Chapter 7

SATELLITE SYSTEM SIMULATION

Techniques and applications

Franco Davoli, Mario Marchese
CNIT - Italian National Consortium for Telecommunications, University of Genoa Research Unit, Via Opera Pia 13, 16145, Genova (Italy)

Abstract: The chapter highlights techniques to simulate the behavior of real-time and non real-time satellite systems and clarifies possible application fields and motivations. Communication systems and, in particular, satellite components have increased their complexity. Simulation is a reality to explore new solutions without increasing hazards and costs: long phases of space missions are completely dedicated to forecast the behavior of the systems to be implemented and simulations are widely used. The chapter focuses on methods to evaluate the efficiency of a simulator, underlines some traffic models suited to simulate communication systems and explains particular topics where the authors have direct experience. A couple of tools are presented applied to the satellite environment: a C-language based non-real-time software simulator and a real-time simulator (emulator), where real machines can be attached to the tool, so as to avoid traffic modeling. Performance results are reported to better explain the concepts of the chapter.

Key words: real-time, satellite networks, performance analysis

1. INTRODUCTION

Satellite communications have many advantages with respect to terrestrial communications. On the other hand, they amplify also many problems already existing in terrestrial networks. The Quality of Service (QoS) issue is only an example of particular relevance in the satellite environment, involving the study of architectures, access schemes, management, propagation, antennas. In more detail, bandwidth allocation, which has been widely investigated in the literature concerning cabled networks, is of great importance. Many congestion control and resource allocation techniques have been applied to obtain a fair allocation of voice, video and data flows. The need of guaranteeing a certain QoS has allowed developing dynamic bandwidth allocation techniques, which take into account the current status of the channel. These works may represent a reference to design control schemes for satellite channels; nevertheless, the satellite environments are characterized by several peculiarities, which imply the introduction of suited control strategies. Differently from cables in terrestrial networks, satellite channels vary their characteristics depending on the weather and the effect of fading heavily affects the performance of the whole system [1], in particular for systems operating in the Ka-band (20-30 GHz) [2, 3]. Ka-band guarantees wider bandwidth and smaller antennas, but it is very sensitive to rain fading. The practical effect is on the quality of service offered to the users. Many user applications require a high degree of quality, and techniques to provide compensation for rain attenuation are needed. Among others, two methods adopted to provide compensation for rain attenuation are the use of extra transmission power in areas affected by rain, and the reservation of a portion of the system bandwidth to have a rain margin.

The latter may be differently implemented in dependence of the transmission system utilized. For instance, if a TDMA (Time Division Multiple Access) system is used, extra slots may be assigned either for re-transmission or for transmission with adequate redundancy in rainy areas. Other mechanisms with a similar effect may be used in FDMA (Frequency Division Multiple Access) or CDMA (Code Division Multiple Access) systems.

Another issue of topical importance concerns the transport layer: an essential quantity is the "delay per bandwidth" product. As indicated in RFC 1323 [4], the transport layer performance (actually the TCP [5] performance) does not depend directly upon the transfer rate itself, but rather depends upon the product of the transfer rate and the round-trip delay (RTT). The "bandwidth delay product" measures the amount of data that would "fill the pipe", i.e., the amount of unacknowledged data that TCP must handle in order to keep the pipeline full. TCP performance problems arise when the bandwidth delay product is large. In more detail, within a geostationary large delay per bandwidth product environment, the acknowledgement mechanism, described in detail in [6], takes a long time to recover errors. The propagation delay makes the acknowledgement arrival slow and the transmission window needs more time to increase than in cabled networks.

Another problem concerning TCP over satellite networks is represented by each loss, which is considered as a congestion event by the TCP. On the contrary, satellite links being heavily affected by noise, loss is often due to transmission errors. In this case TCP, which reduces the transmission window size, is not effective for the system performance.

While the large delay per bandwidth product mainly characterizes geostationary (GEO) links, transmission errors represent an important component of Low Earth Orbit (LEO) systems, which also require a fast data transfer, due to the limited time of visibility. The heterogeneous scenario, composed of LEO and GEO portions, is used as an example in this work.

The problem of improving TCP over satellite has been investigated in the literature since the end of the eighties [4]. More recently, references [7-11] summarize issues, challenges and possible enhancements for the transport layer (namely the TCP) over satellite channels. Reference [12] lists the main limitations of the TCP over satellite and proposes many possible methods to act. A recent issue of International Journal of Satellite Communications is entirely dedicated to IP over satellite [13]. Reference [14] proposes a TCP splitting architecture for hybrid environments (see also reference [15]). Also international standardization groups, as the Consultative Committee for Space Data Systems - CCSDS, which has already emitted a recommendation (reference [16]), and the European Telecommunications Standards Institute - ETSI [17], which is running its activity within the framework of the SES BSM (Satellite Earth Station - Broadband Satellite Multimedia) working group [18, 19], are active on these issues. In this perspective, an important indication is presented in the CCSDS File Delivery Protocol (CFDP) [20, 21], which is considered as a solution suitable for the communication systems at the application layer.

To improve the performance, the satellite portion of a network may be isolated and receive a different treatment with respect to the other components of the network: methodologies as TCP splitting [7, 9, 14, 15] and TCP spoofing [7, 15] bypass the concept of end-to-end service, by either dividing the TCP connection into segments or introducing intermediate gateways, with the aim of isolating the satellite link.

Considering also trends [22] in telecommunication systems, which affect the satellite environment, as the growing importance of services and the technology convergence for cabled and wireless communications, there is, on one hand, the opportunity of extending the use of satellite networks and the need of developing new instruments and schemes to improve the efficiency of the communications; on the other hand, it is important to observe the difficulty to test the solutions. A satellite system may be hardly studied on the field. Such testing is expensive and it often concerns only software components for earth stations. Alternatives are necessary to investigate new systems and to evaluate the performance. The first one is the analytical study. It is very attractive but complex. It often requires simplifications and approximations. The second alternative is non-real-time simulation. The behavior of a system is simulated via software. It is possible to by-pass the complexity of real systems, and solutions not yet technically feasible may be tested in view of future evolutions. The drawback is the need of modeling. A model is often not accurate enough to consider all the aspects of a real system. A third alternative, which seems to summarize most of the advantages of the solutions mentioned, is real time simulation (also called emulation). Emulation is composed of hardware and software components that behave as a real system. An emulator allows using real traffic and it is similar to the real system also from the point of view of the physical architecture.

The chapter presents some parameters to evaluate the efficiency of a simulator and the requirements to build a real-time simulator. The importance of traffic models is highlighted and a proposal to implement a real-time simulator is presented, as well as a short description of a non-real-time C-based simulator. Section II describes the evaluation parameters and the requirements of the emulator. Section III shows the proposal concerning the architecture and specifies the problems related to the real-time issue and the transport of information. Section IV reports some detail about the non-real time simulator. Section V shows possible application environments based on the experience of the authors and some results. Section VII contains the conclusions.

2. REQUIREMENTS

The design of a telecommunication network simulator (and, in particular, of a satellite simulator), heavily depends on the business field of users and on the scope of the simulation. Business modeling and service demonstration, for instance, require different characteristics than design or validation or performance evaluation. A tool useful for a University and a Research and Development Center is different from an instrument suited for a Satellite Provider, a Satellite Operator and a Small/Medium Enterprise (SME). The European Space Agency has recently developed a questionnaire [23] to review the future evolution of satellite systems simulation capabilities across Europe and Canada. *"For this reason parallel studies, focusing at satcom systems, have been initiated in the context of ESA New Media Support Centre, and are aimed at identifying current and future simulation needs and identify potential users, in order to prioritise the needs and derive the Terms of Reference and Capabilities of a set of system simulations facilities, taking into account the potential benefit to users"* [23].

The authors, in the following, have tried to summarize the requirements a satellite simulator [24], oriented to research and development and performance evaluation, should have, to match the requests of a large number of users with a special attention to the real-time issue.

Aim. The aim is the emulation of a real satellite network composed of earth stations and satellite devices including the satellite itself.

Scope. The emulation should range from Geostationary (GEO) to Low Earth Orbit (LEO) satellite systems. Earth stations may be connected to external PCs implementing algorithms oriented to the QoS (Quality of Service) at the network layer and new implementations at the transport layer. It should be possible to test different types of data link layers and different packet encapsulation formats at the data link layer.

Modeling. The statistics about losses and delays should take into account the real system and the status of the channels.

Transparency. The emulator should result as more transparent as possible towards the external world; it means that it should be seen as a real satellite device (e.g. a modem or a hardware card) by the external users (e.g. Personal Computers (PCs), routers, switches).

Scalability. The complexity of the overall system should be, as much as possible, independent of the enlargement of the emulated network. Adding a new component to the system (e.g. a new earth station) may increase the traffic and the computational load; but it should not affect the architectural structure of the emulator.

Traffic class support. The emulator should support different traffic classes at the MAC layer.

Reliability. The results obtained by the emulator should be as close as possible to the results obtained by a real satellite system that implements the same packet switching strategy.

Architecture. It is not necessary that the internal architecture of the emulator is a mirror of the real system from a physical and topological level (e.g. not necessarily each hardware component of the emulator should correspond to each device of the real system).

Simplicity. The emulator should result very simple from the point of view of the computational load.

Real time. The tool should work under stringent time constraints; in particular, at the interface with the external world.

Implementation. The implementation should use, as much as possible, hardware and software material already implemented in other projects.

Interface. The hardware interface towards the external PCs should be represented by devices (actually PCs, properly configured for this), called Gateways (GTW). The communication between the layers should be guaranteed by the use of exchanging Protocol Data Units (PDUs), where the information coming from the external PCs is encapsulated.

Core. The core of the emulator should be represented by a single tool, which imposes losses, delay and jitters, following a statistics, to each single PDU entering the emulator.

Transport of information. Each single PDU should be transported from the input to the output gateway.

QoS (Quality of Service). The PCs that utilize the emulator should be able to implement bandwidth reservation schemes and allocation algorithms to guarantee QoS to the users.

3. REAL TIME SIMULATOR

3.1 Revision of a Real System

A satellite system is constituted by a certain number of ground stations (each composed of a satellite modem that acts both at the physical and at the data link layer) and a satellite that communicates with the ground station over the satellite channel. The modem may be an independent hardware entity connected to other units by means of a cable (as in Fig. 1) or also a network adapter card plugged into a unit (e.g. the router itself or a PC), as in Fig. 2. In practice, it can be though as a data link layer of an overall protocol stack. For example, if an IP router is directly connected to the modem (Fig. 1), the IP layer of the router interacts with the modem by sending and receiving traffic PDUs. Whenever a satellite modem receives a PDU from the upper layers, its main function is to send it towards the desired destination. On the other hand, when a modem receives a PDU from the satellite network, it must deliver it to the upper layers. The emulator should allow testing various kinds of protocols, switching systems, and whatever else, in order to evaluate suitable solutions to be adopted.

It is possible to identify, in a real satellite system, the following main parts: a modem with an interface towards the upper layers (namely the network layer); a channel characterized by its own peculiarities; a data link protocol over the satellite channel and a satellite with its on-board switching capabilities.

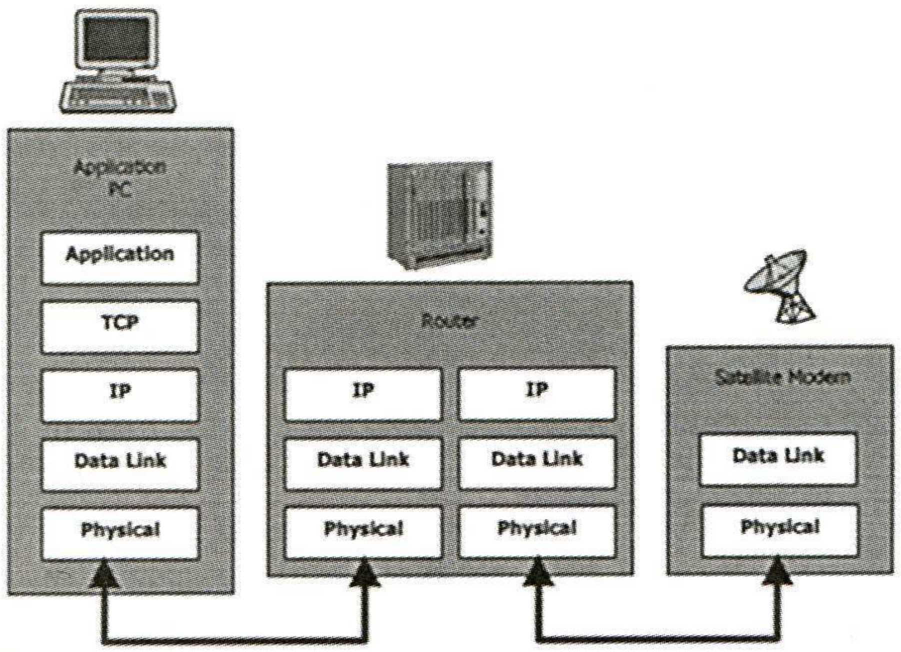

Fig. 1. Possible architecture of the real system - cable connection.

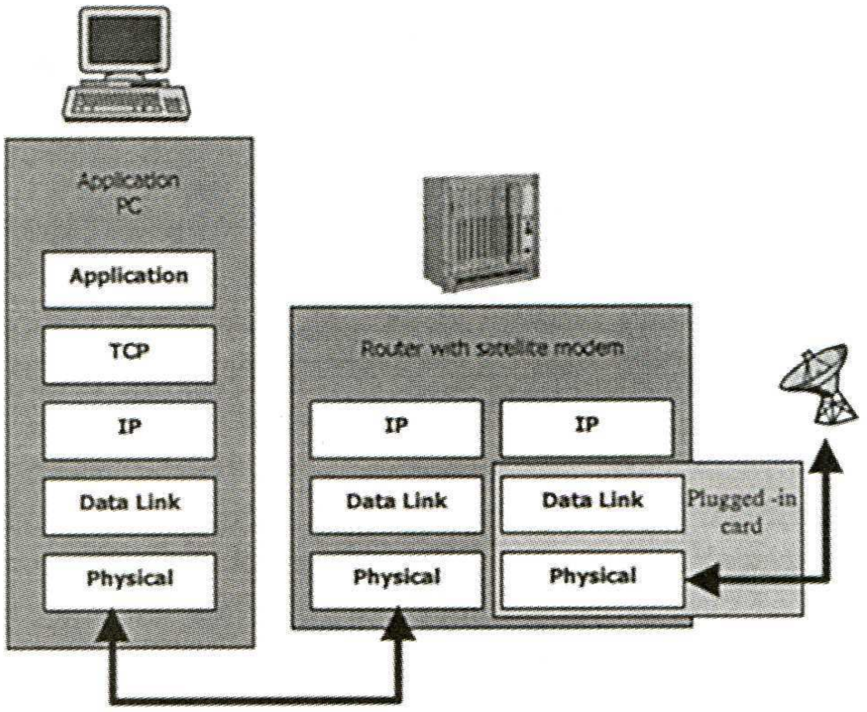

Fig. 2. Possible architecture of the real system - plugged-in card.

3.2 General Architecture

The reference architecture of the emulator is shown in Fig. 3, along with one possible system to be emulated enclosed in the cloud (a GEO satellite system has been depicted in this case). Different units called Gateways (GTW) operate as interface among the emulator and the external PCs. Each GTW is composed of a PC with two network interfaces: one towards the external world (a 10/100 Mbps Ethernet card), the other towards the emulator. An Elaboration Unit (EU), which has a powerful processing capacity, carries out most of the emulation, such as the decision about the "destiny" of each PDU.

The interface towards the external world concerns the GTWs; the loss, delay and any statistics of each PDU regard the EU; the real transport of the information PDU through the network concerns the input GTW and the output GTW. The various components are connected via a 100 Mbps network, completely isolated by a full-duplex switch. In such way, the emulator has an available bandwidth much wider than the real system to be emulated, which should not overcome a maximum overall bandwidth of 10/20 Mbps.

In more detail, Fig. 4 shows how the different parts of the real system (modem, data link protocol, channel and switching system, as mentioned in the previous sub-section) are mapped onto the different components of the emulator. It is clear in Fig. 4 that, as indicated in the requirements, the architecture of the emulator is not exactly correspondent to the real system. The earth station, identified by the gray rectangle, is divided, in the emulator, into two parts (GTW and EU). The network layer, the network interface towards the external world and the interface between the network layer and the satellite modem are contained in the Gateway (GTW). The other parts of the modem (i.e. the data link layer, protocol and encapsulation), the overall transmission characteristics (e.g. bit error ratio, channel fading, lost and delayed packets), the on-board switching architecture as well as the queuing strategies are contained in the Elaboration Unit (EU).

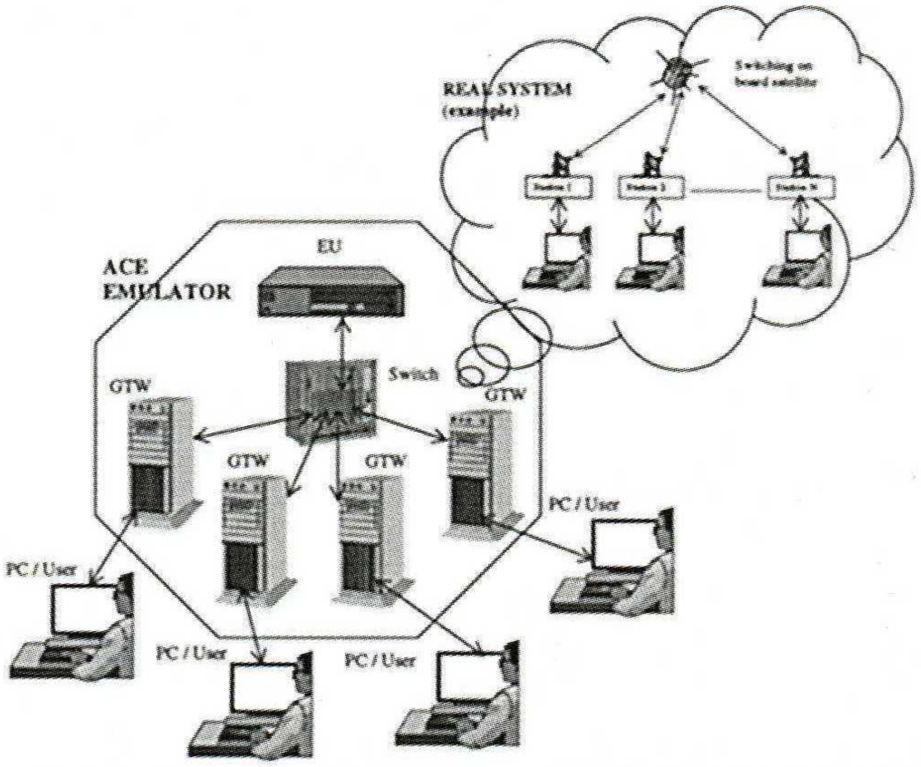

Fig. 3. Overall Emulator Architecture and Real System to be emulated.

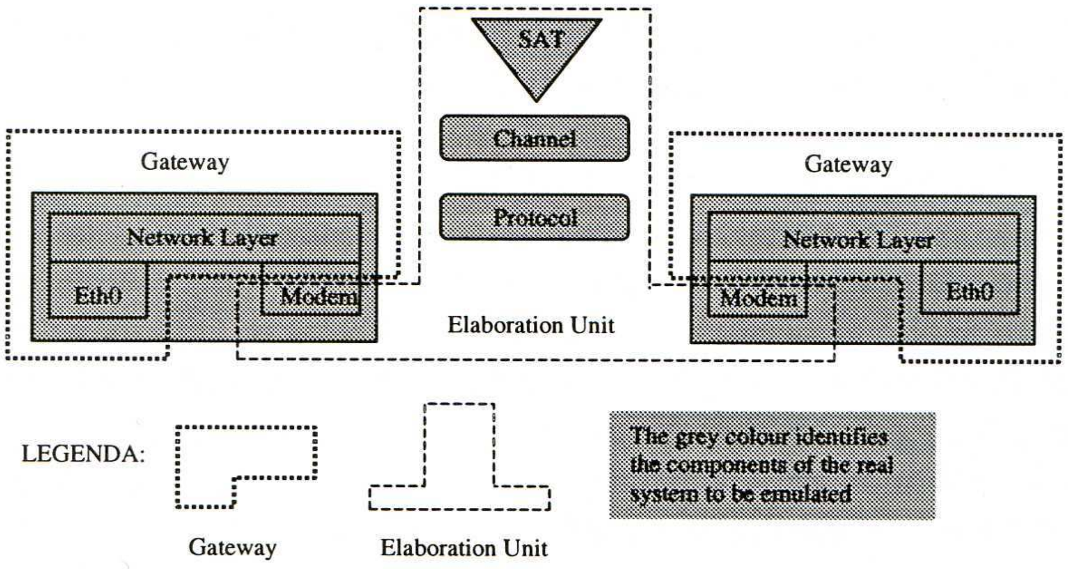

Fig. 4. Emulator versus Real System.

3.3 The interface

The interface between the modem at the ground station and the protocols of the upper network layers has been implemented in the emulator by creating a virtual device. It must be created on each of the GTWs and appears to the user and to the operating system as a network adapter (as a new Ethernet or Token Ring adapter). The Linux kernel, where the emulator is implemented, gives the possibility of creating such device by means of the *tun/tap* device [25]. It may be used in two possible ways: a point-to-point device (namely the *tun* device, suitable for a direct link between two PCs) and a broadcast, Ethernet-like device (namely the *tap* device, useful to connect many PCs as if they were connected to the same Local Area Network - LAN). The latter choice has been adopted in the emulator, because it is more similar to the real system structure. By the use of the *tap* device, the GTWs are actually connected by means of a virtual network that takes the PDUs and transports them as if they were transmitted over the real satellite system. After running the emulation software, a new network adapter (the *tap* device) is ready to be used as an Ethernet adapter, with its new 48-bit address. The new network adapter can be linked to any network protocol (such as IP, IPX, Novell). For example, if IP is used, an IP address shall be configured over the *tap* device and static routes or a routing daemon (e.g., *routed*) shall be started. Moreover, other IP configuration tasks may be performed, exactly as if the virtual device were the real satellite modem, which acts at the data link layer. Transparency is the real advantage of this solution. The approach allows the emulator to behave exactly as a broadcast link that connects several stations. It is not mandatory to use only one network layer protocol; any layer 3 protocol suitable to work with an Ethernet network adapter may be adopted.

3.4 Transport

The topic concerns the transport of information between the different components that constitute the emulator (i.e. GTW and EU). Two different kinds of information have been identified: the real traffic (i.e. a packet containing the PDU), which the emulator transmits between the Gateways, and the control information (namely a control packet), which is related to the computation of the emulation results (i.e. the "destiny" of each PDU). A traffic packet (e.g. *pkt_X*) is directly sent to the proper output Gateway. A control packet (e.g. *ctrl_pkt_X*), which contains concise information about the traffic packet *pkt_X*, is sent to the EU. The EU performs most of the emulation and, on the basis of given statistics describing the channel behavior, takes decisions about the traffic packet *pkt_X*. The EU reports them in *ctrl_pkt_X* and sends it to the output Gateway. The latter, upon reception of *pkt_X*, must store it until the arrival of *ctrl_pkt_X*. When the output GTW gets such information, it can properly act on the PDU by discarding it, by delivering it to the upper network layer at an exact time instant, by corrupting some bits or by performing other actions as indicated in the control packet. The same communication technique has been adopted for both types of information. A new protocol (called ACE, ASI CNIT Emulator) has been created in order to transport information. Due to the structure of the emulator (a set of directly connected PCs), the ACE protocol is encapsulated into the layer 2 PDU of the GTWs and of the EU. No routing is needed, because all PCs are directly connected among each other. The approach allows saving the amount of memory dedicated to contain the header bytes concerning the upper protocol address (20 bytes for an IP encapsulation). At the moment, Ethernet is the layer 2 protocol, but it may be changed by a small adaptation of the source code. Another advantage is the following. Once the traffic packet enters the ACE emulator (through the virtual interface implemented by the *tap* device), it does not pass through other protocol layers (e.g. IP) until it exits from the emulator at the output Gateway (through the *tap* device). It means that the ACE protocol is a layer 2 protocol. Traversing more layers (i.e. encapsulating ACE at higher layers) could overload the gateways, because the traffic should pass twice through the network layer: the first time as it would do in the real system (e.g. through an external router or through the source PC) and the second time when it is sent by the emulator software to the output GTW. In the working hypothesis made, the emulator must perform only a simple mapping between the layer 2 address of the virtual device (the *tap* device) and the address of the physical device (e.g. the Ethernet card connecting the units among them). The structure of an ACE packet is very simple: the first byte carries the identification code of the packet, while the other 1499 bytes carry the ACE packet header and payload. Only two kinds of packets have been created so far: the traffic and control packets, identified, respectively, with code identifier 1 and 2. If a PDU cannot be entirely contained into an ACE traffic packet, part of it may be sent to the destination through the control packet. In such a way only two packets are necessary to emulate a single PDU. If the payload size of the virtual device packet is larger than the physical device packet, then more than one physical packet should be used to send the PDU to the destination. Both interfaces are Ethernet in the present configuration of the emulator, so a virtual PDU of 1500 bytes can be divided into two parts: the larger one (nearly 1450 bytes) is sent directly to the destination, while the remaining part flows through the control packet. A comparison between the sequence of operations necessary to deliver a PDU from the input and the output GTW in real system and in the emulator is reported in Table 3.1.

Table 3.1 Comparison between Real System and Emulator operations

Real System	Emulator
The network layer sends the PDU to the input modem specifying a destination address	The network layer sends the PDU to the virtual device specifying a destination address
The input modem sends the PDU to the output modem through the satellite link	The virtual device (contained in the input GTW) collects the PDU; the input GTW resolves the physical address of the output GTW and sends the PDU to the output GTW by using the ACE protocol; the output GTW receives the PDU, stores it in a buffer and waits for an ACE control packet from the EU; the input GTW sends the ACE control packet to the EU; the EU receives the control packet, performs the necessary operations and sends another ACE control packet containing the "destiny" of the PDU to the output GTW
The output modem delivers the PDU (how and if the modem receives it) to the upper network layer	The output GTW receives the ACE control packet and delivers the PDU to the upper network layer or drops it, according to of the indications contained in the control packet

3.5 The Real-Time Issue

The emulation system should act under stringent time constraints at the external interfaces. The interfaces should be real-time devices. The emulator interfaces the upper layers of the protocol architecture in two different ways: collection and deliver of a PDU. The software component that carries out these tasks shall operate with precise timing. Otherwise, the results obtained by the emulator could be unreliable. For these reasons a real-time support has been inserted in the GTWs only for the operations of collecting and delivering of a PDU. Synchronization among all the emulator components is also necessary. A tool called Network Time Protocol (NTP) is used in the current implementation. On the other hand, it is not necessary that the operations performed inside the EU act under strict time constraints, but it is needed they provide the results in time to be applied to the real traffic in transit. The aim may be reached by optimizing the emulator code or by using more computing power in the Elaboration Unit.

4. NON-REAL TIME SIMULATOR

Most of the complexity of the tool presented in the previous chapter is due to the need to match real-time requirements. If there is no need to use real traffic and computers, a non-real time simulator, which is a simpler tool, may be used. The satellite network reported in the cloud of Fig. 3 may be simulated either by using commercial tools or by using software developed for the aim, as in the authors' case, who used this type of tool to get the results within a LEO/GEO environment within the DAVID satellite mission ([26], Data and Video Interactive Distribution, which has been promoted by the Italian Spatial Agency in collaboration with the University of Rome "Tor Vergata", the Polytechnic of Milan and CNIT, as scientific partners; Alenia Spazio, Space Engineering and Telespazio, as industrial partners) and concerning bandwidth allocation. A short description is reported below.

4.1 DAVID Mission Simulator

The tool applied within the LEO/GEO environment is a discrete-event network simulator, implemented in C-language, dedicated to study information transmission over heterogeneous architectures. The nodes that constitute the network can be hosts, routers and proxy units. Each terminal host is characterized by an application layer, a transport, a network and a data link layer.

The study where the simulator has been applied is related to data transmission mechanisms employed in satellite telecommunication environments composed of LEO and GEO orbits. A link in the W band connects the earth station to a LEO satellite (called DAVID). An inter – satellite link in the Ka band connects DAVID with a GEO satellite called ARTEMIS, which is linked to the destination earth station with a communication channel in Ka band. The visibility window of the LEO satellite is not continuous: when the satellite is no longer visible to the source terrestrial station, the transfer is interrupted and it can be restarted only at the next visibility window. The data transmission path is so structured: source earth station to LEO (when the LEO is visible from the earth), LEO to destination earth station through GEO (when LEO is in view of GEO). The return link necessary for the acknowledgment transportation is not always guaranteed on the GEO path. As a consequence, an effective protocol architecture is required in order to assure a reliable data communication over the sketched environment.

Several investigations about novel network architectures have been produced in order to individuate the solution that meets all the network requirements and characteristics in terms of delay, reliability and speed. Two types of solutions have been simulated: the first one, where the terminals are modified and no additional tool is inserted in the network; the second one, based on a protocol – splitting philosophy. This latter supposes to add special tools called gateways to improve the performance.

The work provided by the authors within the DAVID Project simulates concepts already known in the literature as transport layer splitting within the operative scenario characterized by the peculiarities described, introduces possible protocol architectures to be used and investigates and compares the behavior of protocols specifically designed for this environment (e.g. Satellite Transport Protocol, STP) and of protocols of the CCSDS family (e.g. CFDP).

Concerning the implementation, not each layer is described in detail (this constitutes the advantage with respect to the emulator), but it is modeled in dependence of what it is of interest. Application implies the implementation of both FTP [27] and CFDP protocols [20, 21]. TCP and its improvements (e.g., STP) are implemented at the transport layer, while IP is the network layer, which matches only the routing function within the simulator. Lower layers are much simplified and simulate only frame (packet) loss: bit errors uniformly distributed are supposed within each packet (i.e. the *i.i.d.* channel model [28]). BER (Bit Error Ratio) values ranging from 10^{-7} to 10^{-9} have been used in the tests. The percentage of packet corruption is dependent on the (BER), which has been fixed in each simulation. In detail, the packet error rate (PER) is considered $1-(1-\mathrm{BER})^{\ell}$, where ℓ is the packet length. If

BER<<1, as in the approach considered, the packet error rate can be approximated with $\mathrm{PER} \cong \ell \cdot \mathrm{BER}$. The simple model used tries to describe the behavior of the channel both at the physical and at the data link layer, for which no particular hypothesis has been done. The value of PER is aimed at describing the effect of low layers (physical and data link), seen by the network layer. This is the motivation for using term "packet". The performance metric is the throughput of the file transfers measured in bytes/s. It is defined as [dimension of file / overall transfer time]. Each protocol layer is modeled as a group of queues and servers to consider the flow between adjacent layers and possible congestion problems. An event routine is associated to each protocol layer and its code is started in correspondence of the related event (for example, arrival at the transport layer, processing at the network layer and so on). The description of the network and its topology is carried out by using parameters as adjacent node matrix, bandwidth information, BER, propagation and transmission time, which are read from a text file at the beginning of the simulation. Transmission is provided both by using just one connection and more than one connection at the same time (multi-connection case). Packets of the same length flow within the same portion of the network.

4.2 Bandwidth Allocation Simulator

This tool, whose structure is simpler than the previous one, is mainly characterized by the fade and its modeling.

Fade countermeasures are widely used in satellite systems, especially in Ka band [29, 30, 31]. Bandwidth allocation in the presence of multiple traffic classes and of different operating conditions at the earth stations (e.g., owing to diverse severity levels of the rain fading effect) is an important aspect that addresses the QoS issue in two ways. One is in terms of Bit Error Rate (BER), as the fade countermeasure is concerned; the other is in terms of service quality parameters, like call blocking and packet loss probabilities. Approaches that take both aspects into account have been proposed, among others, in [32, 33], where the fade countermeasure adopted is based on code rate (and, possibly, peak transmission rate) adaptation. Then, the effect of the fade countermeasure on guaranteed bandwidth traffic can be seen as an increase in the bandwidth that is needed to sustain a connection with the desired BER level. Without going into details, it can be mentioned here that the allocation can be made more or less "dynamic" (in the capability of responding to traffic or fading level variations), according to the information that is available at the decision maker. If real-time measurements of fading attenuation can be taken at the earth stations, the latter can either pass them along to a master station, which will then perform calculations on the bandwidth allocation, or directly compute the amount of bandwidth they would need to comply with their QoS requirements, and directly issue a bandwidth request to the master. In this case, a real-time control that attempt to closely follow variations in both fading and traffic load can be devised [33]. From the modeling and simulation point of view, two main aspects are of particular interest here: i) traffic models and ii) the use of real fading traces in the simulation. The latter is due to the high complexity and difficulty of modeling the rain fading process; the former may be addressed at different levels of accuracy for control and simulation purposes. In particular, the analytical/numerical derivation of a control strategy may be based on analytical traffic models (e.g., for the expression of quantities of interest, as buffer overflow probabilities [34]), whereas a detailed behavior of the processes generating the phenomenon of interest must be adopted in the simulation model for performance evaluation purposes.

5. APPLICATION ENVIRONMENTS

5.1 Real-time

The real-time simulator has been used to emulate the behavior of a real system. The test-bed satellite network is shown in Fig. 3. Real computers are connected through the emulator. Only two hosts (source and destination have been used) to simplify the validation process.

Validation

The performance offered by the emulator has been compared with a real satellite test-bed operating in the same conditions. The comparison has been carried out at the transport layer, which operates end-to-end and represents a fair measure of the system performance. The characteristics of the transport layer have been changed both to test the emulator in different conditions and to get meaningful research results within a hot research field. A short summary of the tests performed is reported in the following. The transmission instant for each data packet released to the network is the performance parameter, the tests are obtained by performing a data transfer of about 3 Mbytes.

In first instance the case of standard TCP employment is considered (supposing as standard TCP SACK New Reno [35]), when an initial transmission (IW) equal to 2 segments and a TCP Transmitter/Receiver Buffer of 64 Kbytes are assumed. The comparison between the emulator ACE and the real system is depicted in Fig. 5. The difference between the two cases is really minimal; the curves, describing how the packet transmission instant changes during the data transfer, just overlap.

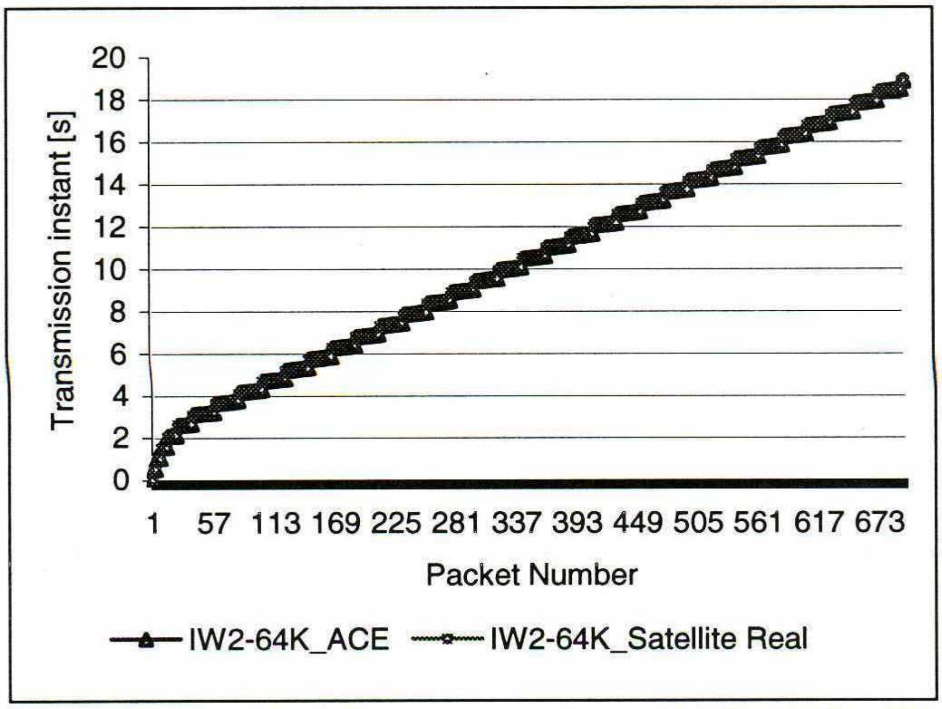

Fig. 5. Emulator and Real System: comparison with TCP (64 K – IW 2).

An important aspect of the analysis is how ACE emulator behaves when a more aggressive transmission mechanism is employed, i.e. varying the system conditions.

The case shown here, which is an example, concerns the employment of a modified TCP version, where an initial transmission window (IW) equal to 4 segments and a TCP Transmitter/Receiver buffer of 256 Kbytes are assumed. In this approach a faster filling of the channel pipe is expected, because of the increased initial transmission window and the bigger value of advertisement window, strictly dependent on the TCP receiver buffer. The comparison of ACE and real system behavior is shown in Fig. 6. The two curves completely overlap, as it happened in the previous case. This result is very significant, because shows the effectiveness of the emulation system also when a heavy load of data traffic is delivered to the client application.

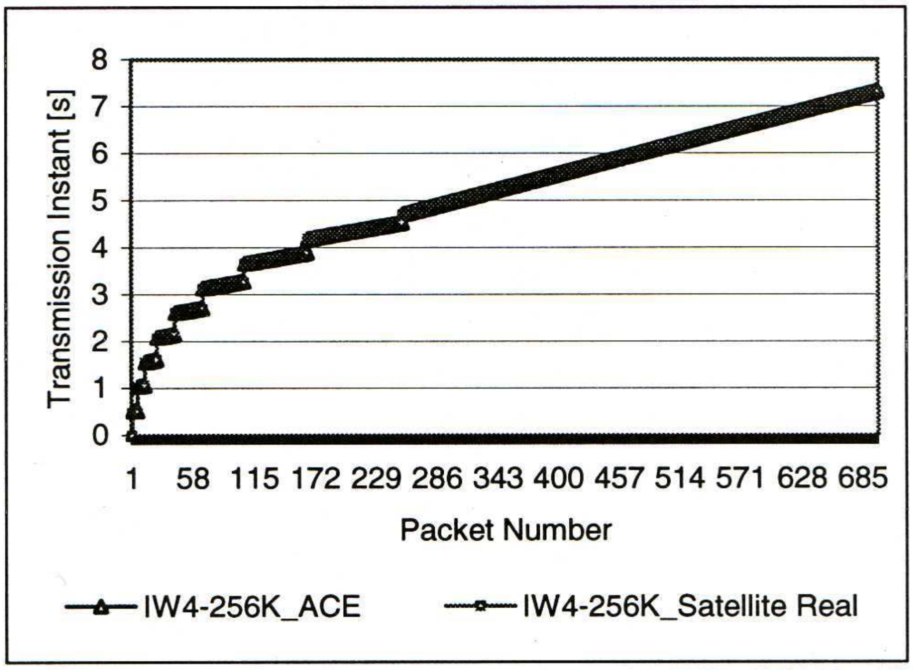

Fig. 6. Emulator and Real System: comparison with TCP (256 K – IW 4).

Application

In order to evaluate the system capabilities in a wider architecture, a more complex network configuration is considered. The aim is to check the effectiveness of ACE when several applications share the emulator. In this perspective the following applications are employed:

- FTP (File Transfer Protocol) communication
- SDR (Video communication achieved by using a proper Webcam)
- Ping

The general architecture employed is depicted in Fig. 7.

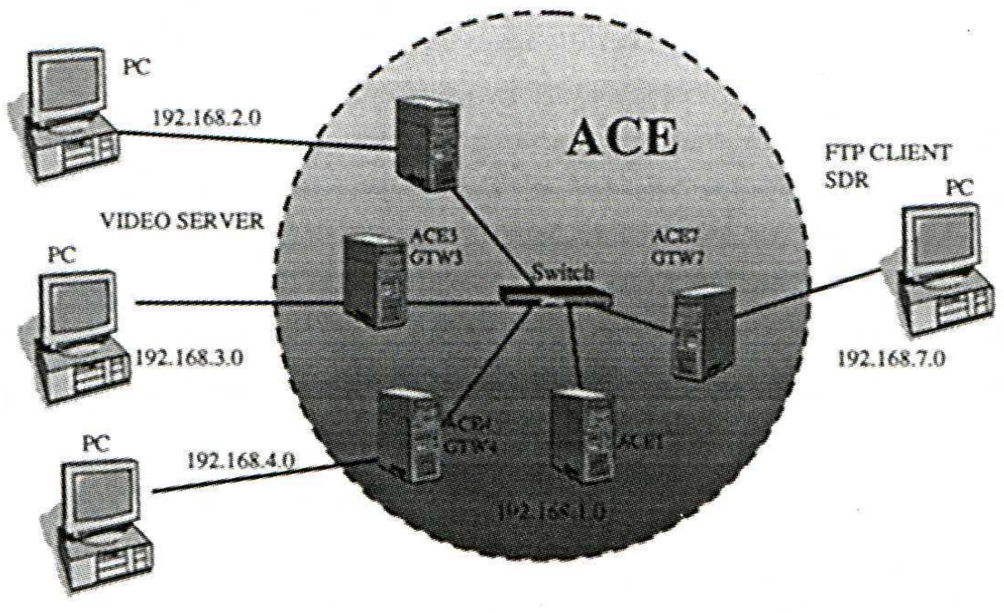

Fig. 7. Test-bed general architecture.

Four PCs are connected to gateways belonging to the emulation system; a further PC within the emulator is responsible of the emulation functionalities. All the machines are connected through a switch. The processes considered in the test-bed are resident on the PCs shown in the figure. It also important to spend some more words about the application employed. The three processes act at the same time and are considered to analyze how the whole system behaves when different real data communications are performed. In more detail, FTP is considered in order to evaluate the emulator performance when a big transfer of data ruled by the TCP protocol is accomplished. SDR allows showing the ACE behavior when a real – time service is required. PING is employed to verify that a correct RTT value, proper of Geostationary link, is set. A snapshot of the overall behavior is presented in Fig. 8, where different windows show the applications considered.

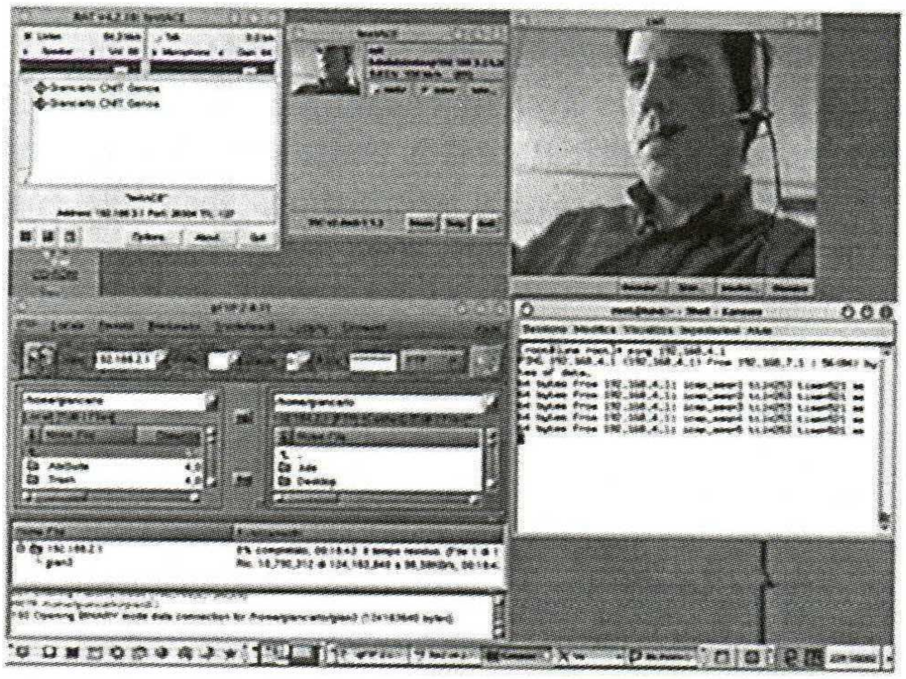
Fig.8. Application mask.

The windows on the left side describe the FTP session, the webcam application and allow measuring the emulator performance. The RTP - based real–time service and the TCP-based communication are accomplished at the same time and the emulator system is able to deal with them without introducing further delays during processing. On the right side there is the PING window showing that the system performs the correct Geostationary satellite RTT, set to about 520 ms.

5.2 Non-real time

DAVID mission

The operative environment simulated is depicted in Fig. 9.

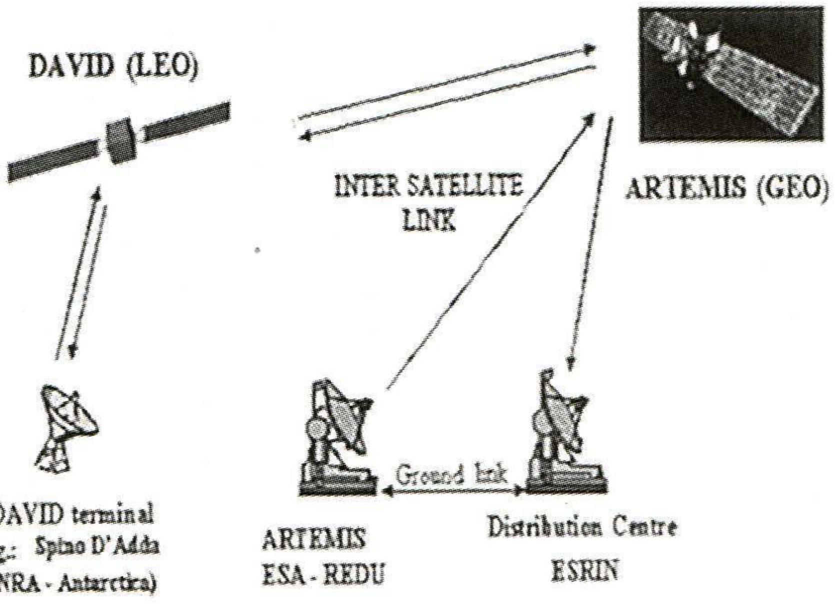
DAVID (LEO)
INTER SATELLITE LINK
ARTEMIS (GEO)
Ground link
DAVID terminal (e.g.: Spino D'Adda PNRA - Antarctica)
ARTEMIS ESA - REDU
Distribution Centre ESRIN

Fig. 9. DAVID operative environment.

Over the LEO link the W bandwidth is applied with 100 Mbps over the up-link channel and 10 Mbps over the downlink channel. The propagation delay is 5 ms. The GEO link is characterized by 2 Mbps and a propagation delay of 250 ms. The backward channel is not always available.

Concerning the LEO link, several solutions are compared by varying the bit error ratio (BER) of the channel. Solutions implemented within the terminal hosts are compared with architectures where an intermediate gateway is used to split the terrestrial portion of the network and the satellite link. In summary, "FTP –TCP 64K" and "FTP –TCP opt" indicate the two TCP – based solutions implemented within terminal hosts without adding any tool in the middle of the network; "CFDP – STP" and "CFDP – TCP opt" represent the two protocol – splitting solutions proposed with the CFDP protocol operating in unreliable mode. "CFDP reliable" is the protocol – splitting based solution with the CFDP protocol operating in reliable mode and communicates directly with datalink layer.

Fig. 10 summarizes the result. In this chapter the object is not to evaluate the efficiency of the solutions but only to show an example of application of the simulator.

Similar results could be shown concerning the GEO portion.

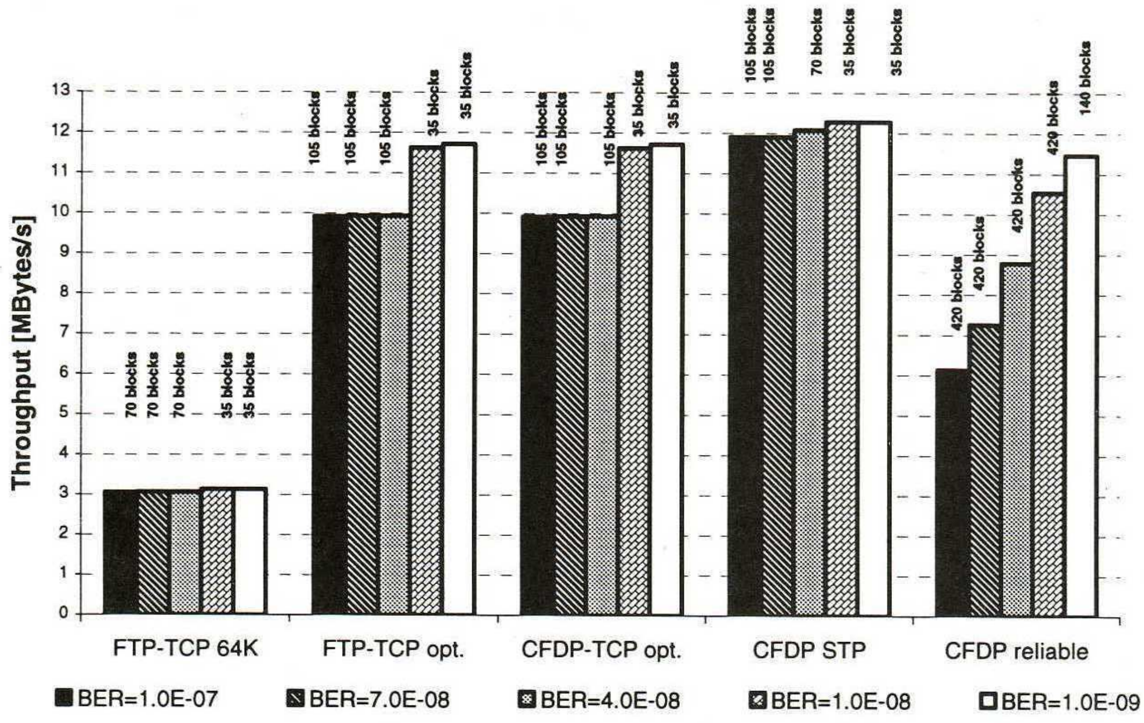

Fig. 10. Comparison of performance experienced by the different protocol architectures employed using non-real-time simulator.

Resource Allocation

In [33] a Multi Frequency – TDMA (MF-TDMA) satellite system with 10 stations has been simulated, in order to evaluate the performance of call admission control and bandwidth allocation strategies, in the case of two traffic types (guaranteed bandwidth and best-effort). The attenuation data have been taken from a data set chosen from the results of the propagation experiment, in Ka band, carried out on the Olympus satellite by the CSTS (Center for the Study of Space Telecommunications) Institute, on behalf of the Italian Space Agency (ASI). The up-link (30 GHz) and down-link (20 GHz) samples considered were 1-second averages, expressed in dB, of the signal power attenuation with respect to clear sky conditions. The attenuation samples were recorded at the Spino d'Adda (North of Italy) station, in September 1992. The fade situation at station 4 has been simulated by shifting in time the up-link fade of station 1 and the down-link fade of station 2, and by assuming that at station 4 the two fade events occurred simultaneously. As an example result, Fig. 11 shows the behavior of the bandwidth allocated to station 4 by the control strategy, together with the variations occurring in the fading pattern (β_{level} is a parameter that is derived from the fade attenuation through the link budget calculation, to express the reduction in bandwidth "seen" by a station as an effect of the additional redundancy that must be applied to maintain the BER level within the acceptable range (10^{-7} in the specific case)). Within this assigned bandwidth, the station performs admission control on the guaranteed bandwidth requests, and allows best-effort traffic to take the residual bandwidth, unused by guaranteed bandwidth connections in progress.

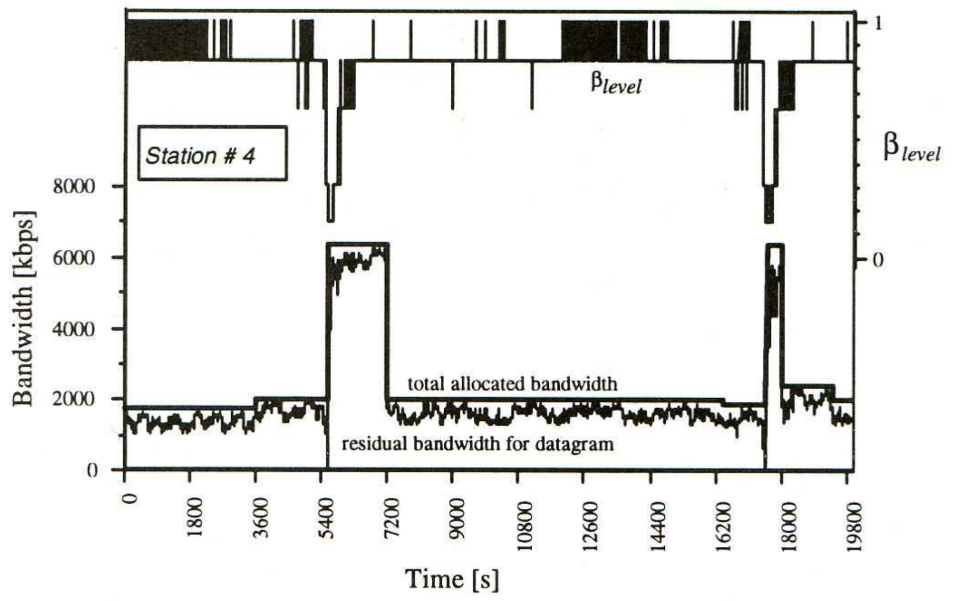

Fig. 11. Bandwidth allocated to station 4 by the control strategy – Variations of the bandwidth parameter caused by changes in the fading pattern.

6. CONCLUSIONS

The chapter summarizes the requirements and the characteristics of real-time and non-real-time simulators applied to the satellite environment, where the need of developing new instruments and schemes to improve the efficiency of the communications contrasts with the difficulty of testing the solutions, which is an action often expensive if performed in the field. Alternatives are necessary to investigate new systems and to evaluate the performance. Real-time simulation seems to be a proper solutions. It is composed of hardware and software components that behave as a real system and allows using real traffic so as to avoid traffic modeling. Anyway, the complexity of the implementation is not always justified by the aim of the tests and a non-real-time solution is recommendable. The chapter has compared the two solutions by presenting real cases where the alternatives are applied. The examples considered are: a real-time-simulator aimed at emulating a real satellite network composed of earth stations and air devices, including the satellite itself, mainly dedicated to test transport layer protocols and applications; two non-real-time tools dedicated, respectively, to the LEO-GEO heterogeneous networks and to resource allocation over faded channels. Results have been reported to better show possible applications of the simulators.

REFERENCES

1. Louis J. Ippolito, "Propagation Consideration for Low Margin Ka-Band Systems". Proc. Ka Band Utilization Conference, Italy, pp. 161-167, September 1997.
2. Richard T. Gedney e Thom A. Coney, "Effective Use of Rain Fade Compensation for Ka-Band Satellites". Proc. of Ka Band Utilization Conference, Italy, pp. 153-160, September 1997.
3. Roberto J. Acosta, "Rain Fade Compensation Alternatives for Ka-Band Communication Satellites". Proc. Fourth Ka Band Utilization Conference, September 1997. Italy, pp. 145-152.
4. V. Jacobson, R. Braden, D. Borman, "TCP Extensions for High Performance", IETF, RFC 1323, May 1992.
5. Information Sciences Institute, University of Southern California, "Transmission Control Protocol - Darpa Internet Program - Protocol Specification", IETF, RFC 793, September 1981.
6. M Allman, V. Paxson, W. S. Stevens, "TCP Congestion Control", IETF, RFC 2581, April 1999.
7. C. Partridge, T. J. Shepard, "TCP/IP Performance over Satellite Links", IEEE Network, Vol. 11, n. 5, September/October 1997, pp. 44-49.
8 T.V. Lakshman, U. Madhow, "The Performance of TCP/IP for Networks with High Bandwidth-Delay Products and Random Loss", IEEE/ACM Transactions On Networking, Vol. 5, No. 3, June 1997
9. N. Ghani, S. Dixit, "TCP/IP Enhancement for Satellite Networks", IEEE Comm. Mag., Vol. 37, No. 7, July 1999, pp. 64-72.
10. T. R. Henderson, R. H. Katz, "Transport Protocols for Internet-Compatible Satellite Networks", IEEE Journal on Selected Areas in Communications (JSAC), Vol. 17, No. 2, February 1999, pp. 326-344.
11. C. Barakat, E. Altman, W. Dabbous, "On TCP Performance in a Heterogeneous Network: A Survey", IEEE Comm. Mag., Vol. 38, No. 1, January 2000, pp. 40-46.
12. M. Allman, S. Dawkinks, D. Glover, J. Griner, T. Henderson, J. Heidemann, S. Ostermann, K. Scott, J. Semke, J. Touch, D. Tran, "Ongoing TCP Research Related to Satellites", IETF, RFC 2760, February 2000.
13. A. Ephremides, Guest Editor, International Journal of Satellite Communications, Special Issue on IP, Vol. 19, Issue 1, January/February 2001.
14. V.G. Bharadwaj, J. S. Baras, N. P. Butts, "An Architecture for Internet Service via Broadband Satellite Networks", International Journal of Satellite Communications, Special Issue on IP, Vol. 19, Issue 1, January/February 2001, pp. 29-50.
15. Y. Zhang, D. De Lucia, B. Ryu, Son K. Dao, "Satellite Communication in the Global Internet", http://www.wins.hrl.com/people/ygz/papers/inet97/index.html..
16. Consultative Committee for Space Data Systems (CCSDS), Space Communications Protocol Specification-Transport Protocol, CCSDS 714.0-B-1, Blue Book, May 1999.
17 European Telecommunication Standards Institute - ETSI, http://www.etsi.org.
18 TC-SES - Broadband Satellite Multimedia - Services and Architectures, ETSI Technical Report, TR 101 984 V1.1.1 (2002-11).
19 TC-SES - Broadband Satellite Multimedia – IP over Satellite, ETSI Technical Report, TR 101 985 V1.1.1 (2002-11).
20. Consultative Committee for Space Data Systems (CCSDS), CCSDS File Delivery Protocol, CCSDS 727.0-R-5, Red Book, August 2001.

21.Consultative Committee for Space Data Systems (CCSDS), CCSDS File Delivery Protocol "Part 1: Introduction and Overview", CCSDS 720.1-G-0.8, Green Book, August 2001.
22.H. Skinnemoen, H. Tork, " Standardization Activities within Broadband Satellite Multimedia", ", ICC2002, New York, April 2002, ICC02 CD-ROM.
23.http://www.esys.co.uk/surveys/ssu/default.htm
24.M. Marchese, M. Perrando, "A Packet-Switching Satellite Emulator: A Proposal about Architecture and Implementation ", ICC2002, New York, April 2002, ICC02 CD-ROM.
25 Universal TUN/TAP Device Driver, http://www.linuxhq.com/kernel/v2.4/doc/networking/tuntap.txt.html.
26 C. Bonifazi, M. Ruggieri, M. Pratesi, A. Solomi, G. Varacalli, A.Paraboni, E. Saggese, "The David Satellite Mission of the Italian Space Agency: High Rate Data Transmission of Internet at W and Ka bands", 2001 (http://david.eln.uniroma2.it).
27 J. Postel, J. Reynolds, " File Transfer Protocol (FTP)", IETF, RFC 959, October 1985.
28 L. Tassiulas, F. M. Anjum, "Functioning of TCP Algorithms over a Wireless Link", CSHCN T.R. 99-10, 1999.
29.F. Carassa, "Adaptive Methods to Counteract Rain Attenuation Effects in the 20/30 GHz Band", Space Commun. and Broadcasting, 2, pp. 253-269, 1984.
30.N. Celandroni, E. Ferro, N. James, F. Potortì, "FODA/IBEA: a flexible fade countermeasure system in user oriented networks", Internat. J. Satellite Commun., vol. 10, no. 6, pp. 309-323, Nov.-Dec. 1992.
31.N. Celandroni, E. Ferro, F. Potortì, "Experimental Results of a Demand-Assignment Thin Route TDMA System", Internat. J. Satellite Commun., vol. 14, no. 2, pp. 113-126, March-April 1996.
32.R. Bolla, N. Celandroni, F. Davoli, E. Ferro, M. Marchese, "Bandwidth Allocation in a Multiservice Satellite Network Based on Long-Term Weather Forecast Scenarios", Computer Commun., Special Issue on Advances in Performance Evaluation of Computer and Telecommunications Networking, vol. 25, pp. 1037-1046, July 2002.
33.N. Celandroni, F. Davoli, E. Ferro, "Static and dynamic resource allocation in a multiservice satellite network with fading", International Journal of Satellite Communications, Special Issue on QoS for Satellite IP, 2003 (to appear).
34.B. Tsybakov, N.D. Georganas, "Self-Similar Traffic and Upper Bounds to Buffer-Overflow Probability in an ATM Queue", Performance Evaluation, vol. 32, pp. 57-80, 1998.
35.M. Mathis, J. Mahdavi, S. Floyd, A. Romanow, "TCP Selective Acknowledgement Options", IETF, RFC 2018, October 1996.