A Reliable Multicast Transport Protocol with Feedback Implosion Suppression for Satellite Communication Systems

Gün Akkor and John S. Baras

***Abstract* — In this paper, we propose a reliable multicast transport protocol for satellite communication systems. Many of the emerging applications in the Internet would benefit from reliable multicast services, and broadband satellite communication systems have attractive characteristics for supporting such services. However, many of the protocols designed primarily for terrestrial networks do not perform well over satellite networks. Therefore, it is necessary to look at the problem of reliable multicast in the solution space of satellite communications. Our protocol makes use of a special form of forward error correcting codes and couples it with an adaptive window based control mechanism to dynamically adjust the number of encoding packets forwarded to the users. Protocol makes very good use of the broadcast nature of the satellite channel and attempts to minimize the feedback volume by a novel feedback implosion suppression algorithm. We evaluate the protocol performance by computer simulations.**

***Index Terms* — reliable multicast, satellite networks, feedback implosion suppression, transport protocols.**

I. INTRODUCTION

MANY of the emerging applications in the Internet, such as distributed computing, software updates, distance learning, and Internet gaming would benefit from reliable multicast services. These applications are distributed in nature and require concurrent transmission of the same content to multiple users. Broadband satellite communication systems offer wide-area coverage and ubiquitous access to a potentially large number of users. Therefore, they are a natural option for carrying such services. However, many of the protocols, designed primarily for terrestrial networks, do not perform well over satellite networks. TCP, which is the dominant protocol in the Internet for reliable delivery of data, for example, suffers from performance degradation over satellite links due to long propagation delays and high loss rates [1]. It also does not scale well to concurrent transmission of the same content to multiple users because of *feedback implosion* [2], and *loss path multiplicity* [3] problems. Therefore, it is necessary to look at the problem of reliable multicast in the solution space of satellite communications.

In this paper, we propose a reliable multicast transport protocol that specifically addresses the transport level issues in the context of satellite communication systems. The paper is organized as follows. In the next section, we discuss related work on this topic. Section III describes our network architecture. In Section IV and V, we present the details of our protocol. Simulation results are discussed in Section VI and Section VII concludes the paper.

II. RELATED WORK

In order to protect data against channel errors and avoid retransmission of packets over high latency satellite links, primary focus on this topic has been on providing some form of forward error protection by proactively transmitting redundant information along with the forwarded data. Forward error protection helps recover corrupted data and thus minimizes the need for retransmissions over the satellite links. For multicast services, application of forward error correction (FEC) coding at the packet level has been shown to improve protocol performance and scalability. In [2] and [4], the authors show the potential benefits of packet level FEC coding on a generic reliable multicast protocol. In [5] and [6], the authors discuss the integration of packet level FEC in the context of satellite systems. In [7], the author proposes transmitting redundant packets using a separate channel along side the original data packets. In [8] and [9], the authors propose schemes for adaptively adjusting the number of parity packets transmitted to the receivers. Implementation of packet level FEC coding is difficult for large packet sizes. Also, the number of parity packets that can be transmitted against packet losses is limited to the block size of the code. Therefore, in [10] the

The material is based upon work supported by NASA under award number NCC8235. Any opinions findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Aeronautics and Space Administration.

Gun Akkor is with the Electrical and Computer Engineering Department and the Institute for Systems Research of University of Maryland, College Park, MD 20742 USA (akkor@isr.umd.edu).

John S. Baras is with the Electrical and Computer Engineering Department and the Institute for Systems Research of University of Maryland, College Park, MD 20742 USA (baras@isr.umd.edu).

authors propose a scheme using Tornado codes, which are capable of working over large block sizes.

Our protocol favors integration of packet level FEC coding at the transport layer. We use a special form of packet level FEC codes, namely the LT codes [11], which are capable of generating a large number of encoding packets for the same group of input packets. This flexibility in the number of encoding packets improves the efficiency of the protocol considerably. We improve the ideas in [5] and [6], and build a novel control mechanism that dynamically adjusts the number of encoding packets transmitted for each input packet group.

Packet level FEC coding alone is not sufficient for providing full reliability and a reliable transport protocol has to still rely on feedback from the users. Current design alternatives for feedback implosion avoidance primarily depend on whether an alternative terrestrial path is available for transmission of user feedback. When terrestrial links are available, user feedback could be sent to the source over the terrestrial links. Feedback implosion could be avoided on these paths by using feedback scoping and aggregation techniques. However, when no such alternative paths exist, we have to consider a satellite system with bi-directional links. In this case, uplink channel resources have to be accessed and shared by all receivers. To our knowledge, proposals described in [12] and [13] are among the first papers to address feedback implosion avoidance problems in a satellite only context.

Our protocol implements a novel feedback implosion suppression algorithm that allows the use of a fixed number (<< number of receivers) of uplink return channels by the multicast session, while ensuring that the critical information is conveyed to the source in a timely fashion.

III. NETWORK DESCRIPTION

We consider a satellite communication system, where a Ka-band satellite provides broadband services to a large number of users located inside its footprint. In this scenario, users that are equipped with two-way direct communication terminals access the terrestrial network through a *gateway* node referred to as the network operations center (NOC). The structure of the network is shown in Fig. 1.

We assume that the data is to be distributed reliably to many end-users, and that data are located at a server on an IP-based terrestrial network. We are primarily interested in providing reliability over the satellite-only portion of the network, and assume that connection is established using *TCP connection splitting* [14]. The DVB standard for data broadcasting with Multiple Protocol Encapsulation (MPE) is used for transmission of packets over the satellite network [15]. MPE allows encapsulation of IP datagrams into 188 Byte MPEG-2 Transport Stream packets at the NOC. The proposed breakup of the communication session is shown in Fig. 2.

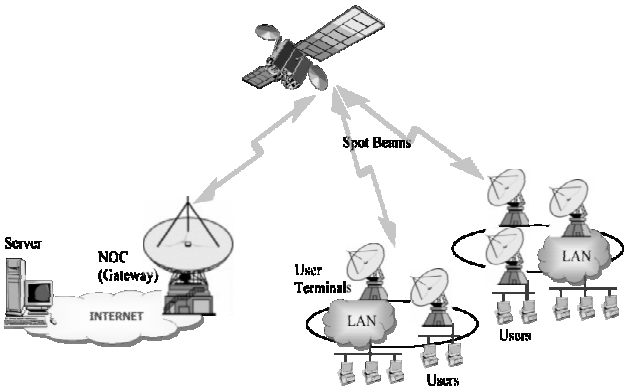

Fig. 1. Network Architecture

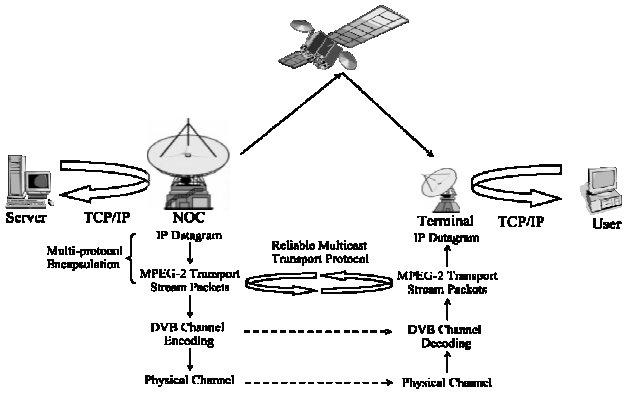

Fig. 2. Communication Session

IV. PROTOCOL OUTLINE

Our protocol implements a special class of packet level FEC codes, namely the *Luby Transform* (LT) codes [11]. This coding is applied at the packet level and assists in packet retransmissions rather than guarding against channel errors. It is performed at the MPEG-2 level, and is applied to the MPEG-2 transport stream packets. Let k denote the number of MPEG-2 transport packets currently buffered at the satellite gateway. The LT encoder outputs individual encoding packets using this batch of k MPEG-2 packets by: (i) randomly choosing the degree d of the encoding packet from a degree distribution, (ii) choosing, uniformly at random d distinct input packets as neighbors of the encoding packet, and (iii) calculating the value of the encoding packet as the exclusive-or (XOR) of the d

neighbors. Each encoding packet, generated in this manner, is uniquely identified by its header and has the same packet size as the input packets. The encoding packets are further encoded by the DVB channel encoder and are transmitted over the channel.

LT codes allow end-users to recover the original batch of k transport packets, if they manage to accumulate a slightly larger set of $K = (1+\varepsilon)\cdot k$ encoding packets that were generated from the same input packet batch. Additionally, the number of encoding packets that can be generated from the same input packet batch is typically much larger than the size of the batch. The benefit of this construction is two fold. First, it allows the satellite gateway to transmit additional encoding packets as long as there are users that have not yet accumulated enough packets for the batch. Secondly, it simplifies the user request for additional packets, since now the only information that has to be forwarded to the satellite gateway is comprised of the number of additional encoding packets (from the same batch) that is required by each user until all of them (users) complete the reception. This construction also makes very good use of the broadcast nature of the satellite channel since every transmitted encoding packet benefits all the users equally.

The transport protocol, in the first round, generates a set of $K = (1+\varepsilon)\cdot k$ encoding packets for an input batch of k MPEG-2 transport packets and transports them over the satellite channel. Each user, at the end of this initial round, evaluates its success and reports the number of additional encoding packets required to complete its reception. The satellite gateway collects these reports and in the second round, transmits enough encoding packets to accommodate the need of the worst-case user (the one that requested the most packets in the first round). In the subsequent rounds, the transmission proceeds in rounds until all receivers successfully complete their reception of the batch. In the next section, we will describe the behavior of our reliable transport protocol.

V. PROTOCOL DETAILS

A. Protocol Behavior at the Satellite Gateway (*NOC*)

We assume that data can be organized into B batches of k input packets. End-users have to accumulate at least K encoding packets in order to successfully recover any input batch. Therefore, for each batch $B_i, i = 1,2,\ldots,B$, the protocol has to transmit *at least K* encoding packets in the first round. The number of encoding packets transmitted in the first round of each batch transmission is controlled by the *transmission window* W_1. The minimum value of W_1 is always equals to $W_1^{\min} = K$. A user may fail to complete the recovery process of a batch after the first round, if the packet loss over the channel is too severe, and not enough packets have been accumulated. In this case, additional encoding packets are transmitted in the subsequent rounds. A second window, W_2, controls the transmission of additional packets for failed batches. In the best case, all batches are recovered at the end of the first round. Therefore, W_2 has a minimum value $W_2^{\min} = 0$. The transmission of the packets in windows W_1 and W_2 follows a time division, where the transmission of packets in window W_1 is followed by the transmission of the packets in window W_2. One of the goals of the protocol is to try to minimize the value of $W = W_1 + W_2$ per transmitted input batch over the lifetime of the session.

Users send feedback reports back to the source based on their status of recovery. When the protocol receives a feedback report from receiver $r_j, j = 1,2,\ldots,R$, it gets two pieces of information: (i) a request for b_{ij} additional encoding packets for batch B_i that has previously failed in the recovery process, and (ii) a weight value w_j, which is the average of additional encoding packet requirements for all the failed batches at the user, at the time of the transmission of the report. The weight value of a user is an indicator of how well the user succeeds in the recovery process and it is used at the gateway side of the protocol to adjust the size of the transmission windows (W_1 and W_2). The protocol generates

$$b_i^{\max} = \max_{j \in R}\{b_{ij}\} \quad (1)$$

additional encoding packets for batch B_j and stores them in a request queue (RQ). The weight value of the user (as stored at the gateway) is also updated to the value last reported. From the user perspective, the protocol must make sure that every batch can be recovered at the end of the first round of transmission, because additional rounds add to the overall delay. At the same time, the protocol should transmit as few encoding packets as possible (ideally completing the recovery after transmitting K encoding packets). Satellite channels, like many other wireless communication systems, are time-variant. Therefore, the protocol has to adjust its transmission windows so that it transmits enough encoding packets for every batch in the first round for high recovery probability while avoiding to transmit more than the encoding packets needed by the user group.

Packet transmissions and the size of the transmission windows are adjusted according to the following rules at every transmission round:

i. At round n, $W_2(n)$ is set to the current size of the request queue:
$$W_2(n) = \sum_{i \in A(n)} b_i \,, \qquad (2)$$
where, $A(n)$ is the set of batches for which there exists a request in the queue. i.e. all requests that are in the queue are served at the next transmission round.

ii. Let $c_1(n) = W_2(n) / |A(n)|$ be the average number of additional encoding packets requested per batch as observed at start of round n. Let $c_2(n) = w_{\max} \cdot |A(n)|$ be an estimate of the maximum number of additional encoding packets that may be requested for a set of batches of cardinality $|A(n)|$, where $w_{\max} = \max_{j \in R} \{w_j\}$. Then the size of $W_1(n)$ is set to:
$$W_1(n) = \min\{W_1(n-1) + c_1(n), K + c_2(n)\}, \qquad (3)$$
if $W_1(n) < \min\{W_1(n-1) + c_1(n), K + c_2(n)\}$. Else, it is decremented: $W_1(n) = W_1(n) - 1$.

iii. If $n \le B$, then $W_1(n)$ packets are encoded and transmitted for batch n, followed by $W_2(n)$ additional encoding packets. Else, only $W_2(n)$ additional encoding packets are transmitted in response to requests.

iv. Following the transmission of $W(n) = W_1(n) + W_2(n)$ encoding packets, the NOC protocol broadcasts a control packet, COLLFDB, to poll all users on the success of the current round of transmission.

v. The protocol waits for one round-trip time to collect new requests from users. It then proceeds with the next round.

The transmission continues in rounds until all users indicate that they have completed all batches successfully.

B. Protocol Behavior at the End-Users

At the beginning of the transmission, the user protocol initializes the weight value of the user to $w_j = 0, j = 1,2,\ldots,R$. As encoding packets for the input batches are received, they are stored in temporary buffers awaiting the start of the recovery process. Whenever the user protocol detects the start of encoding packets for a new batch, or it receives the COLLFDB control packet, it starts the recovery process for the last transmitted batch. The user protocol polls the temporary buffers for all batches that are now in the recovery phase. If the recovery process of any batch is successful, the input packets are forwarded to upper layers for reassembly. For all batches for which the recovery process has failed, the protocol files requests in the users' request buffer (RQ_U) for additional encoding packets or updates a previously filed requests to reflect the new packet requirements.

After temporary buffers are polled and completed batches are removed from them, the user protocol calculates the new weight value of the user by calculating the average of the packet requirements of all batches that have failed in the recovery process. Up on receiving the COLLFDB control packet, the user protocol transmits the first request in the request buffer together with the new weight value of the user. The session continues as described until all batches complete the recovery process. At this time, the user protocol transmits a TERM control packet to the gateway protocol to terminate the session.

C. Feedback Suppression Policy

Looking at the return information, we observe that, it is sufficient for the NOC to only track, $b_i^{\max}, \forall i$, the maximum number of encoding packets requested per encoding block. The volume of feedback would be *minimized* if only the user with the maximum packet requirement responds to the NOC per encoding block. We can consider two extreme scenarios for this situation: (i) every user is assigned a separate return channel to communicate its request information to the NOC, and NOC computes the maximum as in Eq. (1); (ii) all users communicate among each other through a secondary network (possibly a terrestrial connection) and suppress the feedback of all users but the one with the maximum. The former scenario gives rise to feedback implosion problem as well as the waste of uplink resources, while the latter scenario requires additional infrastructure and collaboration between the receivers and is contradictory to reasons for deployment of satellite networks.

Our feedback suppression policy minimizes the number of transmitted reports via a knapsack-based algorithm that runs at the NOC without relying on any collaboration among users. We assume that $m << R$ return channels are allocated to the multicast session. The problem is, then, to determine at every uplink transmission, which of the R multicast users will be allocated a channel to transmit a request in one of the m channels. The NOC solves this problem by employing a *knapsack* algorithm. We view each available channel as a knapsack with capacity $c_k = \max\{w_1, \ldots w_R\} = c$ for $k = 1,2,\ldots,m$ and each user as

an item with weight w_j and profit $p_j = w_j$ for $j = 1,2,\ldots R$, where R is the number of users. We solve the following multiple-knapsack problem:

$$\begin{aligned} &\max \sum_{k=1}^{m}\sum_{j=1}^{R} p_j x_{kj} \\ &\text{subject to} \sum_{j=1}^{R} w_j x_{kj} \le c_k = c \text{ for k} = 1,\ldots,\text{m} \\ &\sum_{k=1}^{m}\sum_{j=1}^{R} x_{kj} \le 1 \end{aligned} \quad (4)$$

In the solution of this problem, $x_{kj} = 1$, if user r_j is assigned to return channel k. If a user has weight, w_j that is closer to c, it is more likely to be assigned to a channel alone, since it fills the capacity of the knapsack (channel). A higher weight indicates that the user on the average needs a higher number of additional encoding packets to complete the pending batches and hence its feedback will probably cover the requirements of the group as well. Therefore it is feedback is given priority and is assigned to a channel alone. Users with smaller weights will be sharing the channel with other receivers. A smaller weight indicates that the user needs, on the average, only a few encoding packets to complete the pending batches and hence its feedback will only partially cover the group. In order to avoid collisions in the channels shared by more than one user, users transmit their reports with probability proportional to their current weights, given by:

$$p_r(w_r) = \frac{\exp(\mu \cdot w_r / c)}{\exp(\mu)} \quad (5)$$

where, $\mu = 1 - m/R$.

The solution vector is broadcast to all users when the NOC transmits the COLLFDB control packet. The users transmit their request according to the channel assignment broadcasted in the control packet.

VI. SIMULATION RESULTS

A. Simulation Environment

We use NS-2 Network Simulator to simulate the behavior and performance of our protocol. The satellite node is a geo-stationary satellite located at 100° W longitudes, and the ground terminal nodes are located between 30°-45° N latitudes and 80°-122° W longitudes. One of the ground terminal nodes acts as the source and runs the satellite gateway (NOC) side of the protocol, while the remaining nodes act as sinks and run the user side of the protocol. We assume that the satellite gateway has a fixed number $B = 50$ input batches, each with $k = 220$ MPEG-2 transport stream packets. The LT coding instance requires $K = 235$ encoding packets to be transmitted for a successful recovery of an input batch. Therefore, $W_1^{\min}$ is set to 235. The one-way propagation delay between the satellite node and ground terminal nodes is 150 milliseconds and the effective link rate at the MPEG-2 transport layer is 4 Mbps.

B. Channel Model

The channel model we use in our simulations is a simple threshold-based Markov Chain model with six states of 5dB to 30dB attenuation at 5dB intervals. The probability of observing a particular attenuation threshold and the duration of the attenuation state are taken from the statistical work carried out by German Aerospace Center (DLR) with the 40 GHz beacon of the Italian satellite ITALSAT [16]. For simplicity, we assume that for signal attenuation of less than or equal to 10dB, channel coding and the link-budget power margin are capable of compensating for the signal fade, and therefore, no packet losses occur at the transport layer. For signal attenuation of 15dB and more all packets are corrupted.

C. Results

In this section, we provide simulation results on the performance of our protocol when combined with the feedback implosion suppression algorithm. In this set of simulations, multicast group size is kept fixed at $R = 100$ while the number of available return channel assigned to the session is varied. In Fig. 3, we plot the average size of transmission windows W_1 and W_2 as a function of the number of available return channels. We see that the average window size per transmitted batch remains quite steady, i.e. the protocol maintains the same performance level with the feedback implosion suppression in place. In Fig. 4, we plot the average reception efficiency and the average delay per batch per receiver as a function of the number of available channels. The results show that the receiver side performance metrics remain steady as well. Finally, in Fig. 5, we show that it is possible to reduce the volume of feedback by as much as %40 without significant performance degradation. Fewer return channels, however, results in more receivers being assigned into the same return channel. Therefore, number of feedback packet collisions increases as number of available return channels is decreased.

VII. CONCLUSIONS

In this paper, we have introduced a reliable multicast transport protocol that operates at the satellite-only tier of a hybrid satellite-terrestrial communication system for reliable delivery of data to end-users. The protocol effectively uses

a special class of FEC codes and complements it with a dynamic window management scheme to adjust number of

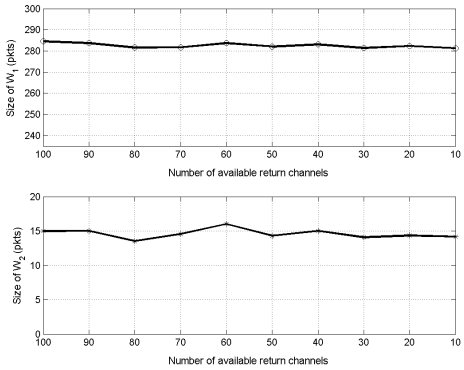

Fig. 3. Size of transmission windows versus number of available return channels for a receiver set of R=100.

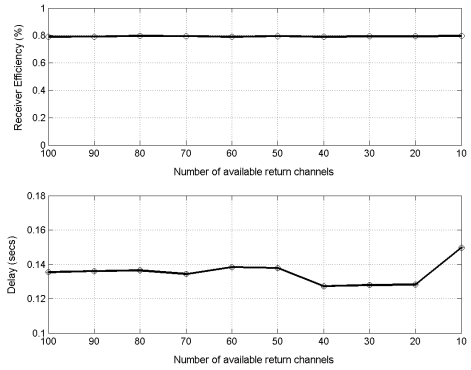

Fig. 4. Receiver bandwidth efficiency and delay versus number of available return channels for a receiver set of R=100.

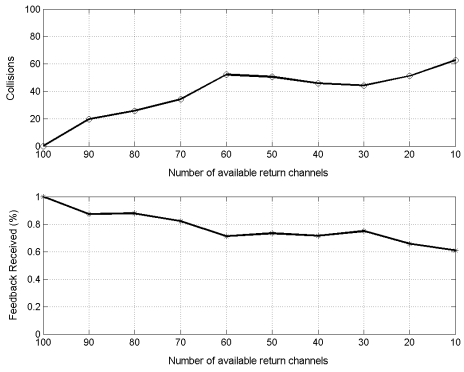

Fig. 5. Number of collisions and percent feedback received versus number of available return channels for a receiver set of R = 100.

encoding packets forwarded to the end-users. Protocol is lightweight, and uses only a single type of feedback information from the end-users for both reliability purposes and dynamic window adjustment. A novel knapsack-based feedback implosion algorithm is implemented to effectively reduce the volume of user feedback carried over the satellite network.

REFERENCES

[1] I. F. Akyildiz, G. Morabito, and S. Palazzo, "Research issues for transport protocols in satellite IP networks," *IEEE Personal Commun.*, vol. 8, no. 3, pp. 44-48, June 2001.

[2] J. Nonnenmacher, E. W. Biersack, and D. Towsley, "Parity-based loss recovery for reliable multicast transmission," *IEEE/ACM Trans. Net.*, vol. 6, no. 4, pp 349-361, Aug. 1998.

[3] S. Bhattacharyya, D. Towsley, and J. Kurose, "The loss multiplicity problem in multicast congestion control," in *Proc. of IEEE INFOCOM*, vol. 2, March 1999, pp. 856-863.

[4] L. Rizzo and L. Vicisano, "A reliable multicast distribution protocol based on software FEC techniques," in *Proc. of HPCS*, pp. 116-125, June 1997.

[5] A. Donner, S. Bovelli, and S. Shabdanov, "Reliable multicast based on DVB-RCS," in *Proc. of AIAA ICSSC*, May 2002.

[6] H. Ernst, S. Shabdanov, A. Donner, and S. Scalise, "Reliable multicast for land mobile satellite channels," in *Proc. of AIAA ICSSC,* April 2003.

[7] K. Manuousakis and J. S. Baras, "Reliable multicasting for flat hierarchy networks based on adaptive air caching," in *Proc. of MILCOM*, Oct. 2003, pp.1295-1299.

[8] D. E. Gossink and J. P. Macker, "Reliable multicast and integrated parity retransmission with channel estimation considerations," in *Proc. of IEEE GLOBECOM*, vol. 6, pp. 3609-3613, 1998.

[9] S. Cho, A. Goulart, I. F. Akyildiz, and N. Jayant, "Adaptive FEC with QoS provisioning for real-time traffic in LEO satellite networks," in *Proc. of IEEE ICC*, June 2001, pp. 2938-2942.

[10] J. W. Byers, M. Luby, M. Mitzenmacher, and A. Rege, "A digital fountain approach to reliable distribution of bulk data", in *Proc. of ACM SIGCOMM,* Sep. 1998.

[11] M. Luby, "LT codes," in *Proc. of IEEE Symposium on Foundations of Computer Science*, November 2002, pp. 271-280.

[12] S. Cho and I. F. Akyildiz, "A poker-game based feedback suppression algorithm for satellite reliable multicast," in *Proc. of IEEE GLOBECOM*, vol. 3, pp. 29030-2934, Nov. 2002.

[13] E. Ichihara, K. Kikuchi, T. Tsuchida, K. Kawazoe, and H. Kazama, "Reliable IP-multicast protocol for bi-directional satellite communication systems," in *Proc. of AIAA ICSSC*, April 2003.

[14] X. Zhou, X. Liu, and J. S. Baras, "TCP over GEO satellite hybrid networks," in *Proc. of IEEE MILCOM*, vol.1, Oct. 2002, pp. 29-34.

[15] European Telecommunication Standards Institute, "Digital Video Broadcasting (DVB): DVB specification for data broadcasting," ETSI EN 301 192, v.1.3.1, 2003.

[16] U.-C Feibig, "A time-series generator modeling rain fading", in *Proc. of Open Symposium on Propagation and Remote Sensing*, URSI Commission F, Garmish-Partenkirchen, 2002, http://www.kn-s.dlr.de/Projects/kaband/kaband_160198.html