INTERNATIONAL JOURNAL OF SATELLITE COMMUNICATIONS
Int. J. Commun. Syst. Network 2009; **27**:163–191
Published online 2 March 2009 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/sat.930

Performance evaluation of TCP-based applications over DVB-RCS DAMA schemes

M. Luglio, Cesare Roseti*,† and F. Zampognaro

Electronics Engineering Department, University of Rome 'Tor Vergata', Via del Politecnico 1, 00133 Rome, Italy

SUMMARY

Transmission Control Protocol (TCP) performance over Digital Video Broadcasting—Return Channel via Satellite (DVB-RCS) standard is greatly affected by the total delay, which is mainly due to two components, propagation delay and access delay. Both are significant because they are dependent on the long propagation path of the satellite link. The former is intrinsic and due to radio wave propagation over the satellite channel for both TCP packets and acknowledgements. It is regulated by the control loop that governs TCP. The latter is due to the control loop that governs the demand assignment multiple access (DAMA) signalling exchange between satellite terminals and the network control center, necessary to manage return link resources. DAMA is adopted in DVB-RCS standard to achieve flexible and efficient use of the shared resources. Therefore, performance of TCP over DVB-RCS may degrade due to the exploitation of two nested control loops also depending on both the selected DAMA algorithm and the traffic profile.

This paper analyses the impact of basic DAMA implementation on TCP-based applications over a DVB-RCS link for a large set of study cases. To provide a detailed overview of TCP performance in DVB-RCS environment, the analysis includes both theoretical approach and simulation campaign. Copyright © 2009 John Wiley & Sons, Ltd.

Received 12 March 2008; Revised 14 November 2008; Accepted 22 December 2008

KEY WORDS: DAMA; DVB-RCS; NS-2; TCP

1. INTRODUCTION

Digital Video Broadcasting—Return Channel via Satellite (DVB-RCS) is the standard [1–3] conceived to support bidirectional broadband communication via satellite. DVB-RCS systems are based on a star topology where many satellite terminals (STs) access telecommunication networks and are connected through a transparent geostationary (GEO) satellite to a central Hub. DVB

*Correspondence to: Cesare Roseti, Electronics Engineering Department, University of Rome 'Tor Vergata', Via del Politecnico 1, 00133 Rome, Italy.
†E-mail: roseti@ing.uniroma2.it

Contract/grant sponsor: European IST-FP6 project

over Satellite (DVB-S) standard [1], adopted in the forward link, regulates physical layer and multiplexing of a broadcast channel, while the return channel is shared among a group of users using multifrequency time division multiple access (MF-TDMA). Resources in the return channel are managed by a network control center (NCC) adopting demand assignment multiple access (DAMA) mechanisms [2]. In general, DAMA introduces an extra delay to data delivery, defined as 'access delay', because it is necessary to perform explicit requests for bandwidth. To exploit DAMA multiple access and obtain capacity the exchange of both 'capacity requests' (from STs to NCC) and 'notifications of capacity allocation' (from NCC to STs) messages is necessary.

DVB-RCS links present different levels of asymmetry [4]:

- Bandwidth asymmetry—Several Mbit/s in the forward link and up to 2 Mbit/s in the return link, both shared among all users.
- Delay asymmetry—A time-varying access delay, just due to DAMA technique implementation, is added to the propagation and processing delays in the return link only.
- Loss asymmetry—Different baseband processing (i.e. coding and modulation) in the forward and return link can lead to different loss rates perceived at the upper layers.

Such a communication environment is harsh for the Transmission Control Protocol (TCP) currently used for most of the Internet applications: web browsing, e-mail and File Transfer Protocol (FTP). In fact, in its original design [5,6], TCP considers both round-trip time (RTT) measurements and loss detection as a meter for an estimation of the network congestion, and on this basis it adapts the transmission rate. This process is usually performed without identifying which component of the overall RTT is due to congestion and the causes of the experienced losses. In addition, congestion is always imputed to the link carrying TCP data and not to the link carrying acknowledgement (ACK) packets. Definitively, in addition to the intrinsic high propagation delay characterizing GEO links, DAMA dynamics exploitation also makes TCP behaviour critical and thus strictly depends on both DVB-RCS framing and DAMA configuration.

This paper is focused on the effects of the joint utilization of TCP and DAMA on data transfer over DVB-RCS systems. TCP is clocked by the reception of ACK for received TCP packets, while DAMA is related to the exchange of explicit capacity request–allocation messages. Such control loops are nested and based on the same timescale (RTT). The time needed to fully exploit the two control loops depends on the propagation delay and determines the total perceived RTT. In addition, the performance of the specific TCP-based application is evaluated in order to identify the access scheme most suitable to the application needs.

To support this complex analysis, DVB-RCS system has been introduced in the network simulator (NS-2) platform [7]. It aims to model the most critical aspects allowing flexible tests to easily gather statistics at different protocol layers (i.e. medium access control (MAC), transport or application layer).

The rest of this paper is organized as follows: Section 2 describes the reference scenario providing details of the involved protocols and technologies; Section 3 presents the analysis of the TCP dynamics over DVB-RCS links and provides an analytical model to synthesize the interaction between TCP and DAMA control loops; Section 4 summarizes the main characteristics of the simulator; Section 5 summarizes the evaluation of TCP performance under various combinations of DAMA configuration, traffic direction, traffic load, physical constraints and application characteristics; finally, Section 6 provides conclusions.

2. REFERENCE SCENARIO

The reference scenario is represented in Figure 1. It envisages a star-based architecture with multiple RCS STs connected to a gateway/earth station via a transparent GEO satellite.

Each ST usually comprises an outdoor unit, antenna and transceiver and an indoor unit, performing baseband processing. Each ST can offer connectivity to one or more end-systems.

Users are supposed to utilize common IP services and more specifically TCP-based applications (i.e. WWW, FTP, e-mail, etc.). Furthermore, an interoperable performance-enhancing proxy (I-PEP) [8] agent, promoted by the SatLabs group [9], is installed both at ST and Hub side.

2.1. DVB-S(S2)

DVB-S standard [1] is adopted for data broadcasting in the satellite forward link. The baseband processing adopts moving picture expert group-2 (MPEG-2) as source coding of different input types (audio, video and data). The basic component of MPEG-2 system is known as elementary stream (ES). A programme (i.e. TV programme) is a combination of ESs generated by independent video, audio and data encoders. Each ES is an input for an MPEG-2 processor, which assembles data into a stream of packetized elementary stream (PES). A number of PES are finally multiplexed into the MPEG-2 transport stream (TS), which is composed of packets with a fixed length of 188 bytes with a packet identifier, which allows the receiver to reassemble the original PES packets.

The satellite channel adaptation concerns the types of modulation and coding schemes to be adopted to meet the target quality of the signal (bit error rate of around 10^{-11}).

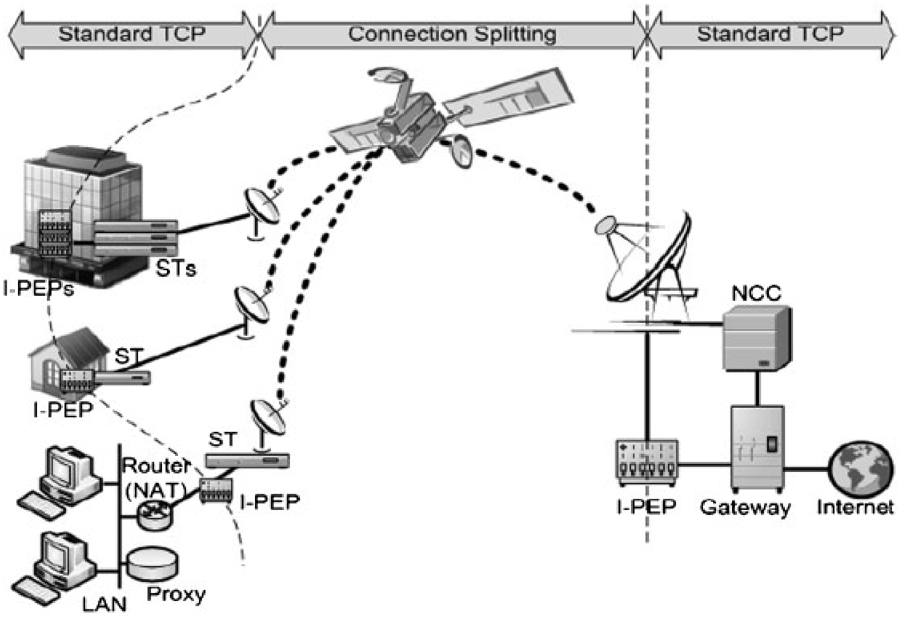

Figure 1. DVB-RCS reference scenario.

The processes involved in this adaptation are transport multiplex adaptation and randomization for energy dispersal, outer coding (i.e. Reed–Solomon), convolutional interleaving, inner coding (i.e. punctured convolutional code), baseband shaping and modulation.

Recently, the DVB-S standard was updated in the DVB-S2 version [10,11]. The new standard introduced adaptive modulation and coding (ACM), modifying the satellite channel adaptation block only. ACM requires the user terminals to notify the propagation channel conditions to adapt modulation and coding to comply with Bit Error Rate (BER) requirements.

2.2. DAMA over DVB-RCS

DVB-RCS allows bidirectional communications adopting DVB-S in the forward link-up and allowing up to 2 Mbit/s in the return link utilizing either MPEG-TS (as for the forward link) or ATM [12] on top of the physical layer. STs share the bandwidth on an MF-TDMA discipline [2,3] through a DAMA scheme. NCC assigns time slots as a response to explicit requests (bandwidth on demand) coming from STs issuing terminal burst time plan in the forward link every superframe.

DVB-RCS standard allows fixed or dynamic time slot allocations according to five different schemes:

1. *Continuous rate assignment* (*CRA*) is a fixed and static allocation of resources agreed between the ST and the NCC in the initial set-up phase;
2. *Rate-based dynamic capacity* (*RBDC*) allocates capacity dynamically requested on the basis of rate measurements performed by the ST; each request is absolute and overrides all previous requests from the same ST;
3. *Volume-based dynamic capacity* (*VBDC*) allocates capacity dynamically requested on the basis of data volume measurements performed by the ST; these requests are cumulative (i.e. each request shall add to all previous ones from the same ST), indicating a total number of traffic slots that are needed to transmit all data currently present in the MAC queue;
4. *Absolute volume-based dynamic capacity* (*AVBDC*) is similar to VBDC, but requests are absolute and AVBDC is used for loss recovery in the VBDC mode;
5. *Free capacity assignment* (*FCA*) allocates capacity to STs from 'spare' capacity that would be otherwise unused; such a resource assignment is automatic, not involving any request from the ST.

2.3. TCP over satellite

TCP was originally designed to work efficiently in wired and congested networks [6,7] but suffers from a certain number of factors when it runs over satellite links [13–18]. Specifically, high latency, large bandwidth-delay product, link asymmetry and channel errors can negatively impact TCP performance.

In the literature a large number of solutions have been proposed to improve TCP performance over satellite links [19]. A possible classification of proposed solutions is the following:

- *Non-TCP enhancements*—Provision of enhanced lower layers; in other words, TCP can benefit passively from mechanisms running at different protocol layers [20];

- TCP standard enhancements—Extensions compliant to standard TCP protocol specification [21];
- TCP variants—Modifications of the standard flow control, congestion control and error recovery schemes [22];
- Performance-enhancing proxies—Modifications of the TCP end-to-end semantic (including architectural modifications) [23].

Transport protocol over DVB-RCS is defined into I-PEP specification [8], which includes the use of several TCP enhancements as either mandatory (i.e. new selective negative acknowledgements (SNACK) option [8]) or optional (i.e. timestamps [24]). In addition to the standard TCP congestion control algorithm, TCP Vegas [25] is also allowed as the TCP variant.

2.4. I-PEP

To address the problem of degraded TCP performance, I-PEP specification [8] defines a protocol stack to be implemented at the edges of a DVB-RCS network with the twofold objective to optimize FTP and Hyper Text Transfer Protocol (HTTP) performance and guarantee interoperability among different vendor-specific ST implementations. TCP-based Space Communications Protocol Standard-Transport Protocol (SCPS-TP) [26] is selected as the transport protocol to be used between I-PEPs, with a well-defined set of options and a default congestion control algorithm (TCP Vegas [25]). A broader set of nonmandatory options is also available including the possibility to use a custom congestion control algorithm (a negotiation phase for optional capabilities is foreseen).

2.5. SCPS-TP reference profile for I-PEP

SCPS-TP envisages a set of capabilities designed for environments different from that of the I-PEP, namely deep space and tactical environments. For this purpose, SCPS-TP capabilities have been classified according to their applicability to I-PEP framework as applicable, modified/extended, optional and not applicable. SCPS-TP profile chosen as a reference in this paper relies only on the first two classes. An exhaustive description of all the considered capabilities is provided in [8], while hereafter two particular capabilities, fundamental in the interpretation of this paper's results, are briefly presented.

SNACK: it is an applicable option, which allows receiver to inform the sender of specific segments not received for the purpose of their immediate retransmission. This approach follows the idea of TCP's Negative Acknowledgement (NAK) defined in the RFC 1106 [27].

Congestion control and corruption: SCPS-TP implementations give the possibility of not using congestion control (i.e. performing a rate control) at all or otherwise of using either a standard TCP or TCP Vegas. The former is mainly based on Van Jacobson's Slow Start (SS) and Congestion Avoidance (CA) algorithm and exponential back-off of the retransmission timer for successive retransmissions [5,6,28,29]. In this paper, such a congestion control scheme is referred to as either the standard TCP or TCP Reno (although fast retransmission–fast recovery scheme is not mentioned by SCPS-TP). TCP Vegas introduces two changes to the above-presented TCP: a modified SS algorithm and a Modified CA algorithm [25]. The modified SS algorithm aims at avoiding potential congestion during SS. To this purpose, the exponential congestion window (cwnd) growth is performed only once every two RTT. In between, a comparison between the actual and the expected rate is computed. If the actual rate is below the expected rate, TCP

Vegas suddenly switches in CA. In addition, in the CA, actual rate is continuously compared with the expected rate. If the difference is higher than a given threshold, cwnd is decreased. The goal is to prevent congestion by adjusting cwnd accordingly to the RTT variations.

3. TCP OVER DAMA IN DVB-RCS

In a DVB-RCS environment, the use of DAMA mechanisms implies that TCP can experience an actual RTT well above the two-way propagation delay and the available capacity may vary strongly and abruptly. Of course, actual DAMA implementations might support algorithms, more or less compliant with DVB-RCS specifications [2], aiming to minimize, or at least keeping under a given threshold, both the access delay and jitter. Nevertheless, DVB-RCS system manufacturers usually keep such algorithms confidential, and also for this reason they are not dealt with in this paper. Packet loss is not meaningful, since DVB-RCS applies a strong forward error correction at the lower layers, providing to upper layers a quasi-error-free channel [1,2].

When TCP operates over a DAMA-based satellite link, two nested control loops govern the data transfer process:

- TCP flow control loop;.
- DAMA bandwidth allocation control loop.

Dynamics of the overall DVB-RCS nested control loops are shown in Figure 2. Data coming from a TCP socket feed MAC buffer activating DAMA control loop. Specifically, STs send a request message to the NCC, which sends back the allocation message. At the end of this loop, if

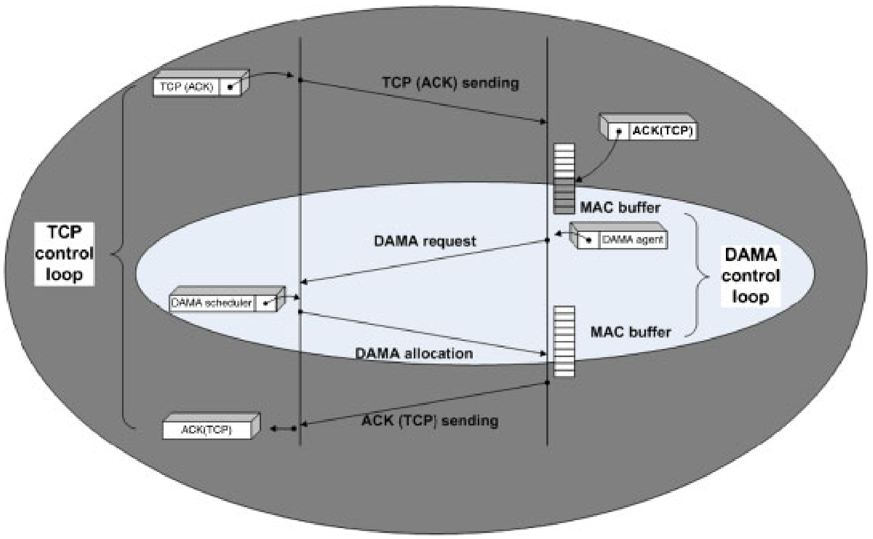

Figure 2. DVB-RCS nested loops.

at least a part of the requested capacity is allocated, the ST is allowed to send data on the return channel.

After the reception of ACKs, TCP updates the sliding window [5] and then sends further packets. Definitively, TCP control loop exploitation is constrained by the DAMA control loop execution. In fact, resources for TCP packets/ACKs in the return link are allocated after DAMA procedure finalization. Thus, TCP and DAMA loops are nested and the duration of both depends on the propagation delay as a multiple of the RTT.

3.1. DAMA performance

To evaluate DAMA performance two metrics are identified:

- *Access delay*, which impacts TCP performance [30];
- *Bandwidth utilization*, which determines the efficiency of network resources utilization.

In terms of the two metrics, the different allowed allocation schemes show different performances from a qualitative point of view. CRA adds no 'access delay' so that the overall delay corresponds to the propagation delay plus the processing delay (around 500 ms for a GEO satellite). Bandwidth is assigned regardless of the actual utilization thus showing low efficiency. VBDC and AVBDC, conceived for best effort service, imply a much longer RTT, typically around 1.5 s, due to the request/allocation cycle. Capacity will be requested only for data already present in the MAC buffer so that the bandwidth utilization is 100% (maximum efficiency). RBDC, intended for constant rate applications, implies a perceived RTT similar to CRA, except some extra delay for the first request and for variations of requested rate. The allocated bandwidth rarely corresponds to the needed bandwidth so that efficiency is not so high, but higher than that for CRA. FCA assigns all the bandwidth left available after slots allocation algorithm exploitation to all STs in an undefined way. Evaluation of delay and efficiency requires knowledge of traffic evolution. Analysis in this paper concerns CRA, VBDC and RBDC.

DVB-RCS standard [2,3] does not provide implementation details for DAMA schemes, but while VBDC is rather straightforward and there are not many degrees of freedom, RBDC is much more open to different algorithms and suitable traffic models must be assumed.

3.2. Calculation of data transfer time for TCP transfer over DAMA

Data transfer time for TCP traffic can be evaluated as in [5,6] assuming instantaneous or negligible resource management process duration. The proposed methodology aims to characterize the data transfer process in terms of the time necessary to perform a TCP-based file transfer.

At the beginning, TCP performs a 'three-way handshake', in which the end-systems exchange the initial sequence number sending SYN segments. The overall time needed for this procedure is equal to one RTT:

$$T_{\mathrm{HS}} = \mathrm{RTT} \tag{1}$$

Data transfer is exploited as long as an application places data in its sending buffer and, assuming no packet loss, it can be divided into two phases [28,29]:

1. SS phase;
2. CA phase.

In SS phase, TCP sender increases its cwnd by one for each received ACK (with the assumption that delayed ACK mechanism is disabled [5]) implying that for every RTT the TCP sender will double its cwnd:

$$\text{cwnd}_{i+1} = 2 \cdot \text{cwnd}_i \tag{2}$$

More in general,

$$\text{cwnd}_i = \text{cwnd}_0 \cdot 2^i \tag{3}$$

where cwnd_0 is the initial cwnd value. Therefore, let d_{ss} be the number of packets expected to be transmitted in the SS phase, where ssthresh is the initial value of the SS threshold. The SS duration is

$$T_{\text{ss}} = \text{RTT} \cdot \left[\log_2\left(\min\left(\frac{d_{\text{T}}}{2}, \text{ssthresh}\right)\right) - \log_2(\text{cwnd}_0)\right] \tag{4}$$

where RTT represents the average round-trip delay and d_{T} is the total number of packets to be transmitted. If data transfer is not completed once ssthresh is reached, TCP switches to CA phase during which cwnd increases linearly over the RTT:

$$\text{cwnd}_{i+1} = \text{cwnd}_i + 1 \tag{5}$$

Then, the number of packets to send in the CA phase is

$$d_{\text{ca}} = d_{\text{T}} - d_{\text{ss}} \tag{6}$$

Considering Equation (5) d_{ca} can be expressed as a function of both the number of RTT needed to transmit it (N_{ca}) and ssthresh. Without loss of generality, it is assumed that file dimension is not so large to reach the maximum cwnd:

$$d_{\text{ca}} = N_{\text{ca}} \cdot \text{ssthresh} + \frac{N_{\text{ca}} \cdot (N_{\text{ca}} - 1)}{2} \tag{7}$$

From Equation (7) it is possible to obtain N_{ca}. Therefore, the time needed to transfer d_{ca} is

$$T_{\text{ca}} = \text{RTT} \cdot N_{\text{ca}} \tag{8}$$

Definitely, the time needed to transfer a file composed of d_{T} TCP packets is the sum of contributions from Equations (1) and (4):

$$T_{\text{tot}} = \text{RTT} \cdot \left[1 + \log_2\left(\min\left(\frac{d_{\text{T}}}{2}, \text{ssthresh}\right)\right) - \log_2(\text{cwnd}_0) + N_{\text{ca}}\right] \tag{9}$$

RTT in Equation (9) represents the average round-trip delay perceived by TCP sender, which includes propagation delay and access delay.

To separately identify the contributions of each control loop, as in the case under study, when resource allocation process duration is not negligible, Equation (9) needs to be upgraded. The goal is to isolate the contribution of DAMA access delay, which depends on the adopted DAMA scheme, on TCP performance to optimize performance as a consequence.

Since capacity assigned to an ST can be changed on a superframe basis, the access delay (or 'system response time') can be expressed as a multiple L of the superframe duration with L as the number of superframes necessary to allocate all the requested capacity. Then, access delay can be given as $L \cdot D_{\mathrm{SF}}$, where D_{SF} is the superframe duration.

With CRA the access delay is not perceived since allocation is performed in a static way at the login phase. The only variable contribution $\Delta_{\mathrm{SF}} \leqslant D_{\mathrm{SF}}$, present in all the DAMA schemes, is due to the difference between the arrival time of packets in the MAC buffer and the instant in which the terminal is allowed to start transmission.

In case of VBDC, requests are computed one-off on a given volume of data. As a consequence, the experienced RTT includes the whole access delay ($L \cdot D_{\mathrm{SF}}$) as long as no congestion occurs.

RBDC computes requests on the basis of the transmission rate and remains valid for some superframes. If the rate is constant and capacity has been already allocated, the experienced RTT is equal to that experienced with CRA and represents the best case. Instead, bursty traffic with a duty cycle larger than RBDC allocation cycle requires for every burst a request making RBDC behaviour similar to VBDC, consequently increasing the experienced RTT up to the VBDC worst case. Intermediate traffic conditions lead to intermediate experienced RTT.

In summary, depending on the used access scheme, RTT in Equation (9) can be expressed as

$$\mathrm{RTT} = \mathrm{RTT}_0 + \mathrm{RTT}_{\mathrm{DAMA}} \tag{10}$$

where RTT_0 represents the contribution of the propagation delay and constant processing delay (encapsulation, interleaving, etc.) and $\mathrm{RTT}_{\mathrm{DAMA}}$ is given by

$$\mathrm{RTT}_{\mathrm{DAMA}} = \begin{cases} \Delta_{\mathrm{SF}}, & \mathrm{CRA} \\ L \cdot D_{\mathrm{SF}} + \Delta_{\mathrm{SF}}, & \mathrm{VBDC} \\ \begin{cases} \Delta_{\mathrm{SF}}, & \text{const rate} \\ L \cdot D_{\mathrm{SF}} + \Delta_{\mathrm{SF}}, & \text{burst tx} \end{cases} & \mathrm{RBDC} \end{cases} \tag{11}$$

DVB-RCS supports CRA and RBDC schemes either as stand-alone or combined with VBDC. In mixed cases, the overall RTT is a weighed combination of values in Equation (11).

4. SIMULATION PLATFORM

4.1. Simulator design

A DVB-RCS simulator has been set up on the NS-2 platform [7]. Specifically, DAMA scheme functionalities (see Section 2.2) have been added to the satellite baseline module.

Most of the DAMA functions have been included into a new 'DamaMac' class (Mac/Sat/Dama), as shown in Figure 3, which both manages allocation signalling among STs and NCC (dotted arrows) and performs DAMA algorithms concerning the request computation (ST side) and the allocation decision (NCC side). In the simulated DAMA, a simple proportional allocator has been used, where capacity is assigned on a superframe basis keeping the original ratio among all requests but with a cap on the maximum number of slots per superframe.

The 'DamaTerminalProfile' class manages the DAMA profile of each ST through a simplified login procedure. Such a class is also in charge to monitor the MAC queue status, periodically delivering allocation signalling and adjusting the ST sending rate accordingly. On top of DAMA, TCP agents are configured to operate as either connection source or sink (Figure 3).

All the packets coming from STs are delivered to the DAMA agent that, on the basis of the assigned resources, schedules their transmission towards the GW.

Finally, TCL scripts are used to set up the simulation session:

- Network configuration (i.e. number of STs, node coordinates);
- Definition of the DAMA profile parameters for each ST (i.e. CRA granted slots, maximum number of VBDC requests);
- Definition of the traffic scheduling;
- Configuration of probes to log the state variables in order to generate post-simulation plots.

Figure 4 shows the sequence diagram of the resource request–allocation messages exchange managed by DAMA module. The time elapsing between sending a capacity request and the corresponding 'capacity response' introduced in the Burst Time Plan (BTP) constitutes the access delay and mainly includes the propagation round-trip delay and a processing delay at the NCC side. This process requires the management of several pending requests at all times through a first in first out queue. Definitively, access to the MAC QUEUE is performed on a superframe basis as well as the change of the assigned resources so that the 'access delay' can be expressed in terms of number of superframes.

4.2. DAMA simulator parameters

All the parameters concerning the frame structure have been chosen in the range of the actual deployed DVB-RCS systems [9]. The capacity of the return channel is assumed to be $C_{\mathrm{RL}} = 2048$ kbit/s, divided into 32 channels of 64 kbit/s, so that $N_s = 32$ slots or cells are present in a TDMA frame. Thus, ST with only one slot allocated per superframe utilizes a net capacity of 64 kbit/s. A frame is assumed to last $T_f = 23.5$ ms (typical value for real TDMA structure) and a superframe is assumed to be composed of four frames, for a total superframe duration of $T_{\mathrm{SF}} = 4 \cdot T_f = 94$ ms. As a consequence each slot carries a total of

$$\frac{C_{\mathrm{RL}} \cdot T_f}{8 \cdot N_s} = 188 \text{ bytes} \tag{12}$$

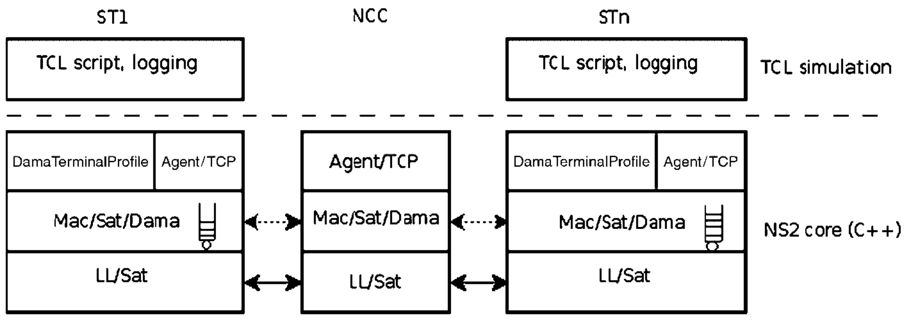

Figure 3. Simulator architecture.

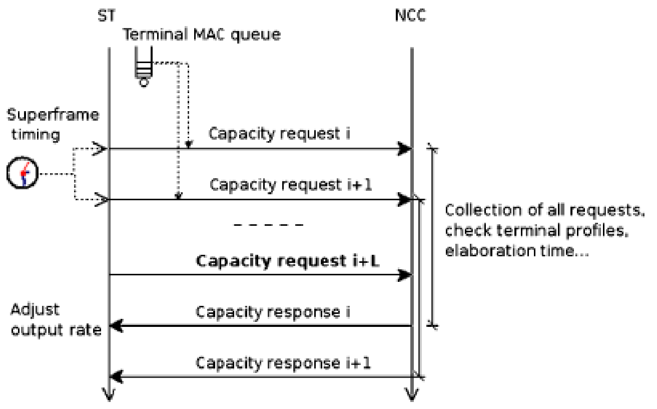

Figure 4. Message time sequence.

corresponding to an MPEGZ-TS cell. However, in the simulation model, encapsulation protocol is not considered: IP packets are sent on the air interface without encapsulating them neither in ATM nor in MPEG-2 cells and the actual segmentation is only performed for resource allocation matters.

The ST profile includes the number of CRA slots statically allocated, and the maximum number of slots that it is entitled to request according to RBDC and VBDC method.

As far as RBDC discipline is concerned, the formula presented in [28] is used to calculate the number of slots to be requested; the principle is to make the output rate equal to the estimated input rate of the MAC queue. The *n*th request is computed as

$$\mathrm{RBDCreq}_i[n] = (1-\gamma)\cdot \mathrm{RBDCreq}_i[n-1] + \gamma \cdot Q_i \tag{13}$$

where Q_i is the instantaneous value expressed in slots of the queue occupancy of the *i*th ST at the moment of the request and γ is the weighting factor between 0 and 1. As default a value $\gamma = 0.4$ has been assumed. With reference to Equation (13), if Q_i is equal to the previous RBDC request, it means that the rate is constant and then the new request is kept equal to the previous one. In case $Q_i > \mathrm{RBDCreq}[n-1]$, the queue input rate is increased and then the new RBDC request is increased accordingly. Obliviously, the case where $Q_i < \mathrm{RBDCreq}[n-1]$ will lead to a reduction of the new RBDC request.

For VBDC requests instead, each byte in the queue must be explicitly notified to NCC to be allowed for transmission. For this reason the request algorithm must take into account the previous pending requests not yet served. The following formula taken from [31] will be used to express the *n*th request:

$$\mathrm{VBDCreq}_i[n] = \mathrm{MAX}\left\{\left(Q_i(n) - \sum_{k=n-L}^{n} \mathrm{VBDCreq}_i[k]\right), 0\right\} \tag{14}$$

where L represents the number of superframes elapsing between a capacity request and the corresponding response or, in other words, the number of pending requests, as shown in Figure 4. To match the current validated system performance (see [28]) in the simulations, L has been set to 11.

The sum of capacity requests issued by all STs into a given superframe can exceed the total system capacity available. In this case, competing STs should receive in response a number of slots less than the requested ones, according to a proportional algorithm defined in the NCC [32]. Management of the unassigned, but required, slots is in charge of the DAMA allocator in the NCC, which stores the pending requests in order to perform the assignment in the next superframe as soon as the capacity is available.

4.3. Simulated applications

Internet traffic is mostly composed of TCP flows generated by different applications: HTTP, e-mail and FTP [33].

E-mail and FTP service have a similar traffic profile, consisting of a preliminary interactive session followed by a batch data transfer (in case of e-mail significant only if there are attachments). In general, the Simple Mail Transfer Protocol envisages a command exchange quite limited with respect to that performed by FTP. Nevertheless, for the scope of this paper, only the batch data transfer is taken into account for performance evaluation. On the contrary, all messages exchanged at both application and transport layer to set up and close connections have been neglected, since they generally represent a trivial percentage of the overall data transfer. Then, both mail and FTP transfers are herein modelled by the batch data size.

The third application class is web browsing via HTTP/1.1 protocol [33,34], which consists mainly of a TCP connection with multiple objects serially transmitted. Specifically, HTTP traffic has been modelled from real application traffic in order to generate a realistic pattern for NS-2. The reference HTTP server is Apache2.0 and the reference HTTP client is Firefox2.0. Both have been used with default options adopting the HTTP/1.1 standard [34]. The server side supports the KeepAlive option so that multiple objects can be delivered on the same TCP connection within a given timeout of few seconds. For the client side options, KeepAlive is accepted with a maximum simultaneous connection number set to two. With these settings, some real web page download has been performed over a Linux-emulated satellite link introducing a propagation delay of 250 ms over the Ethernet network to access the Internet. The request/response series have been monitored and the result is shown in Figure 5, where the sending time of outgoing

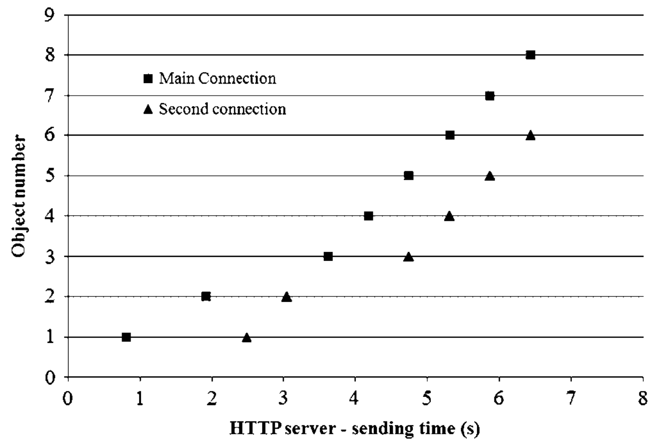

Figure 5. Firefox2.0/Apache2.0 web objects timing over satellite–server side.

objects at the server side is represented. In the figure, time origin corresponds to the initial packet sent by the client, namely the SYN packet of the first TCP connection (squares). When the first object is entirely received from the client (usually the index.html page), it decodes its content and prepares a second connection (triangles), which, together with the already established one, will deliver all the remaining objects in parallel. The object sent with the second connection leaves at around 2.7 s, including the three-way handshaking and the delivery of the first request by the client on the second connection. Such a behaviour has been fully implemented in the NS-2 simulation, with several random variables used to represent the number and size of objects, which are always split along two connections. Object sending times are achieved on the basis of the observed traffic scheduling in Figure 5.

The main characteristics of the traffic classes generated in NS-2 simulations designed according to the three applications are summarized in Table I. FTP and e-mail can be grouped considering the object size from 1 to 12 Mbyte (e.g. e-mail with big attachments or FTP file transfer). In case of HTTP STs are only receivers.

4.4. Simulation set-up, testplan and metrics

Simulations concern TCP-based transfers among I-PEP agents installed in the STs and in the GW/NCC. This means that both TCP sources and receivers can be elsewhere and a split connection between I-PEPs concerns only the DVB-RCS satellite link. Only TCP sub-connections over satellite are considered because:

1. Satellite link is largely the bottleneck of the end-to-end link, both from the bandwidth and performance point of view;
2. The effects of TCP–DAMA interaction from other externals factors can be isolated.

The bandwidth for the forward link is 10 Mbit/s, while it is 2.048 Mbit/s for the return link, shared among the STs according to the pre-configured DAMA scheme. Each terminal has granted at least 1 slot, out of the 32 available slots, assigned with CRA for a guaranteed minimum bandwidth of 64 kbit/s. Further slots can be assigned again as CRA (NCC manages such a static assignment in a set-up stage) to satisfy RBDC and VBDC requests. DAMA algorithms are those presented in Section 2.2. As far as RBDC is concerned, γ parameter is selected to achieve the best trade-off between TCP performance and bandwidth utilization.

Owing to the huge number of free parameters, a subset of possible configurations has been identified with the aim to address the most significant study cases. In particular, Table II lists five classes of selected tests.

T_I concerns analysis of TCP steady-state behaviour based on the measurements of RTT and throughput by running a long connection. In addition, the efficiency in using the assigned resource is evaluated for the different phases of the TCP lifecycle. In all the tests presented in this paper, the receiver window (rwnd) is assumed to be large enough to not limit the sending window (rwnd > cwnd is always verified).

In T_II, multiple TCP flows share the return link capacity using the same algorithm. TCP fairness measures the capability of the algorithm to share the bandwidth when multiple flows are active on the same link.

In the frame of T_III, TCP friendliness measures the capability of a flow using an algorithm to share the bandwidth with other flows adopting different algorithms [35].

Table I. Applications characterization.

	Number of messages for set-up	Objects to transfer	Object size	Parallel connections
HTTP	2	1–30	1–200 kbyte	Yes
FTP/e-mail	4–8	1	1–12 Mbyte	No

Table II. Test classes.

ID	Test name
T_I	TCP steady-state behaviour
T_II	TCP fairness
T_III	TCP friendliness
T_IV	FTP/e-mail transfer time
T_V	HTTP transfers

Finally, T_IV and T_V address the performance of real TCP-based applications over DVB-RCS: FTP/e-mail and HTTP, respectively.

5. PERFORMANCE ANALYSIS

5.1. T_I tests

The set of Figures 6–10 are concerned with the analysis of performance of a single TCP connection implementing CRA, RBDC and VBDC. Similar tests related to general DVB-RCS environments can be found in the literature [30,31,36]. Herein results concern perceived RTT by TCP for different DAMA schemes over 100 s interval (no additional information is derived for longer simulations), while throughput and efficiency are analysed over 400 s. Each figure shows six curves:

1. TCP Reno over CRA (continuous line);
2. TCP Reno over RBDC (thick dotted line);
3. TCP Reno over VBDC (thin dotted line);
4. TCP Vegas over CRA (cross-marked line);
5. TCP Vegas over RBDC (triangle-marked line);
6. TCP Vegas over VBDC (rhombus-marked line).

In case TCP data flow through the forward link, the return link is used for ACKs and hence a few slots are needed. Since one slot is statically assigned in CRA in each of the considered cases, at the start RTT does not include any contribution from DAMA exploitation. With reference to Figure 6, RTT is about 500 ms for the first 20 s (due to propagation), while, once ST has received a number of ACKs such that ST needs additional slots, DAMA contribution is introduced in the overall delay. RTT presents substantial variations for TCP Reno over both VBDC and RBDC access schemes. In the former case, RTT gradually increases. The only slot statically assigned in each superframe is not sufficient to match the ACK rate. Extra capacity is requested with VBDC requests: the corresponding possible assignment is delayed by VBDC access delay (refer to Equation (11)). The higher the ACK rate exceeding the CRA assignment,

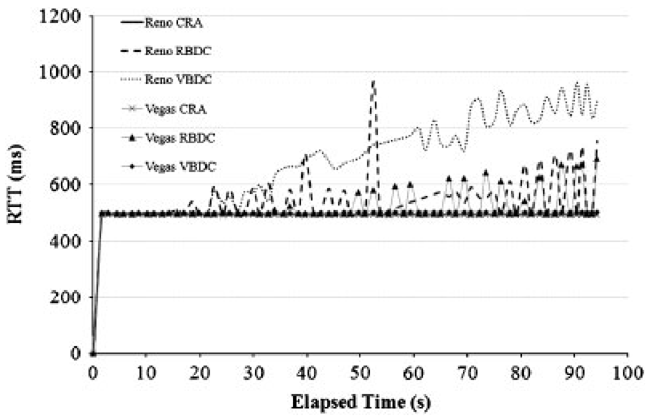

Figure 6. RTT vs time, TCP transfer over forward link.

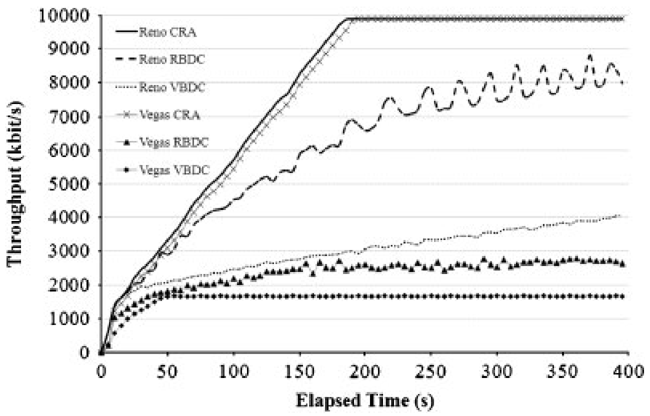

Figure 7. Throughput vs time, TCP transfer forward link.

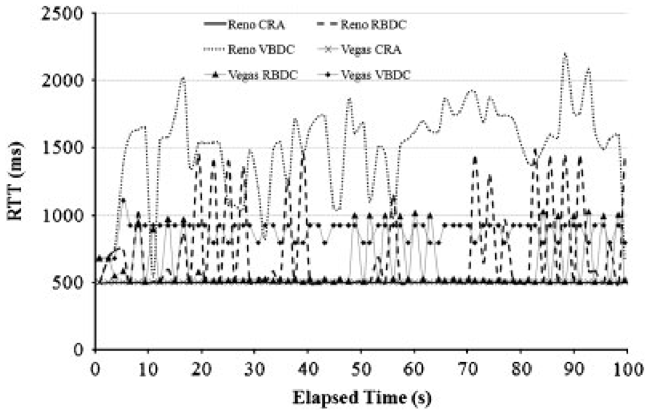

Figure 8. RTT vs time, TCP transfer over return link.

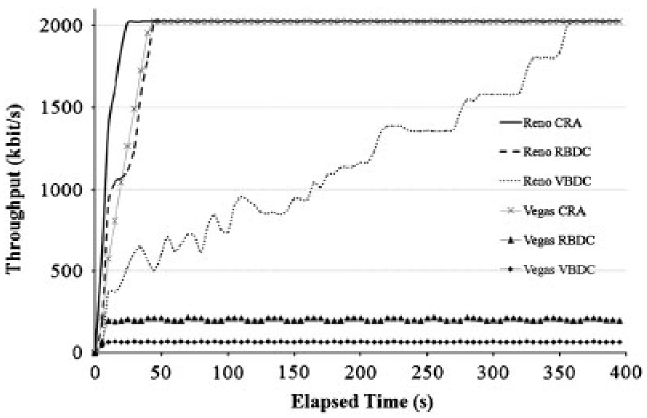

Figure 9. Throughput vs time, TCP transfer return link.

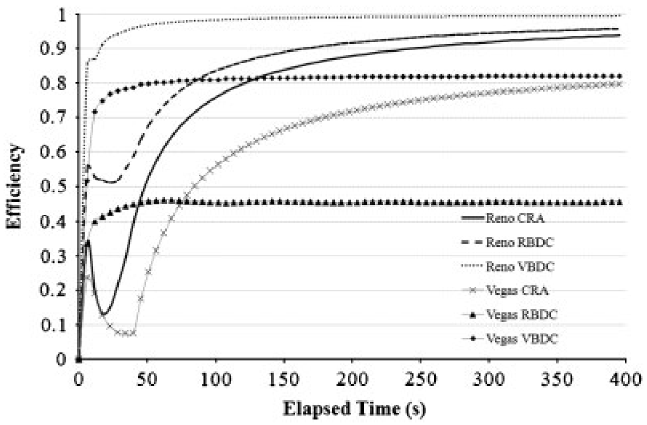

Figure 10. Efficiency in the return link.

the longer the time that ACKs must wait in the ST buffer before actual transmission over the RCS link. According to the physical delay and DVB-RCS frame structure, the maximum VBDC access delay is about 1–1.5 s. In the RBDC case, the RTT trend is oscillatory due to the bursty nature of the TCP flow. Specifically, ACK generation follows the arrival of TCP packets, whose transmission is clocked in turn by the ACK reception. In the simulated scenario, ACKs are received by the TCP sender compressed over time. The rationale is that ACKs are queued in the ST MAC buffer waiting for resource allocation. Such a waiting time is proportional to the 'access delay'. In the following, ACKs are sent at once when resources are allocated (capacity requests are completely satisfied since congestion does not occur). The resulting sender side behaviour is shown in Figure 11, where crosses indicate packet transmission and circles represent ACK receptions. From a MAC point of view, queue occupancy periodically returns to zero. This impairs RBDC control loop, which is slow in establishing a stable estimate of the transmission rate leading to oscillations in the perceived RTT; RBDC needs some time to match resource assignment rate to the actual source activity; in fact, when no new data feed ST queue

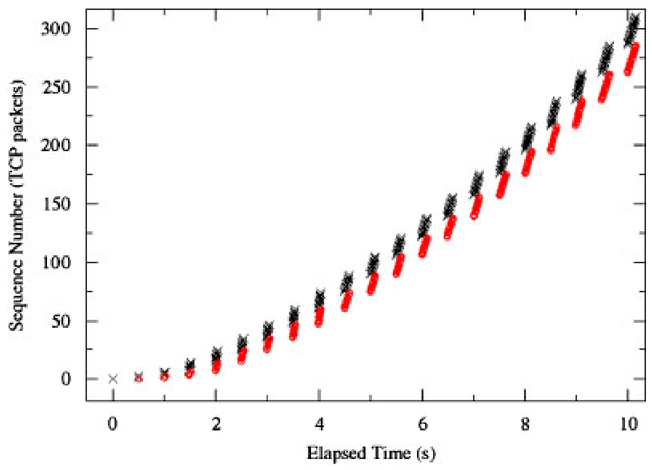

Figure 11. TCP sender behaviour: transmission over satellite forward link with return link accessed in RBDC; crosses represent transmitted packets, circles represent received ACKs.

for several superframes, the RBDC gradually reduces the resources assigned since Q_i in Equation (13) tends to zero and as a consequence also RBDCreq$_i$[n]; on the other hand, when new data feed the queue ($Q_i > 0$), RBDCreq$_i$[n] becomes not null but, as in VBDC, assignment will be exploited upon the reception of the assignment notification.

As far as TCP Vegas is concerned, RTT remains unchanged with VBDC (500 ms) and is subject to slower and lower oscillations with RBDC. The rationale relies on how TCP Vegas interprets RTT variations. In detail, as long as ACKs can be accommodated in the allotted CRA slot, the measured RTT is close to the minimum experienced RTT and TCP Vegas sender attempts to increase its cwnd. However, while further slots need to transmit a higher number of ACKs, the access delay due to either RBDC or VBDC requests inflates the overall RTT and it is interpreted by TCP Vegas as a congestion indication. As a consequence, TCP Vegas sets its cwnd to the maximum value not implying RTT increase. This behaviour is confirmed in Figure 7, where looking at the throughput achieved in the steady state it is shown that only 15% of the available bandwidth in the VBDC case and the 28% in the RBDC case are actually utilized. After 180 s, only in the CRA mode both TCP Reno and TCP Vegas use the whole available bandwidth. Finally, the throughput of TCP Reno with RBDC and VBDC cases increases, although very slowly. In fact, after 400 s the throughput is 80% of the available bandwidth in the RBDC case and only 40% in VBDC. In general, TCP Reno throughput increase is proportional to the overall experienced RTT.

Figures 8 and 9 show, respectively, RTT and throughput in case TCP flow is sent over the return link. TCP Reno performance can be resumed as follows for the three considered access schemes.

- With CRA, the RTT is fixed to about 500 ms (sum of propagation and transmission delay—no DAMA contribution); this implies that the maximum throughput is reached after
  30 s (time needed to achieve the optimal cwnd running Van Jacobson's SS and CA algorithms [29]).

- With RBDC, RTT oscillates from 500 ms to about 1500 ms (see Figure 8), for the same reasons of the forward link, but the throughput is much less affected by the varying RTT (capacity saturation after 50 s). This is due to two main factors. The former is that, since TCP packets are almost 40 times larger than ACK packets, the computed RBDC requests are higher (largely exceeding the average transmission rate) and some slots are allocated for a higher number of superframes (refer to Equation (13)). The overestimation of the needed resources reduces the access delay and as a consequence allows a fast cwnd increase. In addition, return link capacity is one-fifth of the forward link capacity. Then the optimum cwnd is lower.
- With VBDC, RTT oscillates around 1500 ms but remains not strictly constant due to the slot assigned in CRA that 'breaks' the equilibrium coming from Equation (14); the overall larger RTT leads to a slower throughput increase and bandwidth saturation is achieved after 350 s.

TCP Vegas behaves like TCP Reno with CRA, although it is slightly slower in reaching the maximum allowed bandwidth. In particular, Figure 9 clearly shows that TCP Vegas with CRA is equivalent to TCP Reno with RBDC in terms of throughput. This relies on the fact that TCP Vegas implements a modified SS algorithm, which increases cwnd in a more careful way [25]: to be able to detect congestion during SS, TCP Vegas exponentially increases its cwnd one time every two RTT.

With both RBDC and VBDC, TCP Vegas shows the same behaviour observed in the forward link. It takes the minimum RTT of the whole connection as a reference for congestion detection algorithms, fixing its throughput to a lower value of about 190 kbit/s (three slots) with RBDC and 64 kbit/s (one slot) with VBDC. TCP Vegas, due to the described inefficiency, continuously asks for only three slots with RBDC (one with CRA and two with RBDC) with a resulting RTT of about 950 ms, while for VBDC only the CRA slot is used and then the delay remains at 500 ms.

Network efficiency (η_N) in the case of a single TCP flow has been evaluated only in the case of the TCP flow over the return link because for the other direction ACK flow requires a low number of slots. Efficiency has been defined as follows:

$$\eta_N = 1 - \frac{\text{assigned capacity} - \text{used capacity}}{\text{assigned capacity}} \tag{15}$$

Both assigned and used capacity are averaged over the time.

Results shown in Figure 10 highlight that:

- TCP Reno with CRA leads to a very low efficiency at the start (ranging from 15 to 30%), but efficiency grows with the throughput; of course, the rationale is that the assigned capacity is fixed and irrespective of TCP needs;
- TCP Reno with RBDC is not efficient at the start (50%), while it approaches VBDC performance for longer interval times; in fact, at the beginning the bursty nature of TCP is more evident and leads RBDC to periodically require more capacity than necessary; approaching the maximum throughput, TCP rate variations become lower (CA phase), packets are equally spaced in time and as a consequence RBDC requests are more regular and correspondent to the actual TCP rate;

- TCP Reno with VBDC presents the best performance; the time needed to reach the maximum is due to the effect of the CRA slot statically assigned;
- TCP Vegas with CRA is less efficient than TCP Reno under the same conditions, since it takes longer to reach the maximum throughput;
- TCP Vegas with RBDC has the lowest efficiency at the steady state (47%) due to the joint effect of the RBDC algorithm, which tends to periodically overestimate the needed capacity, and the TCP Vegas congestion control that does not allow the increase of the cwnd although there is still free capacity;
- TCP Vegas with VBDC has an efficiency of 80%; VBDC still allows a high efficiency, but TCP Vegas requires rarely extra slots (underestimation of the available capacity) and then the presence of a CRA slot contributes to decrease in the efficiency of 20%.

5.2. T_II tests

In the general case, when link capacity is shared among several TCP flows, two parameters can be monitored to evaluate the good utilization of the available resources:

1. The fairness index;
2. The channel utilization.

The former relies on Jain's fairness index J_{f} [37]:

$$J_{\mathrm{f}} = \frac{(\sum x_i)^2}{n \cdot \sum x_i^2} \tag{16}$$

where

$$x_i = \frac{T_i}{F_i} \tag{17}$$

and T_i is the measured throughput for the ith flow, F_i is the fair throughput for the ith flow and n is the number of TCP flows. The fair throughput for each TCP flow is simply defined as the overall bandwidth divided by the number of TCP flows.

The channel utilization (η_{U}) is defined as follows:

$$\eta_{\mathrm{U}} = \frac{\sum T_i}{B} \tag{18}$$

where B is the whole available bandwidth.

In the first set of simulations, five TCP flows are started over the return link spaced at 5 s. Figures 12–14 show results in the case of CRA, RBDC and VBDC access schemes. This very simple scheduling of the traffic is useful to highlight what happens when a new connection has to compete bandwidth with other pre-existing connections, which have higher cwnd values. In this sense, the analysis of the fairness index aims to evaluate how quickly TCP is able to equally share bandwidth among connections characterized by different cwnd once the full bandwidth is used. In general, two phases can be distinguished over the proposed simulations:

- Transient phase (first $\sim$40 s)—before the achievement of the steady state;
- Saturation phase (from $\sim$40 s on)—during steady state.

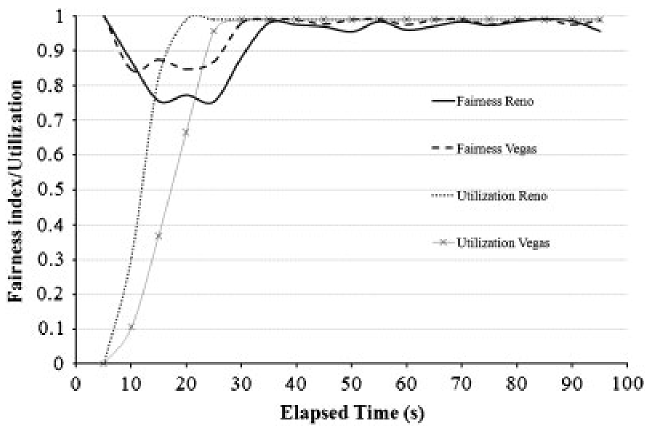

Figure 12. TCP fairness and utilization vs time, CRA.

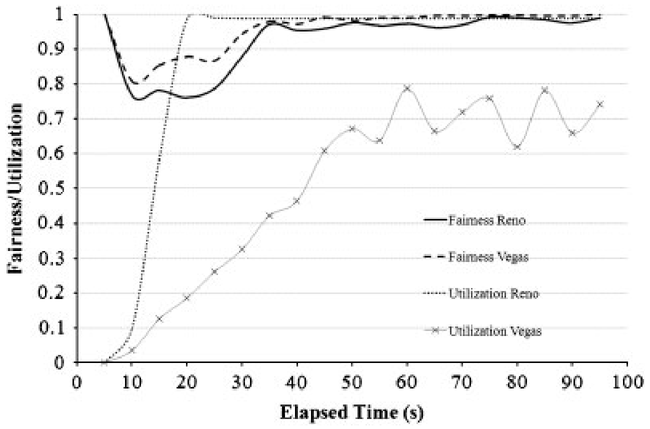

Figure 13. TCP fairness and utilization vs time, RBDC.

With CRA in the transient phase, TCP Vegas flows share the bandwidth more fairly than TCP Reno (fairness index 10% higher), but it achieves a better utilization being faster in reaching the maximum bandwidth. In the saturation phase, both TCP Reno and TCP Vegas achieve optimum performance (Figure 12). Definitively, TCP Vegas allows an optimum sharing of the available bandwidth after only a few seconds from the full bandwidth occupation. On the contrary, more than 15 s elapse between the full bandwidth occupation and optimum capacity sharing among TCP flows as far as TCP Reno is concerned.

With RBDC scheme (Figure 13), fairness index presents the same trend observed when running CRA. As far as the utilization is concerned, the overall capacity got by TCP Vegas flows is drastically decreased and utilization does not reach 70%. The rationale is that TCP Vegas sources misinterpret RTT variations due to RBDC variable access delay as a congestion and then reduce their transmission rate accordingly. As a result, the sum of transmission rate of TCP flows is lower than the available bandwidth. The slightly better fairness of TCP Vegas can

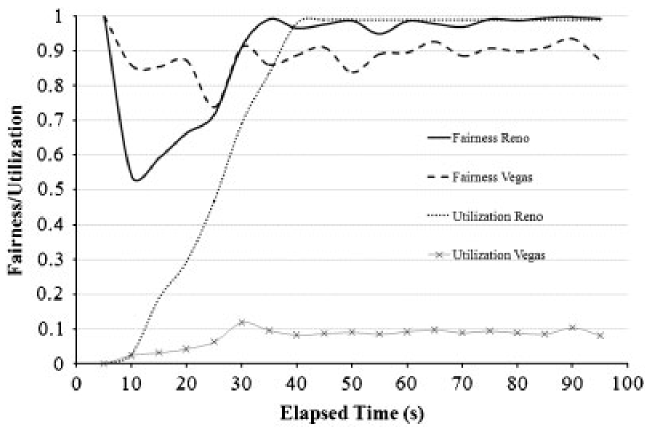

Figure 14. TCP fairness and utilization vs time, VBDC.

be then explained by the fact that sources quickly converge to the same cwnd value proportional to the experienced level of congestion, although mistaken.

When VBDC is adopted, TCP Vegas performance is completely compromised (Figure 14). Outcomes, observed with RBDC, are significantly emphasized with VBDC due to much higher RTT variations. In fact, fairness, although better than TCP Reno in the transient phase, levels off at the 0.9 value, while utilization is always 10%. In addition, TCP Reno performance is affected by the larger RTT and flows are quite unfair in the transient phase and the maximum utilization is achieved 20 s later with respect to the previous cases.

After the above-discussed analysis that gives a general picture on how multiple SCPS-TP flows compete for capacity of DVB-RCS return link, a further set of simulations envisages a more realistic traffic model, where 10 TCP sources behind ST start transmission randomly at irregular interval times following a uniform distribution. Specifically, time between starting two consecutive connections is uniformly distributed from a minimum value of 1 s to the maximum value of 10 s. Furthermore, also the duration of each connection is randomly achieved, considering a uniform distribution of values between 50 and 100 s. Two random generators have been then introduced in the simulation: for each selected 'seed' value, a set of random starting time and a set of transfer duration are obtained. Figure 15 shows the resulting traffic pattern in terms of number of active connections over the simulation time. Then, fairness index has been evaluated in such a traffic scenario in the case of both CRA and VBDC, which represent the boundary cases that have arisen from the previous set of tests. Moreover, in each of the two MAC configurations, both TCP Reno and TCP Vegas flows have been alternatively tested. In the case of CRA (Figure 16), fairness index oscillates between 0.55 and 0.95 in the phase during which the number of active connections grows. This behaviour is detected for both TCP Reno and TCP Vegas. As a new connection accesses return link, TCP point of equilibrium, meant as the fair transmission rate allowing an equal resource sharing among active connections, decreases, fairness results reduced since the pre-existing connections still operate around the old point of equilibrium, while the new point operates well below such a threshold. The larger the number of active connections, the lower the variation of the point of equilibrium. This explains the reason for which fairness oscillations are damped down during the occurrence of the highest traffic load (from 40 to 80 s). On the other hand, when a connection stops, the remaining

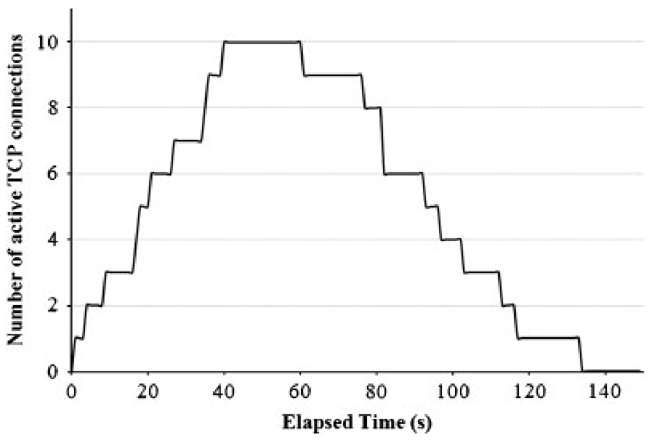

Figure 15. TCP traffic pattern.

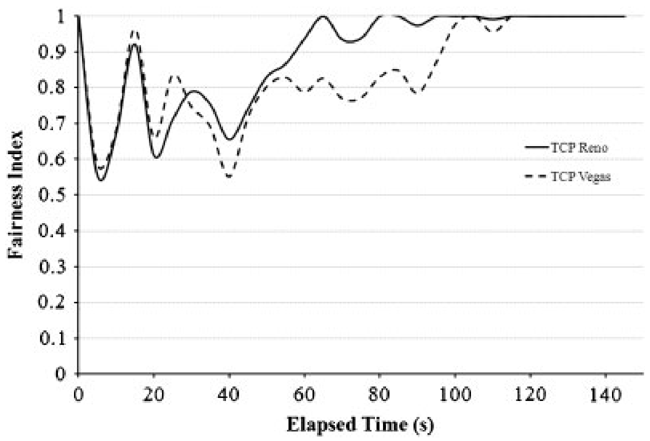

Figure 16. Fairness index over CRA for 10 TCP connections with random start time and duration (uniform distribution).

connections increase their rates starting from the same value. Then, fairness usually does not worsen. In this phase, outcomes demonstrate that TCP Reno outperforms TCP Vegas converging much more quickly to the optimum fairness, intrinsically achieved when only a connection is running.

Figure 17 proposes the same test but envisages RBDC access scheme. All the considerations made for the CRA case are still valid. In addition, TCP Vegas is always characterized by lower fairness index values.

5.3. T_III tests

Friendliness represents a parameter utilized in this case to evaluate the impact that TCP Vegas flows have on the performance of TCP Reno flows. As a reference performance, TCP Reno stand-alone flows are considered.

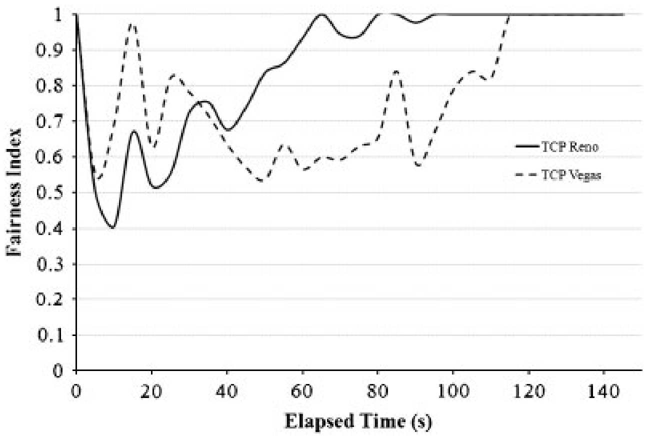

Figure 17. Fairness index over VBDC for 10 TCP connections with random start time and duration (uniform distribution).

Friendliness is evaluated through a simple experiment:

- At the start, three TCP Reno flows are run;
- At 100 s, three TCP Vegas flows are introduced;
- At 150 s, TCP Reno flows leave the channel;
- At 250 s, the three TCP Reno flows access again the channel so that they contend the bandwidth with the three TCP Vegas flows until simulation ends.

Clearly, analysis of friendliness makes sense when TCP Reno and TCP Vegas flows are simultaneously active but we aim to reproduce two cases:

1. TCP Vegas flows take over when TCP Reno flows are in steady state;
2. TCP Reno flows take over when TCP Vegas flows are in the steady state.

The set from Figures 18–20 shows that in all the cases TCP Vegas flows do not particularly affect TCP Reno performance. In fact, throughput of the TCP Reno flows is left quite unchanged after the start of the TCP Vegas flows (from 100 to 150 s). When TCP Reno flows are restarted at 250 s, their throughput increases irrespective of the presence of the TCP Vegas flows. In other words, TCP Vegas flows reduce their rate in advantage of TCP Reno flows. When only the three TCP Vegas flows run (from 150 to 250 s), they are able to use all the available bandwidth only with CRA, while in the other cases lead to an overall throughput lower than the available capacity (Figures 19 and 20). This confirms what is seen in the utilization analysis (Figures 12–14).

In the reproduced simulation scenario, TCP Vegas is ‘friendly’ with TCP Reno, in the sense that TCP Vegas connections leave all the bandwidth to TCP Reno connections as much as they need. In other words, TCP Vegas behaves as a ‘low-priority’ TCP when competing bandwidth with TCP Reno. In this case, fairness among all the flows is desired, and simultaneous running of TCP Reno and TCP Vegas connection is strongly inopportune.

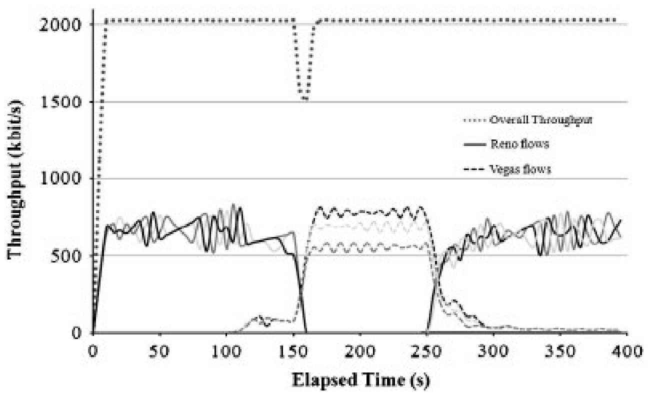

Figure 18. TCP friendliness between Reno and Vegas with CRA.

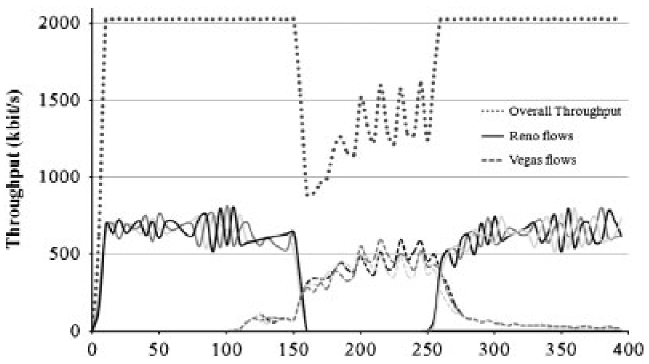

Figure 19. TCP friendliness between Reno and Vegas with RBDC.

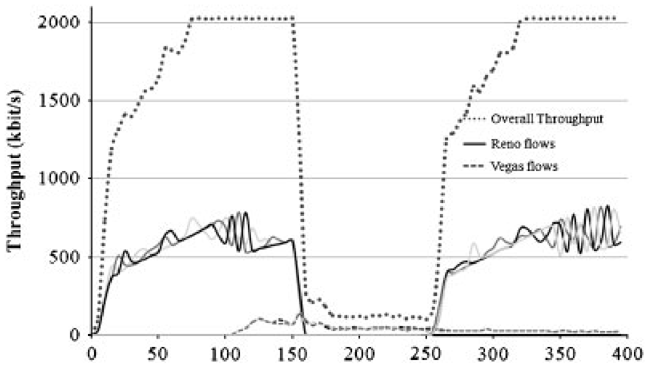

Figure 20. TCP friendliness between Reno and Vegas with VBDC.

5.4. *T_IV tests*

In order to evaluate the performance of real applications over DVB-RCS links, the time needed to transmit a file (i.e. FTP transfer) is calculated for different file sizes.

First, simulation results related to TCP Reno over return link have been compared with those coming from the direct calculation (dotted lines) as presented in Section 3.2. In particular, in Figure 21 overlapping curves represent the file transfer time obtained using Equation (9) with the average RTT from Figure 8 and those obtained by simulation. For the sake of verifying the validity of Equation (9) as far as time transfer computation in both SS and CA is concerned, a generic ssthresh equal to 20 Maximum Segment Size (MSS) is considered. The matching is good enough to demonstrate that analytical calculation assuming a theoretical burst transmission and considering an average RTT value can be satisfactory. In particular, upper bounds of the variations are: 5% in CRA, 3% in RBDC and 19% in VBDC. As far as performance is concerned, file transfer time with RBDC increases by less than 50% with respect to CRA, while in VBDC the time increases by 220%.

5.5. *T_V tests*

With reference to HTTP model presented in Section 4.3, HTTP transfers have been reproduced. Considering different tested environments, several HTTP connections to transfer web pages with different object sizes and numbers have been established. The tested environments include CRA, RBDC and VBDC allocation schemes with both TCP Reno and Vegas. The same test has been repeated with different web page size classes, from relatively small (34.5 kbyte) to medium (144 kbyte) to large (440 kbyte). The random variables used to describe such classes resulted in an average of eight objects per connection, with average size per object, respectively, of 4.4, 18 and 55 kbyte.

Single web page transfer times for each test environment and for each page size class have been averaged to best represent the specific environment statistical performance. Summary results are shown as a bar chart in Figure 22.

It is evident that DAMA allocation strategies and TCP version strongly impact the transfer time for all classes of web page size. Although the effect of DAMA access delay is limited when

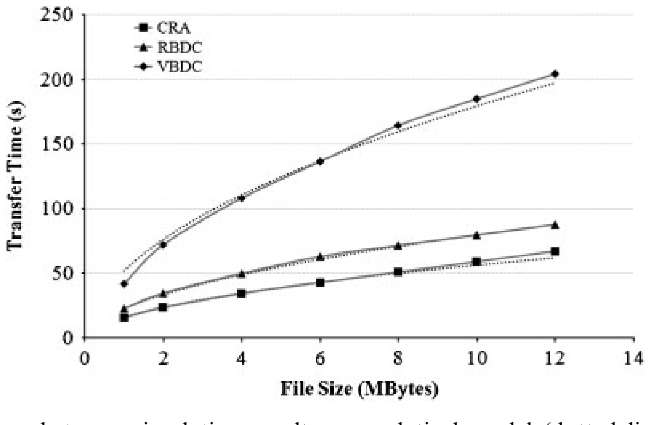

Figure 21. Comparison between simulation results vs analytical model (dotted lines); TCP Reno over return link.

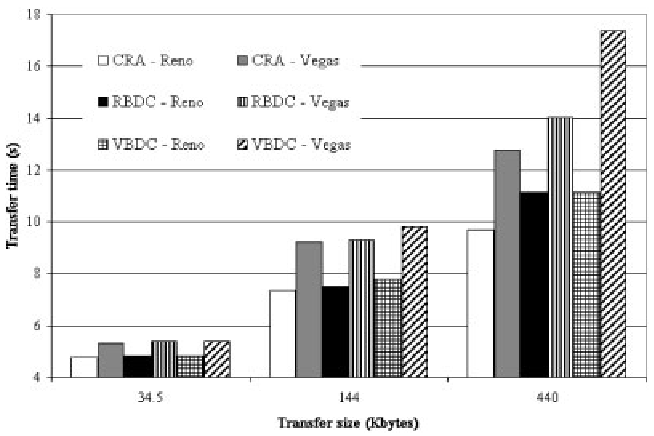

Figure 22. HTTP transfer times.

transmitting small web pages due to the relatively small number of ACK packets that travel on the return link (only one to two slots are mostly needed to accommodate all the ACKs), TCP timing based on RTT results has a significant impact on performance. As a consequence, web pages of just few dozen of kbytes (34.5 in average) need more than 4 s to be transferred. Differences among both considered TCP and DAMA schemes are quite limited, since the whole transfer is completed within few RTT with cwnd still assuming small values.

When web page size increases, the difference of performance when CRA, RBDC or VBDC is used is more meaningful, as well as the relative difference between TCP Reno and Vegas. When the web page has an average size of 440 kbyte, the trend observed in the previous tests is definitively confirmed, with significant transfer time differences for different DAMA allocation schemes and TCP versions. As worst-case comparison, the transfer of a web page can take from less than 10 s when using TCP Reno and CRA allocation, to more than 17 s when using TCP Vegas with VBDC requests.

5.6. Summary

In all the tests performed, CRA scheme guarantees the best performance at transport layer (goodput) at the cost of an inefficient use of resources, while VBDC allows a very high network efficiency (η_N) but very poor performance at transport layer. RBDC represents a trade-off between performance and efficiency. TCP Vegas is particularly affected by DAMA dynamic allocation process discouraging its use with RBDC and VBDC. In general, TCP Reno outperforms TCP Vegas in terms of throughput and efficiency. In case of multiple flows, TCP Vegas presents a higher fairness than TCP Reno (however, satisfactory), while the maximum channel utilization is achieved later compared with TCP Reno over CRA. Furthermore, results make evident that TCP Reno and TCP Vegas cannot be used simultaneously since TCP Vegas tends to leave all the bandwidth to TCP Reno flows.

For file/e-mail transfers (several Mbytes) over the return link, the best-performing solution is to use TCP Reno over RBDC. CRA represents the best solution, but leads to a network efficiency as low as 10%. TCP Vegas allows acceptable performance only with CRA. In case of file/e-mail transfer over the forward link, performance is flattened for all the considered access schemes (return link used to send ACKs). In this case it is then convenient to set DAMA in VBDC mode in order to maximize the efficiency. Web traffic has been supposed only in the forward link since HTTP servers are usually across the Internet and then accessible through the satellite gateway. In the case of TCP Reno, the time for the download is quite independent from the used access schemes. The rationale is that return link is accessed to send just a small amount of ACKs. For larger average object size, the performance gap among CRA, RBDC and VBDC increases, but it is always limited. TCP Vegas performance is slightly worse, although of the same order of magnitude. In both cases, the VBDC scheme is sufficient to have performance comparable to that with CRA, while efficiency in the return link is maximized.

6. CONCLUSIONS

This paper has presented a detailed analysis of the performance of both TCP and related applications when DAMA resource allocation schemes are adopted in DVB-RCS systems. The main goal has been to analyse the effects of the interaction between TCP-based congestion controls, supported by I-PEP, and some basic implementations of DVB-RCS DAMA schemes. Furthermore, the proposed analysis has addressed applicative scenarios where multiple TCP flows compete for return link capacity. Results, expressed in terms of channel utilization, fairness and friendliness, arise some issues concerning I-PEP's transport protocol configuration: i.e. TCP problems with HTTP traffic, TCP Vegas under-utilization over RBDC and VBDC, lack of fairness under several traffic profiles, the excessive friendliness of TCP Vegas with respect to TCP Reno making their co-existence between I-PEP ineffective, etc. Therefore, the most important conclusion is that both I-PEP transport protocol and DAMA optimization will need to take care of a large set of factors such as the nature of the traffic, the network load and Quality of Service (QoS) required by each terminal. In this frame, the proposed work provides the guidelines for seeking an access scheme suitable for the target communication scenario, showing TCP performance under a large set of study cases. In the future, this work can be used as a baseline to identify requirements for the design of a TCP based congestion control scheme optimized to operate over different DAMA schemes and providing connection-based data stream at different traffic sources.

ACKNOWLEDGEMENTS

This work has been partially supported by the European IST-FP6 project 'SatNEx II—Satellite Communications Network of Excellence II'.

REFERENCES

1. ETSI. Digital Video Broadcasting (DVB): DVB framing structure, channel coding and modulation for 11/12 GHz satellite services. *EN 300 421*, 1997.
2. ETSI. Digital Video Broadcasting (DVB): interaction channel for satellite distribution systems, DVB-RCS standard. *EN 301 790*, 2008.
3. ETSI. Digital Video Broadcasting (DVB): interaction channel for satellite distribution systems. *Guidelines for the Use of EN 301 790*, *TR 101 790*, *V. 1.2.1*, 2003.

4. Balakrishnan H, Padmanabhan VN, Sooriyabandara M. TCP performance implications of network path asymmetry. *RFC 3449*, December 2002.
5. Stevens W. *TCP/IP Illustrated*, vol. 1. Addison-Wesley: Reading, MA, 1994.
6. Postel J. Transmission control protocol. *Internet RFC 793*, 1981.
7. *The Network Simulator Ns-2*. Available from: http://www.isi.edu/nsnam/ns.
8. Interoperable PEP (I-PEP). Transport extensions and session framework for satellite communications: air interface specification. October 2005.
9. The Satlabs group. Available from: http://satlabs.org.
10. ETSI. Digital Video Broadcasting (DVB): user guidelines for the second generation system for broadcasting, interactive services, news gathering and other broadband satellite applications (DVB-S2). *TR 102 376 V1.1.1*, February 2005.
11. ETSI. Digital Video Broadcasting (DVB): second generation framing structure, channel coding and modulation systems for broadcasting, interactive services, new gathering and other broadband satellite applications. *EN 302 307 v1.1.2*, June 2006.
12. ATM Forum. ATM specifications. Available from: http://www.ipmplsforum.org/tech/atm_specs.shtml.
13. Partridge C, Shepard T. TCP performance over satellite links. *IEEE Network* 1997; **11**(5):44–49.
14. Allman M, Hayes C, Kruse H, Osterman S. TCP performance over satellite links. *The 5th International Conference on Telecommunication Systems Modeling and Design*, Nashville, TN, 1997; 1–13.
15. Zhang Y. *Interworking and Computing over Satellite Networks*. Kluwer Academic Publishers: Dordrecht, 2003.
16. Lakshman T, Madhow U. The performance of TCP/IP for networks with high bandwidth-delay products and random loss. *IEEE/ACM Transactions on Networking* 1997; **5**(3):336–350.
17. Xylomenos G, Polyzos GC, Mahonen P, Saaranen M. TCP performance issues over wireless links. *IEEE Communication Magazine* 2001; **39**(4):52–58.
18. Charalambous C, Frost V, Evans J. Performance of TCP extensions on noisy high BDP networks. *IEEE Letters on Communications* 1999; **3**(10):294–299.
19. Allman M, Glover D, Sanchez L. Enhancing TCP over satellite channels using standard mechanism. *RFC 2488*, January 1999.
20. Mogul J, Deering S. Path MTU discovery. *RFC 1191*, November 1990.
21. Mathis M, Floyd S, Romanov A. TCP selective acknowledgment options. *RFC 2018*, October 1996.
22. Caini C, Firrincieli R. A new transport protocol proposal for Internet via satellite: the TCP hybla. *Proceedings of the ESA ASMS 2003*, Frascati, Italy, vol. SP-54, July 2003.
23. Border J, Kojo M, Griner J, Montenegro G, Shelby Z. Performance enhancing proxies intended to mitigate link-related degradations. *RFC 3135*, June 2001.
24. Jacobson V, Braden R, Borman D. TCP extensions for high performance. *RFC 1323*, May 1992.
25. Brakmo L, Peterson L. TCP Vegas: end to end congestion avoidance on a global Internet. *IEEE Journal on Selected Areas in Communications* 1995; **13**(8):1465–1480.
26. Space communications protocol specification (SCPS)—transport protocol (SCPS-TP). *Recommendation for Space Data System Standards, CCSDS 714.0-B-2. Blue Book*, Issue 2. CCSDS: Washington, DC, October 2006.
27. Fox J. TCP big window and NAK option. *RFC 1106*, June 1989.
28. Stevens W. TCP slow start, congestion avoidance, fast retransmit, and fast recovery algorithms. *Internet RFC 2001*, 1997.
29. Jacobson V. Congestion avoidance and control. *Proceedings of the ACM SIGCOMM '88 Conference*, Stanford, CA, 1998; 314–329.
30. Roseti C, Kristiansen E. TCP behaviour in a DVB-RCS environment. *Proceedings of the 24th AIAA International Communications Satellite Systems Conference* (*ICSSC*), San Diego, CA, June 2006.
31. Karaliopoulos M, Tafazzoli R, Evans BG. Providing differentiated service to TCP flows over bandwidth on demand geostationary satellite networks. *IEEE Journal on Selected Areas in Communications* 2004; **22**(2): 333–347.
32. Gotta A, Secchi R, Potortì F. Simulating dynamic bandwidth allocation on satellite links. *Proceedings of the International Conference on Performance Evaluation Methodologies and Tools* (*Valuetools*). ACM: Pisa, Italy, October 2006; 8.
33. *Internet Study 2007*, 2007. Available from: http://www.ipoque.com/.
34. Fielding R, Gettys J, Mogul J, Frystyk H, Masinter L, Leach P, Berners-Lee T. Hypertext transfer protocol—HTTP/1.1. *RFC 2616*, June 1999.
35. Luglio M, Roseti C, Gerla M. TCP performance over satellite in case of multiple sessions per links using efficient flow control and real OS. *10th Ka and Broadband Communications Conference*, Vicenza, Italy, 30th September–2nd October 2004; 253–260.
36. Sooriyabandara M, Fairhurst G. Dynamics of TCP over BoD satellite networks. *International Journal of Satellite Communications and Networking* 2003; **21**:427–449.
37. Jain R, Hawe W, Chiu D. A quantitative measure of fairness and discrimination for resource allocation in shared computer systems. *DEC-TR-301*, 26th September 1984.

AUTHORS' BIOGRAPHIES

M. Luglio received the Laurea degree in Electronic Engineering at University of Rome 'Tor Vergata'. He received the PhD degree in telecommunications in 1994. From August to December 1992 he worked, as visiting Staff Engineering at Microwave Technology and Systems Division of Comsat Laboratories (Clarksburg, Maryland, USA). He received the Young Scientist Award from ISSSE 95. From 1995 to 2004 he was research and teaching assistant at University of Rome 'Tor Vergata'. At present he is associate professor of telecommunication at the same university. He works on designing satellite systems for multimedia services both mobile and fixed, in the frame of projects funded by EC, ESA and ASI. He taught Signal Theory and collaborated in teaching Digital Signal Processing and Elements of Telecommunications. In 2001 and 2002 he was visiting Professor at the Computer Science department of University of California Los Angeles (UCLA) to teach Satellite Networks class. Now he teaches Satellite Telecommunications and Signals and Transmission.

C. Roseti graduated cum laude in 2003 in Telecommunication Engineering at University of Rome 'Tor Vergata'. In 2003 and 2004, he was a visiting student at Computer Science Department of University of California, Los Angeles (UCLA). From August to December 2005 he worked at the TEC-SWS division of the European Space Agency (Noordwijk, The Netherlands). He received the PhD degree in 'Space systems and technologies' in 2007. He is currently a research fellow at the University of Rome 'Tor Vergata', teaching in the 'Laboratory of signal processing' and collaborating in both 'Satellite Telecommunications' and 'Signal Processing' class. His research interests include satellites communications and protocol design, cross-layer interactions, security, HAPS/UAV communications, protocol implementation and performance analysis in wired/wireless networks. He is involved in national and international projects in these fields.

F. Zampognaro was born in Palermo, Italy, in 1976. He received the Dr Ing degree in Telecommunications Engineering (summa cum laude) and the PhD degree in Electronics, Computer Science, and Telecommunications from the University of Bologna, Bologna, Italy, in 2001 and 2005, respectively. He is currently a Postdoctoral Researcher with the Department of Electronics, Computer Science, and Systems (DEIS), and the Advanced Research Centre for Electronics Systems (ARCES), at the University of Bologna. In 2003 and 2005, he was a Visiting Scholar with the University of California, San Diego. His main research interests include the area of communication theory for both satellite and terrestrial systems, with particular emphasis on acquisition and synchronization for CDMA, TDMA, and UWB systems. Dr Villanti was a co recipient of the Best Student Paper Award from IEEE ISSSTA 2004, Sydney, Australia, and the Best Paper Award from IEEE ISWCS 2005, Siena, Italy.