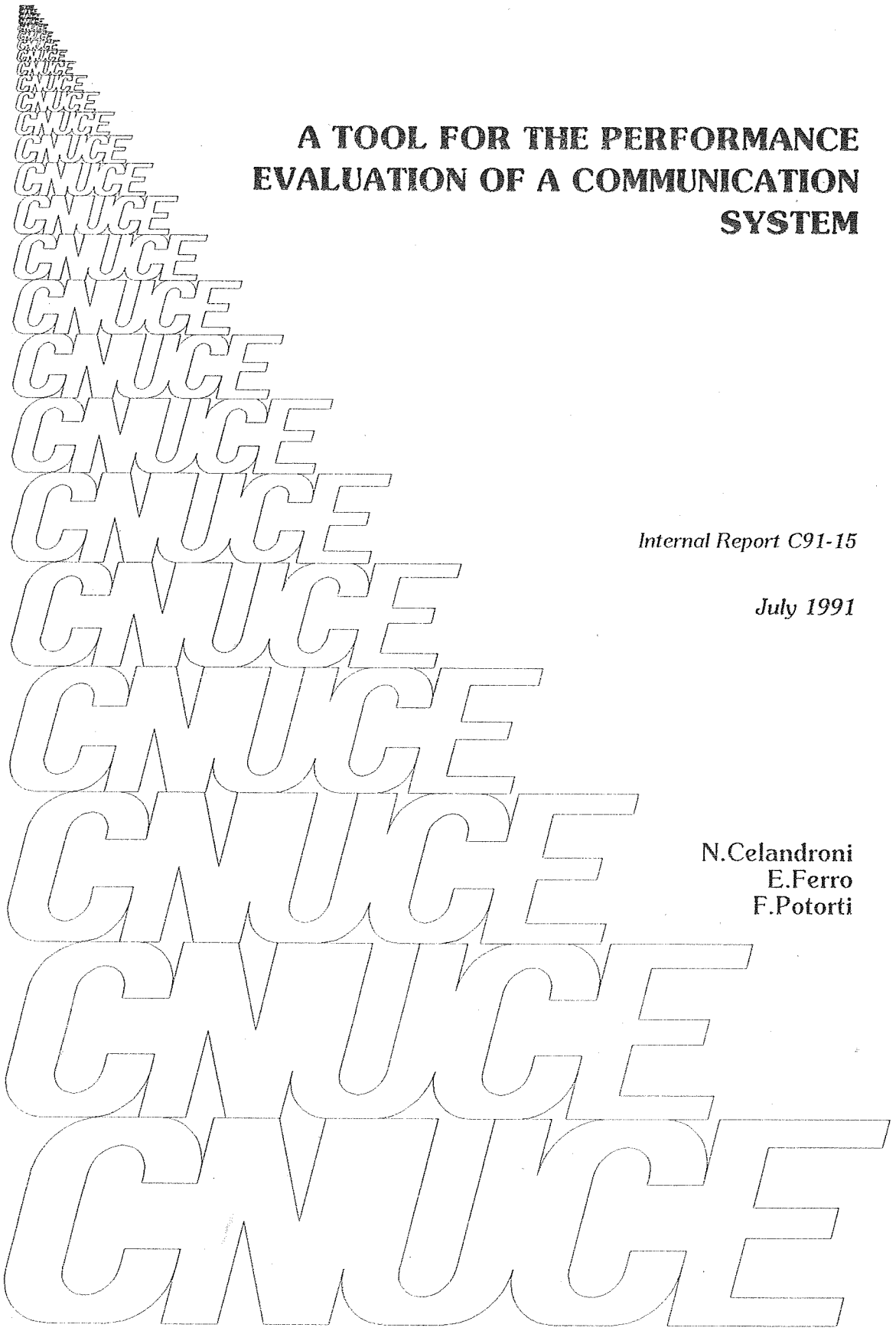
A TOOL FOR THE PERFORMANCE EVALUATION OF A COMMUNICATION SYSTEM



Internal Report C91-15

July 1991

N.Celandroni
E.Ferro
F.Potorti

A tool for the performance evaluation of a communication system(*)

N. Celandroni (+), E. Ferro (+), F. Potortì (°)

(+) CNUCE, Institute of National Research Council
Via S. Maria 36 - 56126 Pisa (Italy)
Fax: +39-50-576751
Telex: 500371
Phone: +39-50-593207/593312/593203
(°) Graduated in Electronic Eng.
Telespazio S.p.A. scholarship-holder

Abstract

A system is presented that generates traffic on a communication medium (LAN/MAN/WAN).

The generated traffic simulates the traffic produced by a number of both synchronous and asynchronous applications. The system, called Multimedia Traffic Generator (MTG), allows the study of the real-time problems relevant to the communication networks. Data generated by MTG are equivalent to data generated by real applications spread over a network interconnected to a Communication System Under Test (CSUT). CSUT can be as general as possible, even if the original idea of designing the MTG arose from the need to test a complex satellite access scheme.

The MTG system is described, its performance is shown and a particular application is presented.

1. Introduction: an overview of MTG

Testing a complex communication system in a real environment poses a problem with regard to the characteristics and the reproducibility of the traffic feeding the system during the performance measurement procedure. MTG is aimed at generating and recording the data packet traffic on a Local Area Network (LAN) in controlled and reproducible conditions. The generated traffic is not distinguishable from the traffic generated by a number of real independent sources spread out over a LAN. Each of these traffic sources can generate *stream* or *datagram* traffic. Stream traffic (like voice, slow scan TV, videoconferencing, etc) is characterized by a constant packet inter-arrival rate and requires short and fairly constant delay. It cannot tolerate out-of-order packet delivery but can tolerate occasional bit errors and dropped packets. Datagram traffic is sub-divided into *bulk* traffic and *interactive* traffic. Bulk traffic (typically bulk data transfer) requires a large number of packets to be transmitted. The speed should be adequate to the application capability and the delay introduced by network(s) crossing is not a critical constraint. It has not the rigid requirements of speech and, provided that a reliable end-to-end protocol is running in the hosts, out-of-order delivery packets may be allowed. Interactive traffic (terminal access to computers, database enquiry, operator messages

(*) Work carried out under the financial support of the National Research Council (C.N.R.) in the framework of the Telecommunication Project.

exchange, etc.) demands reliable delivery and short delays to guarantee almost real-time responses. It often consists of short messages (a few characters) with unpredictable inter-arrival rates.

MTG is able to produce traffic that is the result of the activity of a number of schematically user-defined independent sources called *Traffic Generators* (TGs), whose behaviour is defined by a set of characteristics. The TG concept allows the modelling of a lot of traffic patterns, as it can be found in a real environment.

All TGs are fully described in a common input file by the listing of their characteristics. They work concurrently during the run: the traffic generated by MTG is the sum of the traffics generated by each TG.

The MTG program consists of two distinct sections — called *controller* and *driver* respectively — interacting each other via a well-defined interface. The controller solely depends on the communication protocol with CSUT, while the driver is tied to the LAN and it is independent of the communication protocol. This organization gives some flexibility to the MTG system: when the same CSUT has to be tested over different LANs or MANs (Token Ring, FDDI, etc.), the same test conditions can be achieved with different types of network by just changing the driver section.

MTG is able to evaluate the performance of a communication system it is connected to in full duplex mode. Data generated by MTG are flagged as such and the CSUT is supposed to allow these particular packets to cross through itself. These packets are looped back and received by MTG after having crossed the entire communication system (*closed-loop configuration*). Some characteristics of the communication system, such as packet delay, jitter, bit error rate (BER), etc. can be measured using this procedure.

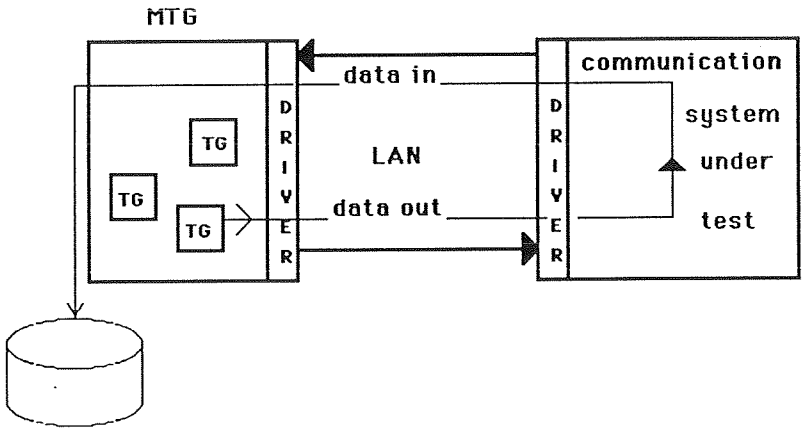


Fig. 1. The closed loop configuration connecting the MTG with the CSUT

2. The MTG header

MTG allows the monitoring of a set of quantities, such as packet delays and jitter, BER, communications channel parameters, etc. The statistics of these parameters are used to test and to tune up the communication system under test.

A communication protocol, dependent on CSUT, is supposed to allow the connection between the MTG and the CSUT.

A special *MTG header,* following the communication protocol, has been added to each packet to allow some parameter recordings (Fig. 2).

All the fields described in Fig. 2 are used when CSUT is in closed-loop configuration.

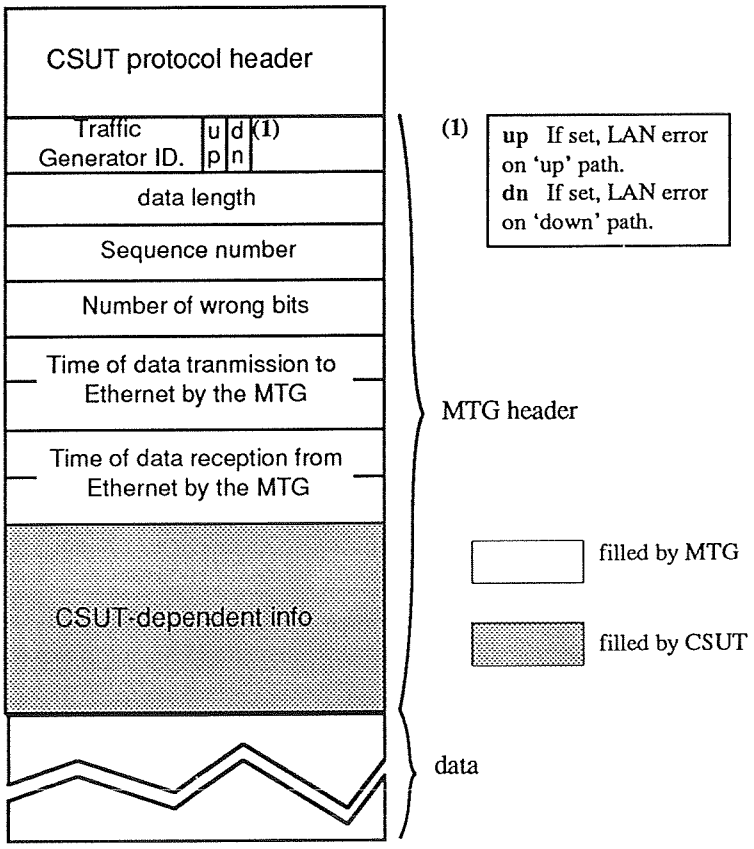


Fig. 2. Structure of a data packet sent from the MTG toward the CSUT

The *traffic generator id* allows all the measurements separately for each TG.

The *sequence number* allows the detection of packet losses.

The two bit fields - *up* and *dn* - are used to flag unusual errors on the LAN.

Both the *number of wrong bits* and the *data length* allow the estimation of the BER relevant to the CSUT. The packets flagged as "corrupted" by the LAN must not be kept into account.

The *timestamps* are expressed in *microticks*, the time unit internal to the system (currently set to 100 μs). The accuracy of the timestamps is one of the most strict requisites for the MTG system. As the CSUT time base and the MTG time base are generally not synchronized, the delays can only be expressed in terms of differences between timestamps coherent to the same time base. The timestamps are used to compute the CSUT+LAN round-trip crossing delay. The overall packet delay can be measured as difference between the two MTG timestamps. Valuable statistics include the delay and the jitter measurements either in stationary conditions or as a response to traffic transients.

In the presented application of the MTG (section 6), more timestamps in the CSUT reserved area are supposed to be filled by the CSUT itself.

For each received packet, MTG records the communication protocol and the MTG header onto disk. A subsequent elaboration of such data produces the requested statistics.

3. The TG parameters

A brief presentation of the TG parameters, shown in Table 1, is here given. The dotted parameters are relevant to the specific CSUT used; they will be described in section 6.

•	DATA TYPE:	Stream / Interactive / Bulk
	DISTRIBUTION:	Fixed Rate / Random / Poisson / Voice
	PACKET SIZE:	Value [bytes]
	MEAN THROUGHPUT:	Value [Kbit/s]
	PATTERN:	Fixed Pattern / Random / Incremental
	START TIME:	Absolute / Relative / Operator request / Poisson
	DURATION:	Absolute / Relative / Operator request / Poisson
	JITTER:	Value [ms]
•	COS:	Data Class of Service
	OPTIONS:	Options mask
•	ADDRESS:	Myself / To a particular station / Broadcast

Table I. Parameters defining a TG

Packets are generated by each TG according to the *DISTRIBUTION* parameter at an average rate specified by the *MEAN THROUGHPUT* and *PACKET SIZE* parameters. A *Fixed Rate* distribution means packets with constant length, sent at constant intervals of time. A *Random* distribution means packets of variable length, sent at constant intervals of time. The length is varied according to a pseudo-random variable with uniform distribution in a given range. A *Poisson* distribution means fixed-length packets, sent at intervals of time which vary according to a pseudo-random exponential distributed variable. A *Voice* distribution emulates voice packets with the silence detection feature. A fixed rate stream of packets is modulated ON/OFF, according to the talk-silence periods of the English speech distribution. More distributions, analytically or numerically defined, can be easily added to the ones so far foreseen.

A bit *PATTERN* can be defined for the data contained in a packet. Three pattern types are defined. The basic unit in the pattern is the 32-bit word. The *Fixed pattern* is a repetition of a given word. The *Random* pattern is a sequence of random, constant-distributed words. The *Incremental* pattern is a sequence of words in ascending order, starting from a given value.

START TIME and *DURATION* define the life span of a TG. A TG starts sending packets with the given distribution at the start time and goes on for a time interval equal to the duration. *Absolute* refers to a time given in hh:mm:ss format. *Relative* indicates a delay with respect to the start of the run. *Operator request* means that the TG is started/stopped by an operator command (the program is fully interactive). *Poisson* — used for the simulation of phone calls — indicates that either the start or the duration times are calculated according to pseudo-random variables following negative exponential distributions.

JITTER indicates that a pseudo-random variable, in the range of ±*Value*, can be optionally added to the sending time of each packet. This feature can be disabled by providing a null *Value* in the JITTER field. At present, only the uniform distribution has been implemented for such a jitter.

The *OPTION MASK* field is a bit mask used to specify some options, such as the setting of the *up* and *dn* bits in the MTG header, or the enabling of the real-time number count of wrong bits.

4. Implementation and real-time problems

A Motorola Delta 3300 single-board microcomputer running the Unix System V 3.5 has been used. It features the following characteristics:

- VME bus interface,
- Motorola 68030 µp and 68882 math co-processor running at 25 MHz,
- 8 Mbyte memory,
- two independent counter-timers with 6.25 µs time resolution,
- Local Area Network Controller for Ethernet AM7990(LANCE)- Serial Interface Adapter AM7992(SIA) chip set for Ethernet interface,
- SCSI interface,
- 1.5 Mbyte/s transfer rate, 16.5 ms av. access time disk drive.

The Unix system provides a comfortable environment to the programmer, but poses a fundamental obstacle because of its non real-time behaviour. Since System V currently provides non standard means to overcome this problem, a brute force approach has been taken. The devices concerned with the MTG (the LANCE and the timer) have been forced to generate non-maskable interrupts. This use of the interrupts allows full use of processor power, leaving the operating system free to carry out its work in the meantime. This is important, because MTG issues calls to the operating system when writing onto disk.

Real-time behaviour is vital to MTG because the packet dispatching scheduling times and the timestamps contained in the MTG header need sufficient precision. These timestamps allow an accurate estimate of the parameters under measure.

The controller section of the MTG starts by reading an input file containing all the descriptions of the TGs that will be active during the MTG run. It allocates a data structure for each TG and builds a Traffic Generator Table (TGT) for each one of them. The TGT is essentially a scheduling table containing the dispatching times and the lengths of the packets which must be generated by its associated TG. The time intervals between two consecutive packet dispatches are calculated according to the distribution and the mean throughput input parameters, and are stored in the TGT entries. Once the TGTs are built by the controller, the start and stop times of each TGT are calculated and two timers are set for each TG. When a *TG-start* timer expires, the relevant TGT is handed over to the driver section of the MTG which, in turn, takes care of sending the packets to the CSUT at the scheduling times specified in the TGT.

When the *TG-stop* timer expires, the controller kills the TG by commanding the driver to cancel the relevant TGT.

The controller is also in charge of the task of protocol handling with the CSUT. The controller builds the necessary control messages and sends them to the driver, while the received control messages are organized in a specialized *control-and-error queue* (different from the *data packet queue*) to achieve fast responses.

The controller continuously monitors the received data packet queue, whose headers are then recorded onto disk. It also monitors the keyboard for operator actions: in fact, it is possible

to have a real-time overview of the active TGs and their parameters, and to start or stop any TG.

The controller section of the MTG runs as a Unix user process, while the driver section is a Unix driver embedded into the Unix kernel. The driver is very specialized: no *read* or *write* operations are provided for general use. The controller gives commands to the driver via the Unix *ioctl(2)* system call and receives packets in a memory area mapped into its virtual space. The task of the driver is to read the TGTs created by the controller section and to send packets to the network accordingly, while respecting the scheduling times as much as possible.

The scheduling times are currently specified with a 1 ms (one *tick*) resolution, while the timestamps have a 100 µs (one *microtick*) resolution.

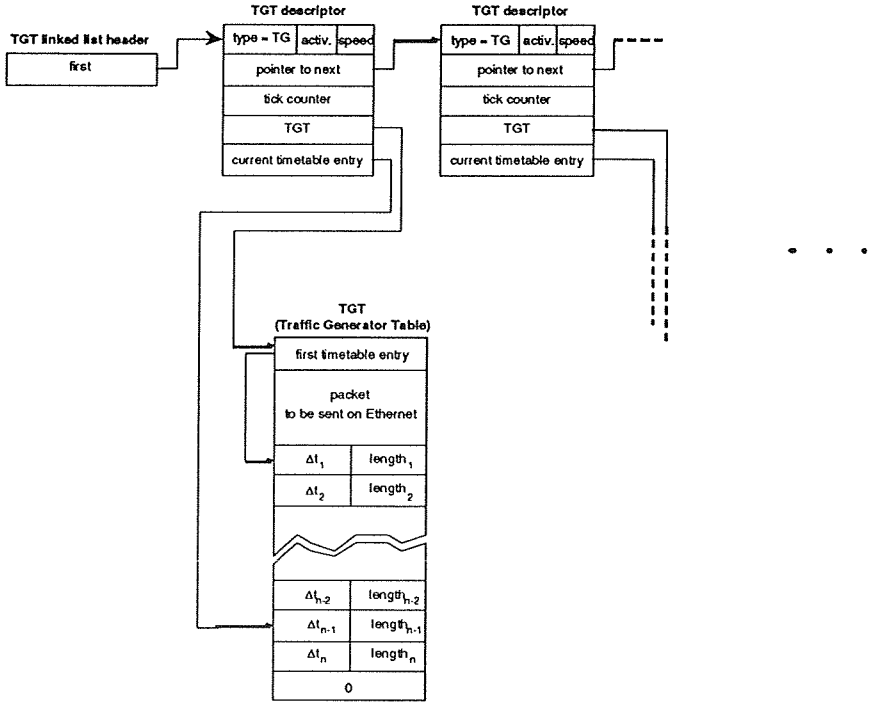


Fig. 3. The TGT linked list organization

Fig. 3 depicts the organization of the TGTs inside the driver. The driver gets the TGTs from the controller (*create* command) and acts on them according to the commands received by the controller. The commands are simple, such as *start*, *stop*, *kill* and *resume*. When the driver receives a start command, it just sets ON the *activity* bit in the indicated TGT. The driver scans the list once a tick, looking for packets to dispatch. When a TGT with the activity bit set ON is found, the *tick counter* is decremented by one. If the result is a non-null value, the next TGT is examined. At the end of the scan, all the tick counters have been decremented. When a tick counter reaches zero, the first *length* bytes of the packet relevant to that TGT are sent on the LAN. The pointer to the current *timetable entry* is then advanced (the timetable is thought of as circular) and the Δt field is copied into the tick counter field. The scan is then suspended until the occurrence of the EOT(1) interrupt. It should be clear from this description that packets to be sent are handled one at a time. This behaviour imposes an overhead of about 60 µs (CPU dependent) on each sent packet, but allows MTG to set accurate transmit timestamps.

(1) End Of Transmission

Both the controller and the driver are written in C language.

5. The MTG performance

Table 2 shows the MTG performance under different test conditions. A general CSUT was simulated by means of a responder which redirected the data it received to Ethernet. Each test was run for 300 seconds.

Ethernet was used both in simplex and in full-duplex. Neither the maximum throughput (10 Mbit/s) nor the maximum number of packets per second (14,880) allowed by Ethernet, when used in simplex mode, can be reached. This is due to the transmit side architecture of MTG. Anyway, the main purpose of MTG is to allow adequate accuracy in the parameters estimation, rather than to attain the top performance of the transmission medium in use.

number of running TGs	Inter-packet generating time for each TG	packet length (bytes)	total throughput in TX	total throughput in Rx	40 bytes recorded on disk every # packets	test run #
2	1 ms (496 Kbit/s for each TG)	64	992 Kbit/s	992 Kbit/s	1	1
3	1 ms (496 Kbit/s for each TG)	64	1.488 Mbit/s	1.488 Mbit/s	100.000	2
5	1 ms (1.040 Mbit/s for each TG)	130	5.20 Mbit/s	0 (no responder)	0	3
8	1 ms (496 Kbit/s for each TG)	64	3.96 Mbit/s	0 (no responder)	0	4
32	32 ms (69.5 Kbit/s for each TG)	22+256	2.224 Mbit/s	2.224 Mbit/s	1	5
90	32 ms (69.5 Kbit/s for each TG)	22+256	6.255 Mbit/s	0 (no responder)	0	6
43	62 ms (195 Kbit/s for each TG)	1514	8.385 Mbit/s	0 (no responder)	0	7
22	30 ms (404 Kbit/s for each TG)	1514	8.882 Mbit/s	0 (no responder)	0	8
12	16 ms (757 Kbit/s for each TG)	1514	9.084 Mbit/s	0 (no responder)	0	9
4	5 ms (2.345 Mbit/s for each TG)	1466	9.382 Mbit/s	0 (no responder)	0	10
1	1 ms (9.360 Mbit/s)	1172	9.360 Mbit/s	0 (no responder)	0	11

Table 2. Performance evaluation tests scenario

All the tests reported in Table 2 were carried out using an Ethernet LAN which connected the MTG and the responder. MTG collected the traffic looped back by the responder and recorded the first part of the packets onto disk. Each recorded packet header was 40 bytes long in these tests. The responder was not used in the tests where only the traffic generation performance was investigated.

Tests 1-4 were aimed at measuring the maximum number of packets per second the MTG can handle. Transmission only and transmission/reception with and without disk recording were the test conditions. The packet length does not influence the MTG performance, as long as an 'ideal' medium (infinite bandwidth) is used. A very small packet length was used to avoid collisions on Ethernet.

Tests 5 and 6 give the maximum number of supportable phone calls. Two results are given, with and without the packet header recordings onto disk. The packet length was set to 256 bytes plus a 22-byte header containing the Ethernet header and other CSUT-dependent information.

In the last five tests the maximum reachable total throughput was investigated with different numbers of running generators. The results are also plotted in Fig. 4. The packet length was chosen as long as possible, in order to reduce the overhead the MTG imposes on each transmitted packet. Fig. 4 shows clearly how the total throughput increases when the number of generators decreases, due to the added work required of MTG by each traffic generator. The throughput with only one generator is smaller because this test was performed with a shorter packet.

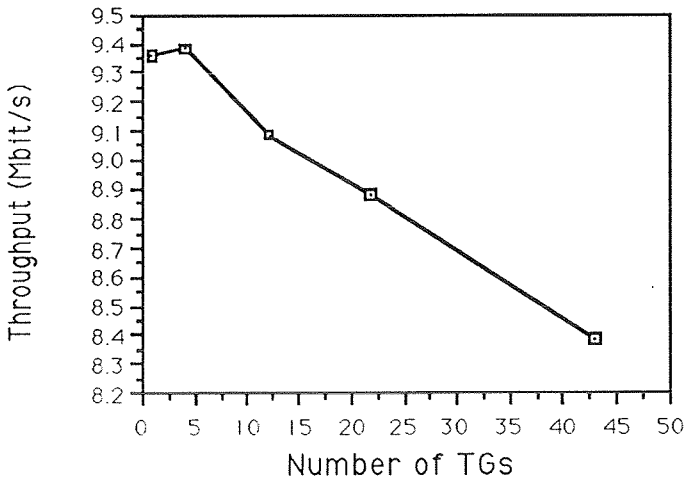


Fig. 4. Max throughput as a function of the number of TGs

The ability of the system to generate packets according to a chosen statistical distribution is shown in Figs. 5 and 6.

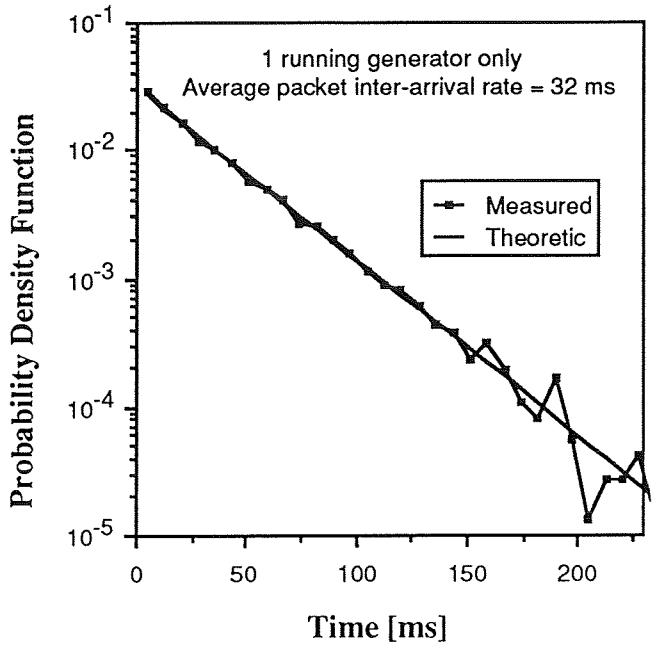


Fig. 5. Packet sending times distribution measured for 1 Poisson generator

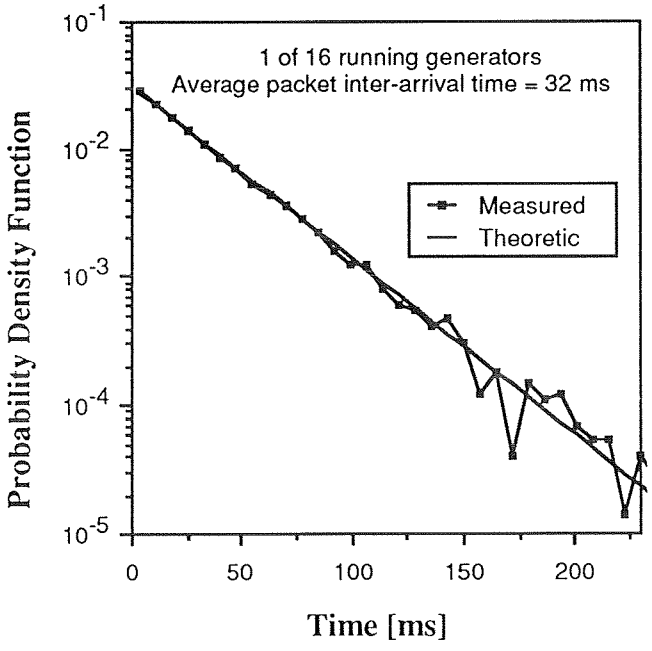


Fig. 6. Packet sending times distribution measured for 1 of 16 Poisson generators

The sending times of the packets generated by a Poisson generator with an average inter-arrival time of 32 ms (64 Kbit/s throughput) were collected and the probability density function was calculated using 10,000 samples (a run of about 300 s). The generator was running alone (Fig. 5) or together with 15 other generators (Fig. 6) of the same distribution type and with throughputs ranging from 16 to 256 Kbit/s. It can be seen that the offsets from the theoretic curve (a straight line in a logarithmic scale) are limited as far as the significant portion of the distribution is concerned. Furthermore, it is evident that the presence of many generators does not make the performance sensibly worse.

It must be pointed out that the ideal LAN to be used is an unlimited bandwidth LAN. The scheduling times of the packets generated by each TG should be those ones computed by the system. Therefore, all the effects introduced by the LAN, such as delays, errors or collisions (if Ethernet is used), are undesired effects, even though these effects can be easily filtered during the collected data analysis. The effects of the non-ideal LAN become significative only when the LAN itself is loaded beyond a certain level. A further increase of the throughput would compromise the measuring method validity anyway. When such a high throughput is requested, our implementation (based on Ethernet) would require the introduction of one more Ethernet board in the MTG hardware and the adoption of two LAN stubs to be dedicated one to transmission and one to reception.

6. An MTG application

The MTG system must be viewed as a general LAN traffic loader/recorder, with fine tuneable characteristics. It was designed as an evaluation tool for the performance of the FODA-TDMA[2] system, a demand-assignment satellite access scheme working at variable bit and coding rates. Literature describing the FODA system can be found in the references, anyway a short account is here given.

A control station is in charge of maintaining the synchronism among the earth stations and of assigning, on request, the transmission time slots. A reference burst is sent by the control station at the beginning of fixed size frames.

Both stream and datagram traffic types are supported by FODA. The requested bandwidth on the satellite channel (one slot per frame) is granted to a stream data source, until an explicit indication is sent by the requesting source itself. The sum of all the stream allocations must not exceed an upper threshold. The remaining amount of bandwidth is reserved for datagram traffic. Datagram allocations are made dynamically, according to an algorithm which equally penalizes all the stations, in terms of queuing delay, regardless of the individual station load. The requests for datagram assignments are sent to the control station at regular intervals of time. The requests are made on the basis of both the amount of data, already enqueued, to be sent on the channel (backlog) and of the instantaneous incoming traffic: $request = backlog + K\ traffic$ where K is a constant factor.

The last term allows a null queuing delay in certain conditions [1, 6] and causes the system to react to transients with better performance. The spare bandwidth, if available after all the allocations, is evenly shared among the active stations. In this case the channel is scarcely loaded and this last feature allows the system to absorb some traffic transients, without increasing the delays.

The access scheme also allows the bandwidth reserved for the stream traffic to be used to send datagram when, inside the same earth station, a stream source has empty periods of

---

[2] Fifo Ordered Demand Assignment - Time Division Multiple Access

time. This happens, for example, during digital speech transmissions with the silent detection feature. The bandwidth reserved for the stream source is not wasted if there is some other datagram source waiting for transmission bandwidth.

Each data source requests a class of service, i.e. a range of BER which must be guarantied during transmission. Maintenance of the requested class of service is attained thanks to the variable speed – variable encoding capabilities of the hardware used by the system. This feature makes the system able to counter fades of the signal due to bad atmospheric conditions, whose effects are particularly considerable when operating in the Ka band [7, 11]. The quality of the signal is continuously monitored [12], and the bit and coding rates of the transmitter are adapted — inside a certain range of values — to the fade conditions.

When a fade occurs, the data transmissions need more space inside the time frame to increase the energy per information bit ratio. This need is equivalent to the need of a wider bandwidth. In case of stream transmissions, a wider bandwidth must be explicitly requested. If not enough space is available in the FODA frame to allow this, two successive steps are foreseen. The stream threshold is first moved to a *special* value, then some stream applications are requested to work at a reduced speed. This is possible only for the applications that declared this feature possible.

In case of datagram transmissions, a fade condition causes an increment of the backlog and of the instantaneous traffic. This automatically increases the station request, making an attempt to get more bandwidth. Due to the increase of the request itself and/or to the eventual compression of the datagram bandwidth to stand by stream applications, the overall information capacity of the datagram may be sensibly reduced. It occurs particularly under heavy loading conditions of the system. An efficient action of the channel saturation control system [1] is however requested to avoid congestion.

At the writing time, only a version of the system working at fixed bit and coding rates does exist. The hardware able to support variable bit and coding rates will be available shortly and the new version of the access scheme, relevant to the new hardware, is still under development. The FODA system is a part of an experiment of LAN interconnections via satellite in the framework of the Olympus Utilization Program.

In our application of the MTG system, a special communication protocol (GA-FO protocol [8]) is included in the MTG to allow connection with the FODA system via an Ethernet LAN.

As far as the TG parameters are concerned, the dotted ones of Table 1 are here described.

DATA TYPE allows the selection of the stream type traffic and of two different classes of datagram traffic (interactive and bulk). FODA gives different transmission priorities to the data belonging to these two datagram classes. Stream traffic requires that the needed bandwidth be granted by the FODA system before sending data. On the contrary, the bandwidth needed by the datagram traffic is automatically and dynamically assigned by the access scheme, according to the datagram assignment algorithm [4]. Therefore, data can be transmitted without any preliminary session.

The *COS* (Class Of Service) field is a number used to specify the range of the requested bit error rate [7], [11].

The *ADDRESS* field is the CSUT destination physical address of the data.

When CSUT is the FODA system, the communication protocol is the one named GA-FO [8]. So, the equivalent structure of Fig. 2 becomes the one depicted in Fig. 7.

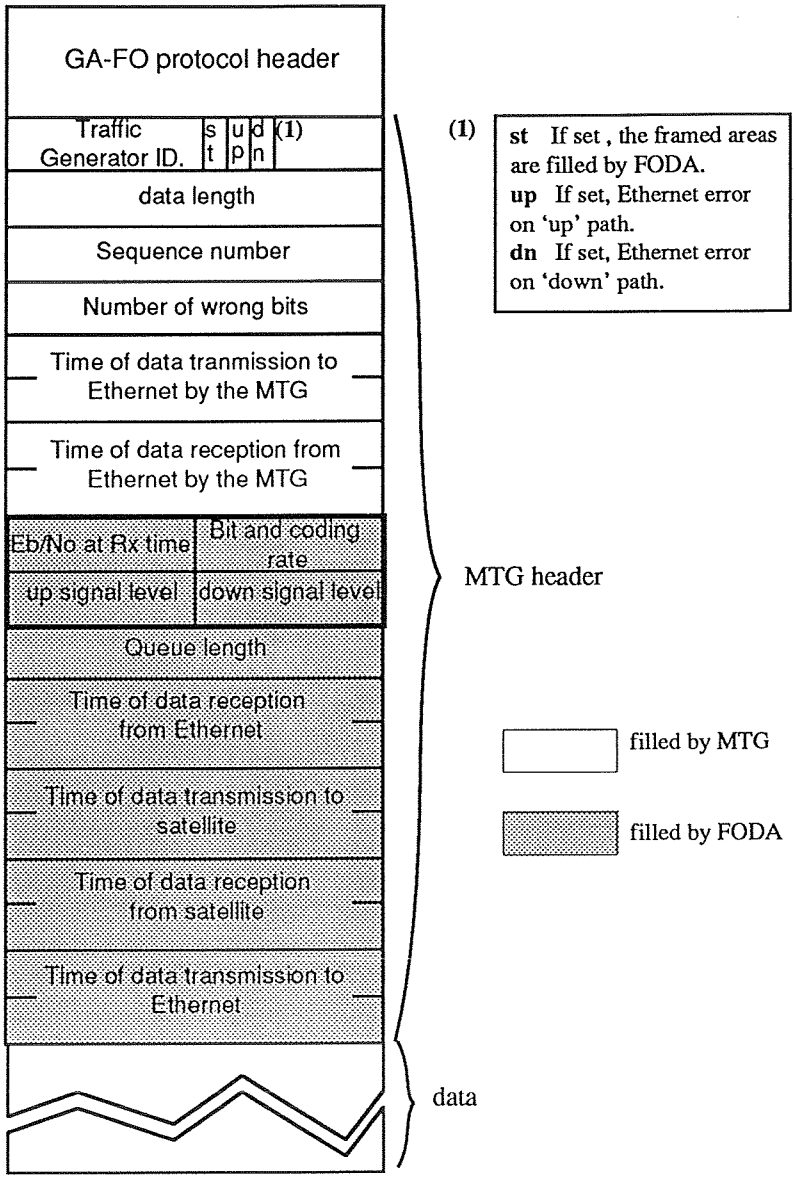


Fig. 7. Structure of a data packet sent from the MTG toward the FODA system

The four FODA timestamps allow the measurement of some important quantities, such as:

- the overall satellite network delay
- the queueing times of packets received by Ethernet and waiting for transmission to the satellite
- the queueing times of packets received by satellite and waiting for transmission to Ethernet
- the packet jitter.

Generally, it is possible to retrace the history of the delay and of the jitter, relevant to each packet, at different test points of the system on trial.

The framed fields (see note 1 on Fig. 7) contain information relative to the transmitting signal. They can be used to check if, under a fade event, the chosen bit and coding rates are suitably adapted to the signal to noise ratio, computable by the E_b/N_o and the down signal level

parameters. Moreover, the E_b/N_0 ratio and the up and down signal levels monitoring allow the tuning of the FODA method in order to detect the up and down-link signal attenuation.

Operations with more than one earth station are usually needed to test a multiple access scheme, such FODA. In these cases, it is necessary to synchronize the clocks of the various MTGs in order to get all plesiochronous events at the different sites. It is foreseen that to accomplish this task a master MTG must broadcast a special synchronization packet. When this packet is received, all the MTGs (including the master one), must reset their clocks (microtick counter).

Generally, data transmission and reception could be performed by different sites, but care must be taken to correctly use the recorded parameters. In fact, some of them are meaningless in this case.

7. Conclusions

All the features implemented in the above MTG system aimed at realizing a sophisticated tool to test, tune up and evaluate the performance of the FODA-TDMA system. The flexibility of MTG in supporting different communication media (changing the driver section) or different CSUT protocols (changing the controller section) with a small amount of work makes MTG a powerful general purpose tool. Moreover, many parameters of the MTG system, such as the accuracy of the timestamps and the granularity of the packet dispatch times, are easily changeable.

The Unix system has been chosen as a comfortable environment for the programmer. The non real-time characteristic of Unix does not affect the MTG performance whatsoever.

Acknowledgments

The authors wish to thank Eng. Izeta Delić for her precious collaboration on the MTG realization.

References

[1] Beltrame R., Celandroni N.
"The performances of the FODA access scheme: theory and simulation results"
CNUCE Report C86-19.

[2] Celandroni N., Ferro E
"FODA-TDMA satellite access scheme: description, implementation and environment simulation", proceedings of the Tirrenia International Workshop On Digital Communications, meeting on Satellite Integrated Communications Networks Tirrenia (Pisa), September 14-16 1987.

[3] Celandroni N., Ferro E
"FODA-TDMA satellite access scheme for mixed traffic. Implementation and testing features at 2 Mbit/s bit rate. Final Report"
CNUCE Report C88-10, April 1988.

[4] Celandroni N., Ferro E.
"The FODA-TDMA Satellite Access Scheme"
CNUCE Report C88-39, June 1988.

[5] Adams C.J., Burren J., Celandroni N., Ferro E., Hine M., Koudelka O., Waibel T.
"SATINE-II: an experiment in high speed wide area networking.
Part I: overview.

Part II:background and hardware development.
Part III: FODA-TDMA access scheme and software.
Part IV: tests and operations.",
proceedings of the European Teleinformatics Conference EUTECO '88 on Research into Networks and Distributed Applications.
Vienna (A), April 20-22, 1988.

[6] Celandroni N., Ferro E.
"The FODA-TDMA satellite access scheme: presentation, study of the system and results", waiting for publication on the IEEE Transaction on Communication magazine.

[7] Celandroni N., Ferro E., Marzoli A.
"Fade detection in the FODA system", proceedings of the Olympus Utilization Conference
Vienna (A), 12-14 April 1989.

[8] Celandroni N., Ferro E.
"Protocol between the FODA system and the outside environment"
CNUCE Report C90-03, August 1990.

[9] Celandroni N., Delić I., Potortì F.
"A Multimedia Traffic Generator for high speed network performance evaluation"
CNUCE Report C89-33, November 1989.

[10] Celandroni N., Delić I., Ferro E., Potortì F.
"The Multimedia Traffic Generator/Recorder. System description and user guide"
CNUCE Report C90-21, August 1990.

[11] Ferro E.
"A satellite network for good weather conditions and for high rain attenuation",
proceedings of the SBT/IEEE International Telecommunications Symposium ITS' 90
Rio de Janeiro, Brazil, September 3-6, 1990.

[12] Celandroni N.
"Estimation of the E_b/N_0 ratio using PSK quantized levels in an additive white gaussian noise (AWGN) environment"
CNUCE report C90-22, September 1990.