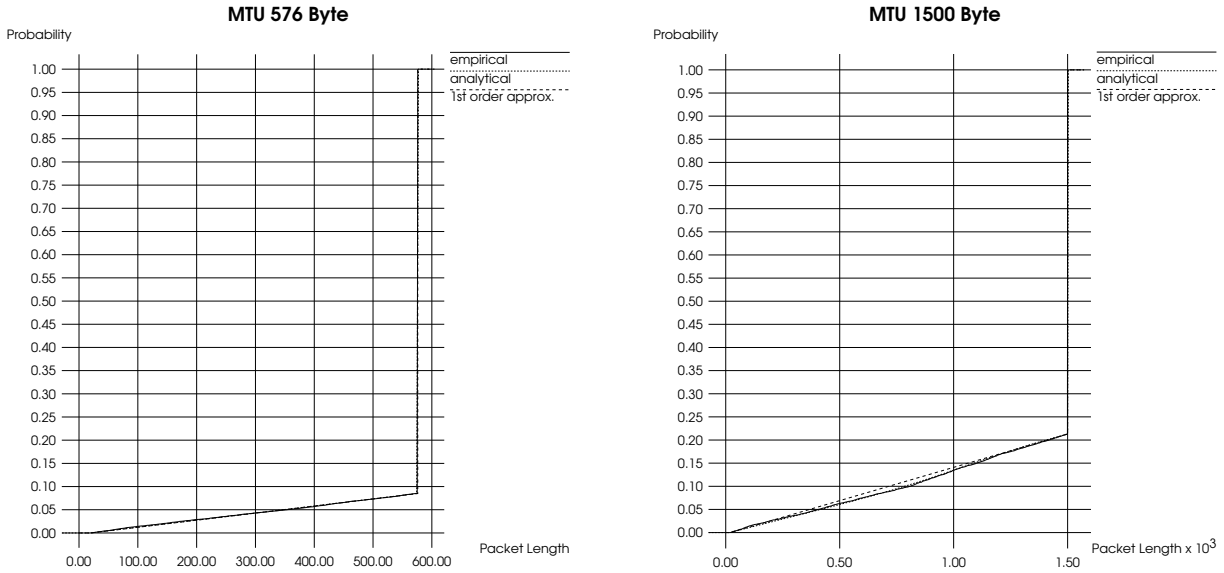


Figure 14: Comparison between measurement results and analytical modeling of secondary load (overlay of 30 sequences)

in most cases will provide astonishingly good and efficiently computable results for secondary load characterization in terms of IP packet lengths as observed in an Intranet or Internet subnetwork.

8 Summary and Outlook

In this contribution we have addressed the challenging problem of workload characterization and modeling for the Internet. Though quite a few researchers share the opinion that Internet can not be modeled at all, our view is slightly different. We agree that the Internet in its totality seems indeed to be much too complex to be modeled but we claim that adequately chosen subsystems or special aspects of the Internet can still be modeled in a sufficiently realistic manner, at least if we choose an appropriate level of abstraction.

The focal point of our paper concerns characterization of secondary load which would be induced by some arbitrarily chosen mix of primary load at some typical lower layer interface within Internet's protocol hierarchy. This approach seems to be much more flexible and should lead to more insight than characterizing secondary load in the conventional way, namely, approximating some randomly observed secondary load by means of a model without taking into account the mix of primary load present during the measurements. In many cases the mix of primary load which induced an observed secondary load is even not known at all.

We have introduced a generalized proceeding for load modeling including some formal description technique for load specification. The proceeding has been exemplified by modeling primary and secondary load in the context of Internet. For secondary load modeling we have applied our new approach to investigate transformations of primary load. The approach has been validated successfully by means of detailed secondary load measurements. As our

modeling of load transformation in many cases can be achieved just by applying mathematical methods directly to the probability distributions reflecting the attributes of primary load, such as message or packet lengths, our approach to load transformation seems quite promising.

The realistic load characterization, which is enabled by our load modeling approach, covering various interfaces within an IP based protocol hierarchy can be used in a straight-forward manner in combination with analytical or simulation models as well as a constituent of an experimental infrastructure for dedicated performance measurements regarding subsystems of the Internet.

Limitations of our approach concern, e.g.,

- the possibly high expenditure which may result in characterizing single sources of primary load in the Internet (in particular, as numerous types of sources exist and as sources may generate their load strongly dependent on the network's state);
- the mutual dependency which may exist between single sources of load, if we consider some complex primary load as an overlay of single sources;
- the possibility that the load generation process may be closely coupled to the state of the communication network (as is e.g. the case, if we try to characterize load as generated by TCP based senders); it is evident that, in case of TCP, load modeling can no longer be done by assuming an environment (comprising the TCP senders) which reacts independently of the underlying service provider (comprising the IP service), cf. also section 3;
- the complexity of some transformation processes which may act on primary load and which - due to their complexity - may not be easily modeled in a sufficiently realistic manner;
- the difficulty which may exist in determining a typical and representative mix of single sources of primary load.

Some of the load modeling problems mentioned may only be very hardly - if at all - solvable for the Internet as a global network in its full complexity. Nevertheless, we hope that our load modeling approach can and will be used to derive valid models for innovative communication networks including subsystems of the Internet. The application of such models would allow one to identify and possibly eliminate (some of the many) bottlenecks in parts of the Internet, to study - by means of modeling - the impact of changes in the Internet protocol stack (e.g. inclusion of protocols to support real-time communication) and, last not least, to investigate and prognosticate the behaviour of Internet subsystems for load situations to be expected in the future.

References

[1] ACM SIGMETRICS: Special Issue on "Network Traffic Measurements and Workload Characterization", *Performance Evaluation Review*, Vol.27, No.2 (1999)

[2] A. Albanese, S. Siemsglüss, B.E. Wolfinger: "Information Dispersal to Improve Quality-of-Service on the Internet", *Proceedings SPIE'98, Internat. Symp. on Voice, Video, and Data Communications*, Vol. 3529, Boston, November 1998, 14-25

[3] M. Arlitt, R. Friedrich, T. Jin: "Workload Characterization of a Web Proxy in a Cable Modem Environment", in: [1], 25-36

[4] M.F. Arlitt, L. Williamson: "Internet Web Servers: Workload Characterization and Performance Implications", *IEEE/ACM Trans. on Networking*, Vol. 5, No. 5 (1997), 631-645

[5] G. Bai: *Load Measurements and Modeling for Distributed Multimedia Applications in High-Speed Networks*, Uni Press Hochschulschriften, Bd. 107, 1999

[6] G. Bai, B.E. Wolfinger: "Possibilities and Limitations in Smoothing MPEG-coded Video Streams: A Measurement-based Investigation", *Proc. of MMB'97*, Freiberg, September 1997 (VDE-Verlag), 119-135

[7] M. Calzarossa, G. Serazzi: "Construction and Use of Multiclass Workload Models", *Performance Evaluation*, Vol. 19 (1994), 341-352

[8] A.B. Downey, D.G. Feitelson: "The Elusive Goal of Workload Characterization", *ACM SIGMETRICS Performance Evaluation Review* , Vol. 26, No. 4 (1999), 14-29

[9] W. Effelsberg, R. Steinmetz: *Video Compression Techniques*, dpunkt-Verlag, 1998

[10] R. Epsilon, J. Ke, C. Williamson: "Analysis of ISP IP/ATM Network Traffic Measurements", in: [1], 15-24

[11] A. Feldmann, A.C. Gilbert, P. Huang, W. Willinger: "Dynamics of IP traffic: A study of the role of variability and the impact of control", *ACM SIGCOMM'99 Conf., Computer Commun. Review*, Vol.29, No.4 (1999), 301-313

[12] A. Feldmann, A.C. Gilbert, W. Willinger: "Data Networks as Cascades: Investigating the Multifractal Nature of Internet WAN Traffic", *ACM SIGCOMM'98 Conf., Computer Commun. Review*, Vol. 28, No. 4 (1998), 42-55

[13] A. Feldmann, A.C. Gilbert, W. Willinger, T.G Kurtz: "The Changing Nature of Network Traffic: Scaling Phenomena", *Computer Commun. Review*, Vol. 28, No. 2 (1998), 5-29

[14] D. Ferrari: "On the Foundations of Artificial Workload Design", *Proc. ACM SIGMETRICS Conf. Measurements and Modeling*, Cambridge, (1984), 8-14

[15] V.S. Frost, B. Melamed: "Traffic Modeling for Telecommunications Networks", *IEEE Communications Magazine*, Vol. 32, No. 3 (1994), 70-81

[16] R. Jain, S.A. Routhier: "Packet Trains - Measurements and a New Model for Computer Network Traffic", *IEEE J. on Sel. Areas in Comm.*, Vol. SAC-4, No. 6 (1986), 986-995

[17] J.L. Jerkins, J. Monroe, J.L. Wang: "A Measurement Analysis of Internet Traffic over Frame Relay", in: [1], 3-14

[18] P. Karlsson, A. Arvidsson: "Traffic Modeling of TCP/IP over ATM", *IEEE Computer Communications Workshop*, October 1998

[19] J.J. Kim: *Formale Lastbeschreibung und eine Methode zur Lastmodellierung für innovative Kommunikationssysteme*, Verlag Shaker, Reihe Informatik, Aachen 1993

[20] L. Kleinrock: *Queueing Systems, Vol.2 - Computer Applications*, Wiley, 1976

[21] G. Kotsis, K. Krithivasan, S. Raghavan: "A Workload Characterization Methodology for WWW Applications", *Proc. Internat. Conf. on the Performance and Management of Complex Communication Networks*, Univ. of Tsukuba, 1997, 145-160

[22] P.J. Kühn: *Integrated Services Digital Networks - Basic Performance Modelling and Traffic Engineering*, Informatik-Fachberichte, No. 154, Springer-Verlag, 1987, 41-64

[23] A.A. Lazar, G. Pacifici, D.E. Pendarakis: "Modeling Video Sources for Real-time Scheduling", *ACM Multimedia Systems*, Vol.1, No.1 (1994), 253-266

[24] D. Liu, F. Huebner, Y. Levi: "A Hierarchical Multi-Class Traffic Model for Data Networks", *Proc. Intern. Teletraffic Congress*, ITC 16, Elsevier Science (1999), 1221-1229

[25] J. Magott, B.E. Wolfinger: "Formal Description Technique to Support Load Modelling for Innovative Communication Systems", *Applied Mathematics and Computer Science Journal*, Vol. 4, No. 4 (1994), 605-633

[26] D. Minoli, E. Minoli: *Delivering Voice over IP Networks*, J. Wiley & Sons, 1998

[27] C. Partridge: *Gigabit Networking*, Addison-Wesley (1994)

[28] P.F. Pawlita: "Two Decades of Data Traffic Measurements: A Survey of Published Results, Experience and Applicability", *Proc. 12th ITC*, (1988), 230-238

[29] V. Paxson: "Automated Packet Trace Analysis of TCP Implementations", *ACM SIGCOMM'97 Conf., Computer Commun. Review*, Vol. 27, No. 4 (1997), 167-179

[30] V. Paxson, S. Floyd: "Wide Area Traffic: The Failure of Poisson Modeling", *IEEE/ACM Trans. on Networking*, Vol. 3 (1995), 226-244

[31] H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson: "RTP: A Transport Protocol for Real-time Applications", RFC 1889, IETF, Jan. 1996

[32] G.D. Stamoulis, M.E. Anagnostu, A.D. Georgantas: "Traffic Source Models for ATM Networks: A Survey", *Computer Communications*, Vol. 17, No. 6 (1994), 428-438

[33] W.R. Stevens: *TCP/IP Illustrated, Vol. 1: The Protocols*, Addison-Wesley, 1994

[34] A. Tanenbaum: *Computer Networks* (Third Edition), Prentice-Hall (1996)

[35] W. Willinger, M.S. Taqqu, R. Sherman, D.V. Wilson: "Self-Similarity Through High-Variability: Statistical Analysis of Ethernet LAN Traffic at the Source Level", *IEEE/ACM Trans. on Networking*, Vol. 5, No. 1 (1997), 71-86

[36] B.E. Wolfinger: "Characterization of Mixed Traffic Load in Service-Integrated Networks", *Systems Science Journal*, Vol. 25, No.2 (1999), 65-86

[37] B.E. Wolfinger, J.J. Kim: "Load Measurement as a Basis for Modeling the Load of Innovative Communication Systems with Service Integration", *Proc. of Second IEEE Workshop on Future Trends of Distributed Computing Systems*, Kairo (1990), 14-21

[38] W. Zhu: "Characterizing Wide Area Conversations on the Internet", Master of Science thesis, Dep. of Computer Science, Univ. of Saskatchewan, Canada (1994)