Deutscher Luft- und Raumfahrtkongress 2009
DocumentID: 121200

A SYSTEM SIMULATOR FOR ADVANCED DVB-S2/RCS MULTIBEAM SYSTEMS

V. Boussemart, H. Brandt, German Aerospace Center, Oberpfaffenhofen, Germany

Abstract

Satellite communication systems play an increasing role for broadband communication in the customer, as well as in specialized segments. They can be used as gap-fillers for areas not covered and not likely to be covered by broadband DSL, they have their advantages for some kinds of applications (massive multicast) and can help closing the digital divide without having to invest into terrestrial infrastructure in challenging territories. The peculiarities of satellite systems – high initial costs, complexity or impossibility of updates – make it especially important to obtain a correct evaluation of the performance of the system before installing it. It is also a major necessity to be able to invent new algorithms and techniques, to tune system parameters, without having to use a real, expensive satellite system. This paper presents a simulator representing a complete satellite communication system based on DVB-S2 and advanced DVB-RCS techniques, including simulation of atmospheric effects, user behavior and all major protocols needed in transparent as well as regenerative systems. This simulator allows developing new algorithms and protocols, tuning satellite systems and analyzing system stability and performance. The paper also presents an advanced forward link scheduling algorithm as well as some results obtained with the simulator.

1. INTRODUCTION

The emerging role of satellite communication systems for the consumer market is based on several outstanding properties of satellite systems: the large coverage of GEO (Geostationary Earth Orbit) satellite based systems, which allows serving areas which cannot be covered by terrestrial systems with reasonable cost; the inherent broadcast capability; and the high availability even in disastrous situations. On the downside is the high initial cost when exploiting a satellite system and the limited reconfigurability – once a satellite is in orbit its elements cannot be changed, though software updates may be possible. These peculiarities require using techniques that provide as much as possible service over a given satellite and to evaluate and tune the performance of the system in advance as good as possible.

This paper presents a software system that simulates a transparent or regenerative satellite communication system including atmospheric effects and interference, protocols and algorithms, users and applications. The system is based on DVB-S2 (Digital Video Broadcast via Satellite) and advanced DVB-RCS (Return Channel for Satellite) but its modularity allows replacing all protocols and modules so systems based on DOCSIS (Data Over Cable Service Interface Specification) or proprietary protocols could be implemented. The simulator has been used to develop and evaluate new resource management algorithms and to verify the importance of adaptive coding and modulation (ACM) in both the forward and the return link of the system. It has been developed in an ESA project[†].

The paper is organized in four parts: in the first part a short overview of the simulator and the simulated system architecture will be given and some basic explanation of DVB-S2 and DVB-RCS will be given, the second part shows a novel scheduling and encapsulation algorithm, the third part describes the elements of ACM in the forward and return link and the last part provides results on the simulations, brings some conclusions and an outlook on further work.

[†] Resources Management using Adaptive Fade Mitigation Techniques (FMT) in DVB-RCS Multi-Beam Systems; Contract 18826/05/NL/US

2. SYSTEM AND SIMULATOR ARCHITECTURE

The usual configuration of a DVB-S2/RCS system is a star using a transparent satellite payload. Transparent means, that the satellite does little processing on the signal: amplification and changing its frequency band. The resulting signal is then sent back to ground. The main intelligence in the system is located in one or more gateways on ground that contain the connections to the internet and through which all traffic in both directions (from the users and to the users) flows. For simplicity it is usually assumed that only one gateway is used, in reality more than one are normally in the system.

The forward traffic is traffic coming from the internet which is destined for one or more satellite terminals or traffic which was received from a satellite terminal and should be sent to another one. Return link traffic is generated by the terminals and sent to the gateway. Depending on its final destination it is either inserted into the internet connection or sent to another satellite terminal. It must be noted, that in this transparent star configuration communication between two satellite terminals result in a very high round trip time.

In the following two sections an overview over the processing of both the forward and the return link is given.

2.1. DVB-S2 Forward Link

The forward link send path as implemented in the simulator in the gateway is shown FIG 1.

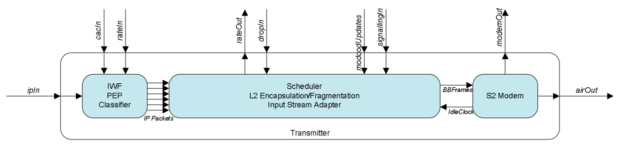

FIG 1. Forward link send path

The incoming traffic (either from the Internet or from other satellite terminals) first enters the IWF (InterWorking Function). This module has several taks:

- Mix the incoming traffic with IP (Internet Protocol) packets from the resource manager in the gateway.
- Packet classification. The simulator implements QoS (Quality Of Service) based on the DiffServ (Differentiated Services) model. The incoming traffic is classified by rules into several QoS classes. The rules as well as the number of QoS classes are configurable. Currently three flows are used: EF (Expedited Forwarding), AF (Assured Forwarding) and BE (Best Effort). The used rules work exclusively on the ToS (Type of Service) field of the IP packets. The classifier inspects all incoming packets and inserts them into per-terminal per-QoS queues.
- Traffic policing. This function is part of the congestion control mechanisms. If it is enabled the rate for each QoS type for each terminal is limited to some amount by means of a token bucket algorithm. This is used by congestion control to stabilize the system in overload situations.
- Multiplexing. All traffic for a given QoS class and a given terminal SLA (Service Level Agreement) class (terminals can come in different types, like SOHO (Small Office/Home Office), SME1 and SME2 (Small/Medium Enterprise) and with different SLAs (which specify, for example, maximum traffic rates per QoS class) is multiplexed into a single output channel. With the standard configuration of 3 QoS classes and 2 SLA classes this results in a total of 6 channels.

The classified output of the IWF enters the scheduler which schedules the packets according to their QoS, inserts signaling packets, and encapsulates the packets according to a configurable protocol and forms so called BBFRAMEs. The scheduler provides the resource management with several measurements for congestion and admission control.

The S2-modem takes the BBFRAMEs, computes their length in time and schedules them onto the output link. It provides the resource management with a measurement of its utilization.

The forward link is organized in BBFRAMEs (BaseBand FRAMES). The structure of these frames is shown in figure FIG 2. It consists of an 80-bit header and a data field which contains the coded data bits. The data is coded using the concatenation of a BCH (Bose, Ray-Chaudhun and Hocquenghem were the inventors of the code) and an LDPC (Low Density Parity Check) code. For transmission the BBFRAMEs get another physical layer header which is BPSK encoded and are modulated with QPSK (Quadrature Phase-Shift Keying), 8PSK, 16APSK (Assymmetric Phase Shift Keying) or 32APSK. Coding rates range from 1/4 to 9/10.

BBFRAMEs (see FIG 2) come in two sizes in terms of bits before modulation: long frames (64800 bits) and, optionally, short frames (16200 bits). With the possible modulations this results in 8 different length in time on the physical link and with the 10 coding values in 19 different length of the maximum data field (short frames don't use 9/10 coding) from 3064 to 58184 bits). Not all combinations of modulation and coding are used because of overlapping critical thresholds – 28 are defined by the standard. For the simulations a subset of 18 has been used (no 32APSK and some almost-overlapping combinations removed).

The modulation and coding combination (ModCod) that is used for a given BBFRAME (and, hence, all containing packets) can be either fixed or variable. The later case is called ACM (adaptive coding and modulation). Here the ModCod is selected based on the measured signal quality that is periodically signaled from the terminals to the gateway.

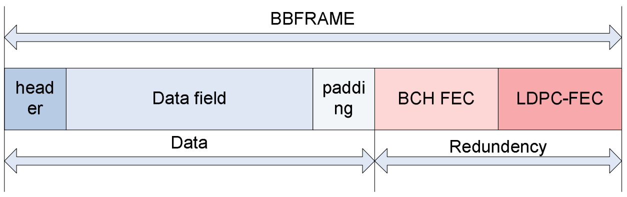

FIG 2. Forward link BBFRAME

Packing IP packets into BBFRAME data fields requires a protocol that is able to fragment IP packets, to pack several IP packets or fragments of them into a single frame and to add data helping terminals extracting their IP packets. The most used protocol is MPE/MPEG (MultiProtocol Encapsulation in MPEG (Motion Pictures Experts Group) cells). This protocol uses fixed size containers of 188 bytes (MPEG cells) which are packed into the BBFRAME (requiring some padding). IP packets are fragment by MPE in fragments that fit into MPEG cells. Because of the large overhead of MPE/MPEG a replacement called ULE (Unidirectional Lightweight Encapsulation) has been standardized by the IETF (Internet Engineering Task Force). Because ULE/MPEG still suffers from problems with MPEG (overhead) an entirely new protocol has been standardized by the DVB-S forum which is called Generic Stream Encapsulation (GSE). GSE uses variable length fragments and minimizes overhead as much as possible.

The receive path in the terminal is simple and is shown in FIG 3.

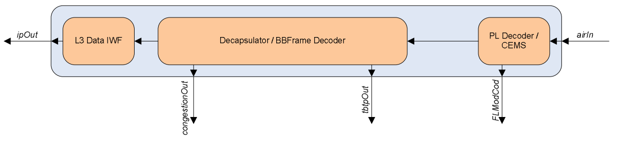

FIG 3. Terminal receive path

The received frames are demodulated and decoded by the receiver. In the simulator this module also applies the transmission delay and decides whether the frame can be correctly received with the actual signal-to-noise-and-interference ratio (SNIR). The decoded frames are forwarded to the layer 2 decapsulator which implements the receiving half of MPE/MPEG, ULE/MPEG and GSE. The decapsulated signaling packets are sent to the terminal resource management and the IP packets to the interworking function, which is just a placeholder in the simulator that outputs the IP packets to the user.

2.2. DVB-RCS Return Link

The current standard for the return link is called DVB-RCS. It features MF-TDMA with optional random access for signaling and burst sizes that are multiple of either MPEG cells or ATM (Asynchronous Transfer Mode) cells. The encapsulation protocol is either MPE/MPEG or AAL5 (ATM Adaptation Layer 5) depending on the used bursts.

The code is either a Turbo code or a concatenation of a convolutional and a Reed-Solomon code. It must be noted, that while there are different code rates, the used rate is fixed in a given system. The modulation is QPSK and cannot be changed.

Allocation of resources is done in the gateway by the return link resource manager. Terminals send requests on a signaling channel to the gateway that are computed based on the traffic in the given terminal. It is also possible to used fixed, login-time allocations although this is not very efficient already for medium numbers of terminals. There are several options for signaling.

Usually a segmented organization of the return link is used. In this case the entire return link is segmented into so-called superframes which represent a portion of time and frequency on the return link (FIG 4).

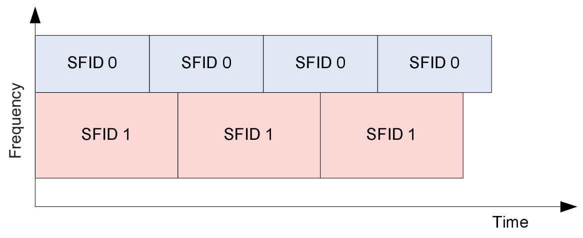

FIG 4. Superframes in the return link

Superframes are identified by their ID and contain a set of carriers (with corresponding associated allowed symbolrates) over a fixed time span (up to several 100 milliseconds). The superframes are consecutive in time and the resource control may select superframes from a predefined set depending on the current traffic situations (in the above example the third superframe with ID 1 could be followed by superframes with another ID).

Superframes are composed of frames much like the link is composed of superframes (FIG 5). Frames in a single superframe may have different durations, bandwidth, frequency and number of timeslots. Frames are organized in timeslots which are the basic allocation unit (FIG 6).

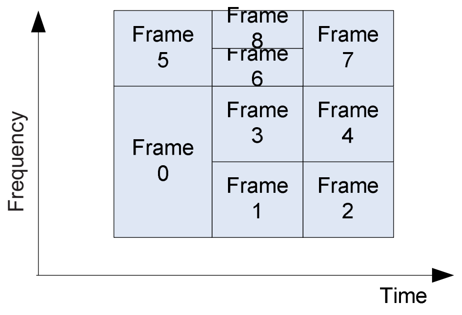

FIG 5. Frames in a superframe

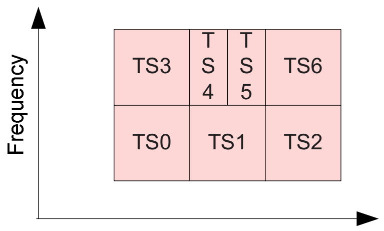

FIG 6. Timeslots in a frame

The organization of the return link is periodically broadcast by the gateway to all terminals in the form of different tables: superframe composition table (SCT), frame composition table (FCT) and time slot composition table (TCT). The actual allocation to the terminals is transmitted in the terminal burst time plan (TBTP) which is sent for each new superframe.

While superframes, frames and timeslots allow for very flexible allocation and management of the return link, this flexibility is normally not used, because it requires a very intelligent resource allocator. The resource allocation is run in real-time and must be fast and scalable to thousands of terminals. Therefore often a simple configuration is used (single superframe ID, all frames the same length as the superframe). The simulator allows multiple superframes with equal sizes which are dynamically chosen, but requires frames to be the same length as the superframe. This results in an organization as in FIG 7.

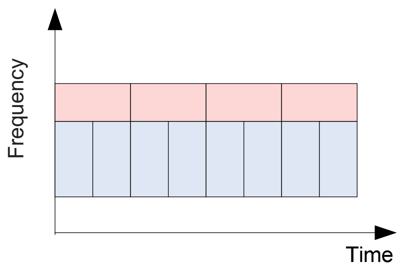

FIG 7. Simulator superframe example

Here the superframe contains two frames with a bandwidth ratio of 1:2. Because all timeslots are the same size (in terms of symbols) one frame contains 4 and the other 8 timeslots. Actual numbers are of course much higher (several thousand timeslots per carrier).

2.2.1. Advanced DVB-RCS

One of the goals for the simulator was the evaluation of an enhanced RCS. Two main features are required:

- Support for ACM (adaptive coding and modulation) as in the forward link.
- Support for fixed burst length segmentation (FBL) as opposed to variable burst length (VBL, or fixed information length (FIL)).

To implement ACM the gateway measures the quality of the received bursts from all terminals and sends these reports to the resource manager. Here a ModCod will be selected and taken into account when allocating the bursts. Unfortunately this adds another dimension of complexity to the resource allocation. The problem is that the timeslots have different length on the physical layer depending on the coding and the modulation when the information size in them is fixed (to an MPEG or an ATM cell, for example). This means that designing the superframe also requires guessing to correct ModCod distribution for the given terminal type and symbol rate that gives maximum performance. The ModCod is fixed during this process and the resource allocator must later find slots matching not only the symbolrate of the terminal but also the required minimum ModCod. Running with the wrong ModCod distribution means that the resource controller may not be able to allocate all requested slots because all remaining free slots during an allocation round have a ModCod that requires a better SNIR than the terminal has.

This problem can partially be overcome by defining several superframes with different ModCod distributions and let the resource controller chose the best one in each superframe allocation round based on the requests and the terminal signal conditions. This is called semi-dynamic allocation. Unfortunately this technique requires two rounds over the requests – first to choose the superframe to use and second to actually allocate bursts. Even with this technique allocation is always sub-optimal.

This problem can entirely be overcome by drastically changing the return link segmentation. Instead of using bursts that fit the same number of data bits for all ModCods one can use bursts that have always the same number of symbols and, hence, length in time. Actually it is useful to have a small set of possible burst length. While the simulator allows full flexibility here, all simulations have been done with burst of length 400 and 800 symbols. The length on the physical layer still depends on the symbol rate, so that the resulting superframe definition looks again as in FIG 7 when only the short bursts are used. Long bursts are automatically used by the terminal if it gets two consecutive short slots.

The win is obvious: the resource controller has full freedom to assign ModCods to the slots – each slot position can accommodate each ModCod. This allows almost optimal allocation (limited by the used algorithm).

The backside of this FBL scheme is that MPE/MPEG and AAL5/ATM are no longer usable. Instead the simulator used the GSE protocol in this case which turns out to have even less overhead than its counterparts.

The send path of the return link which is located in the terminal is shown in FIG 8

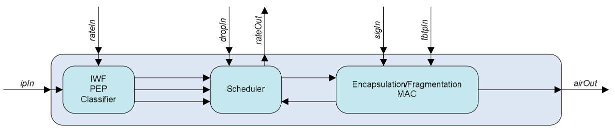

FIG 8. Return link send path

The incoming IP packets are received by the IWF module which multiplexes them with IP packets from the resource management of the terminal. This IP path is used for call admission control. The IWF classifies the packets into the QoS classes and optionally applies policing to each class with a token bucket. The three resulting flows enter the scheduler where they are scheduled according to their QoS requirements. The scheduler also does rate and volume measurements on the traffic so that the local resource manager can generate resource requests. It also allows dedicated dropping of traffic for congestion control. The resulting packet stream enters the layer 2 processing where it gets fragmented, multiplexed with signaling messages and encapsulated. The resulting bursts are sent to the satellite.

In the receive path in the gateway (FIG 9) a multiburst demodulator receives all the bursts of all carriers.

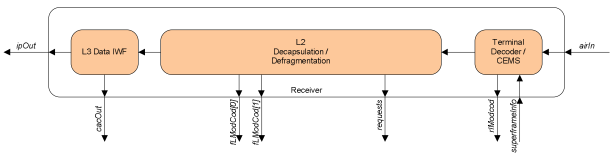

FIG 9. Return link receive path

In the simulator this module also applies the transmission delay and checks the received bursts whether the current signal quality is strong enough to demodulate and decode the burst. The measurement is also sent to the resource manager so that it can select the right ModCod for the given terminal. The decoded bursts are handed over to the layer 2 module where the encapsulation protocol reassembles the IP and signaling packets. Signaling packets are sent to the resource manager while IP packets enter the IWF where IP packets for the resource manager (for call admission control) are filtered out. The remaining IP packets are sent to the Internet.

2.3. Simulator Architecture

The complete simulator for the transparent case consists of one or more gateway modules (FIG 10), up to several thousand satellite terminals, Internet terminals, forward and return link transponders and a big router that represents the Internet. The satellite terminals consist of the transceiver (FIG 11) and a set of application modules, modeling applications like web browsing, VoIP (Voice over IP), video/audio streaming, mail and video conferencing. The internet terminals consist just of the traffic modules. The forward link transponders implement the broadcasting – they duplicate the forward link frames to all terminals, while the return link transponders are just multiplexers to get the return link bursts from all terminals into the gateway.

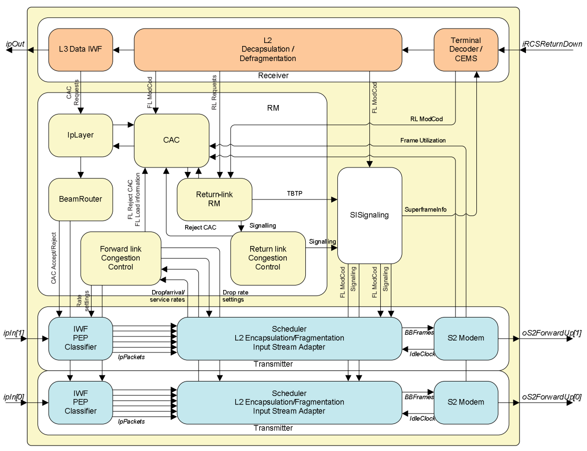

FIG 10. Gateway simulation module for two forward link carriers

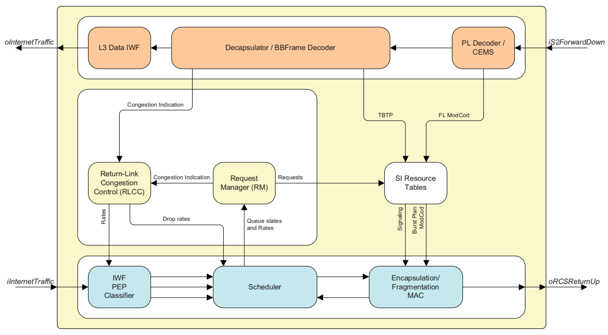

FIG 11. Satellite terminal transceiver module

The simulator is written in C++ and uses the OMNeT++ [1] simulation framework. A preprocessing step implemented in Matlab generates SNIR (signal to noise and interference ratio) timeseries for forward and return links of all terminals, ModCod threshold tables and SNIR estimation tables. A postprocessing step is also implemented in Matlab. It reads the logfiles from the simulator and produces graphs and numbers. It is written in a toolbox-like approach so that new graphs and results can be produced easily.

3. FORWARD LINK SCHEDULING

The forward link scheduling process for a DVB-S2 system is complicated by the fact that not only QoS requirements are to be taken into the account, but also the individual current ModCod of the destination terminal plays an essential role. The problem is, that forward link frames (BBFRAMEs) are rather larger so that usually fragments from more than one IP packet will fit into a single frame. The ModCod of the frame must be selected so that all terminals for which there are IP packets in the frame can receive the frame. This means that the most robust ModCod of the contained packets must be chosen. This, in turn, means that some packets will be sent with a ModCod that is more robust than required and that the efficiency is worse than it could be. The task of the scheduler is to avoid these situations as much as possible. So informally stated the optimization tasks of the scheduler are:

– Keep the minimum QoS requirements (delay, loss, jitter) and try to do better than the minimum.
– Never drop signaling packets because this may destabilize the system.
– Send each IP packet at the most efficient ModCod for the destination terminal's current conditions.
– Optimize the packing on the fragmentation layer and prevent padding as much as possible.
– Fill the link as much as possible if there is enough traffic.

The second point is actually easy – it just requires giving the signaling traffic the highest absolute priority and sending them at the most robust ModCod.

The last two points require intimate interworking between the scheduler and the encapsulator to the extent that both are a single integrated module. The first and third point require a departure from the traditionally queuing model for schedulers where the scheduling process can only see the first packet in one or more queues – it is beneficial for the scheduler to be able to see all available packets. This will also help for the forth point.

3.1. Pool scheduler with ModCod optimization

Because of the described complexity a new scheduler has bee proposed and implemented in the simulator (described in detail in [2][3] and [4]). In addition the simulator contains an additional optional optimization step. The structure of the scheduler is shown in FIG 12.

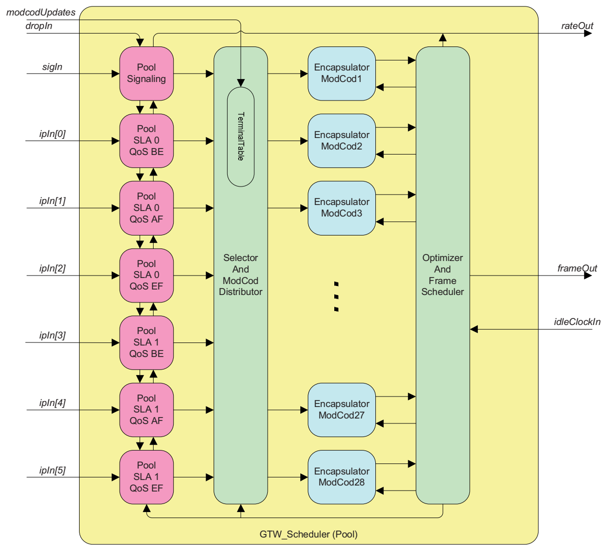

FIG 12. Forward link pool scheduler

In this example three different QoS classes are assumed (EF, AF and BE) and two different terminal SLA classes (gold and silver). As the already classified packets enter the scheduler they are stored in so-called pools. These pools differ from queues in that all the packets in the pool are accessible and selectable for scheduling. Additionally all packets are tagged with a label which contains: (1) a link to the entry into the terminal database for fast retrieval of the currently required minimum ModCod of the destination terminal and (2) a field for storing an arbitrary floating point value the use of which depends on the selection algorithm applied to the pool.

There are three different selection algorithms available: time-based, all-selection and fill-selection. A selection algorithm is tied to a pool at pool creation time and handles both – insertion of packets (so it can set the label field) and retrieval.

The time-based selector sets the floating point field in the packet's label to the arrival time of the packet. When selecting packets it selects those packets that are in the queue for longer than a certain threshold time. Also it can drop packets that are in the queue for too long. This selector is used for expedited forwarding QoS classes.

The all-selection selector always selects all packets in the queue and doesn't use the floating point label field. It is used for the signaling pool and the assured forwarding pool.

The fill-selection selector selects packets while there is still room in the current output frame(s) and stops when they are full. The difference between this and the all-selector is that the latter can create new output frames while the former just fills what frames are already there.

The encapsulation protocol is handled in encapsulator modules – one for each ModCod. At the output of the encapsulators an optimizer implements the optional ModCod optimization.

The scheduler works on a periodic basis which is selected based on the desired maximum jitter – the typical period is 20ms. Each round consists of four steps the third of which is optional:

1) Select packets from the packet pools by calling the selectors in a fixed-priority order. For the selected packets the ModCod is determined by looking up the terminal in the database and the packet is handed over to the encapsulator for this ModCod.
2) Call the selectors in the same order, but this time always select all packets in the given pool.
3) Optimize the resulting partial frames.
4) Flush the remaining frames to the modem.

In steps (1) and (2) the scheduler ensures that the total time the generated BBFRAMEs will use on the link does not exceed the scheduling period. This is a hard stop criterion for all selectors. As soon as a frame is full it gets flushed to the modem immediately (there is a mechanism available to keep BBFRAMEs adjacent which may be required for some encapsulation protocols).

The encapsulators receive the IP packets and do the necessary fragmentation, prepending of headers, appending of trailers as required by the configured protocol. They instantiate new BBFRAMEs as necessary and track the total amount of time used by all BBFRAMEs during the current scheduling round so far.

When the second scheduling step is finished the encapsulators will normally contain partially filled frames. Before sending them to the modem an additionally optimization step may be applied. This step uses the fact that the data needs not to be sent at the most effective ModCod for the given terminal conditions but can also be sent at a more robust ModCod. The algorithm tries to move the data from partial BBFRAMEs with a less robust ModCod to partially filled BBFRAMEs with more robust ModCods so that some of the BBFRAMEs get empty and need no to be sent.

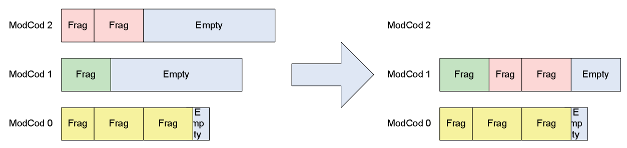

FIG 13. ModCod optimization

In the example in FIG 13 three partial frames remain at the end of scheduling step 2. The algorithm tries to move the fragments from the frame with ModCod 2 (the most efficient and last robust one) into the frames with ModCods 0 or 1. In ModCod 0 there is not enough space, but in ModCod 1 there is. So the fragments are moved there and the ModCod 2 frame can be deleted.

Because the frames produced in a scheduling round do normally not add up to 20ms there is an additional flow control loop between the modem and the scheduler. When the modem gets idle (when it is transmitting the last frame in its buffer) it sends a signal to the scheduler to start a new scheduling round. The signal is produced a configurable amount of time before the modem actually runs idle to give the scheduler a chance to come up with a new frame during this time and keep the modem busy. Under normal operating conditions the scheduler will be driven by this modem flow control. Only when there is very low traffic the scheduler will actually operate in 20ms intervals.

3.2. Simulation methodology

In order to evaluate the pool scheduler a second scheduler has been implemented as a reference point. The structure of this reference scheduler is shown in FIG 14.

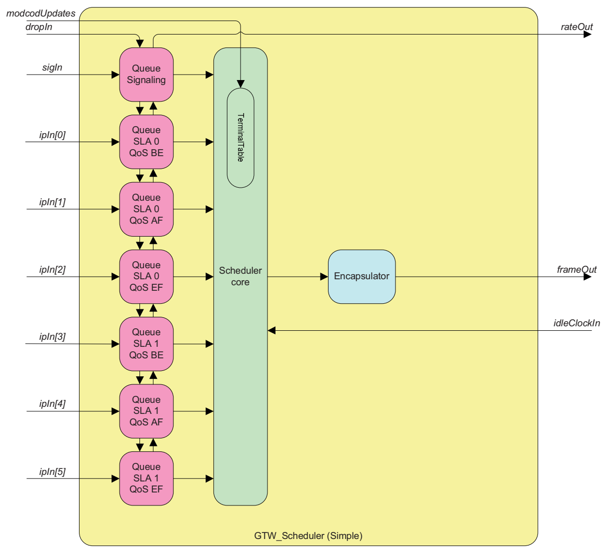

FIG 14. Reference scheduler

The incoming packets are stored in classical queues which are accessible only at the head. The scheduler uses a fixed priority scheme to select the next packet and hands it over to the single encapsulator. If the encapsulator currently has no BBFRAME it creates one and puts the packet into it. If it already has a frame and if the required ModCod of the selected packet is higher (less robust) than the ModCod of the BBFRAME, the packet is put into the frame (here efficiency is lost). If the required ModCod of the packet is more robust than that of the BBFRAME either the frame is changed to a more robust ModCod (this may fail because less data fits in a frame with a more robust ModCod and not all already inserted data may fit) or the frame is flushed to the modem (here again efficiency may be lost because of padding) and a new frame is created. The scheduler is driven by two event sources: the flow control messages from the modem, telling that it soon will be idle, and an encapsulation timeout that prevents that partial frames are held for a long time when there is very low traffic.

Because the most interesting operating point for a scheduler is near overload the number of terminals and users in the systems has been setup so that the traffic on the forward link is around the maximum possible traffic. In fact there are three scenarios low load (90%), medium load (95%) and high load (105%). For simulations with ACM enabled there is a extreme load scenario (40% more terminals for the forward link simulations and 100% more terminals for the return link simulations) that is needed to overload the link in this case.

4. ADAPTIVE CODING AND MODULATION

The link budget of satellite communication system is usually computed by assuming a certain availability (for example 99.99%) and taking into account several limitations (power in the terminal and the satellite, dish size, maximum allowed interference and so on). This results in a certain bandwidth that can be transferred. Because of the high assumed availability this results in a large margin that most of the time is not needed, because most of the time whether conditions are good (rain fading is the prevailing variable influence on the channel in Ka/Ku band). The idea of ACM is to use this link margin when the link is good by switching to higher-order modulations and/or higher coding rates.

ACM requires a feedback loop: the receiver measures the signal quality (SNIR) and sends this back to the sender which changes the ModCod it uses so that the actual SNIR is always larger than the critical SNIR of the ModCod. This decision process needs to take into account the delay it takes from making the measurement to the use of the measurement and the maximum change rate of the SNIR. This results in some margin that is applied to the critical thresholds. These margins are selected in a way that a hysteresis results which prevents extensively fast switching between ModCods (FIG 15).

Here the solid lines are the critical thresholds. They represent the minimum SNIR that is needed in order that the frame or burst can be demodulated and decoded taking into account the characteristics of the coding scheme and rate. The dashed lines are the up thresholds. When the system currently uses a given ModCod for a given terminal and a measurement arrives which is above the up threshold, the system switches to the next higher ModCod. This up threshold is obviously always higher than the critical threshold of the higher ModCod. The dotted lines are the down thresholds of a ModCod. If the SNIR falls below the down threshold the system switches to a lesser ModCod. The thresholds are chosen so that there is a hysteresis – the up threshold of a given ModCod is always larger than the down threshold of the next higher ModCod.

The difference between the down threshold and the critical threshold forms a safety margin.

In the return link this safety margin may be optimized by making it dependent on the age of the measurement (given that no new measurement arrives). This is possible because in the return link measurements can be taken only when the terminal actually sends a burst (the forward link is always active and can be measured all the time). The older the measurement gets that led to the current ModCod the higher gets the margin because the uncertainty about the current channel state gets larger. When the margin finally gets higher than that old measurement the system switches down to the next ModCod. If the age of the measurements gets to old a switch to the most robust ModCod occurs. If the margins are non-adaptive they must be just large enough the accommodate the oldest possible measurement.

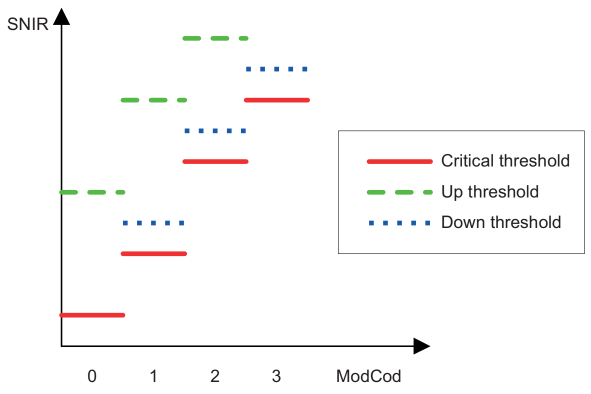

FIG 15. ModCod thresholds

For the DVB-S2 forward link the measurements are done in the terminal, but are needed in the gateway scheduler. Therefore it is necessary to signal these measurements on the return channel. This is done using the same signal mechanism that is used for the resource requests sent to the gateway by the terminal.

In the return link the measurements are done in the receiver in the gateway and they are needed in the return link resource controller in the gateway. Therefore only local transmission of these measurements in the gateway is necessary. The terminals get the ModCod to use in the TBTP that is broadcasted to all terminals for every superframe.

5. RESULTS AND CONCLUSIONS

This chapter provides some results obtained with the simulator. There is a very large number of results that may be obtained so we will focus on two aspects: the performance of the new scheduler and ACM in the forward link and the performance of ACM and the FBL scheme in the return link.

5.1. Forward link scheduler and ACM

The efficiency of the forward link scheduler can be measured in different ways. An ideal scheduler should fulfill the QoS requirements up to a load of 100% without dropping packets. In overload situations it should remain stable, which means it should fill the link to 100% and drop only packets with low priority. Even in this situation QoS requirements should be kept as good as possible.

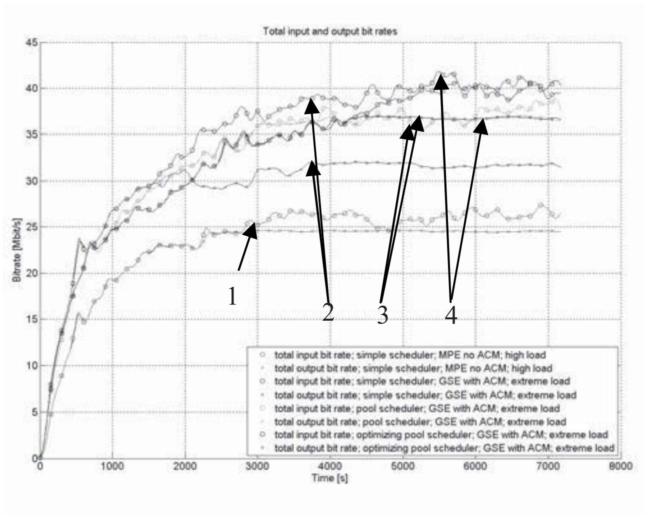

FIG 16. Forward link performance

FIG 16 shows one of the possible measurements. It contains the results of four simulation runs and shows several aspects of the scheduling system. The simulations were setup as follows:

1) A simulation with high load (105% of theoretical forward link capacity) with the reference scheduler, MPE/MPEG as encapsulation protocol and no ACM (marked ‘1’).
2) A simulation with extreme load (40% more terminals than high load) with the reference scheduler, GSE as encapsulation protocol and ACM enabled (marked ‘2’).
3) A simulation with extreme load, the pool scheduler without optimization, GSE as encapsulation protocol and ACM enabled (marked ‘3’).
4) A simulation with extreme load, the pool scheduler with optimization, GSE as encapsulation protocol and ACM enabled.

For each simulation run two lines are shown: the total scheduler input data rate in bits/s (upper line) and the scheduler output data rate in bits/s (lower line). The input data rate starts from 0 and goes up because the traffic models are warming up at the beginning and reach a stable point after approx. 1.5 hours. During the warm up both curves for all scenarios are lying on each other meaning that the link load is below 100% and the output rate equals the input rate. At a certain point the output rate flattens because the link utilization reaches 100% and the difference between the input and the output curve is the loss.

It can be seen that switching to a better encapsulation protocol (GSE instead of MPE/MPEG) and using ACM alone gives 28% more capacity with exactly the same link (difference between output of scenario 1 and 2). Now when switching from the reference scheduler to the pool scheduler there is an additional gain of 13.5%. So the total gain in capacity from using GSE instead of MPE/MPEG, enabling ACM and having a better scheduler is 48%.

A good measure for the efficiency of the entire scheduling and encapsulation system is the IP efficiency. This is the number of IP (that is, useful) bits transmitted per physical layer symbol. Measured in the stable state of the applications it is 1.19 bits/symbol for MPE/MPEG, no ACM, simple scheduler and 1.70 for GSE, ACM and the pool scheduler. This is a win of 42.8%. The IP efficiency takes into account the efficiency of the encapsulation protocol itself, the efficiency of the scheduler and the physical layer efficiency which is determined by the coding rate and the modulation order.

The following two figures show how the pool scheduler reaches the higher efficiency when compared to the reference scheduler.

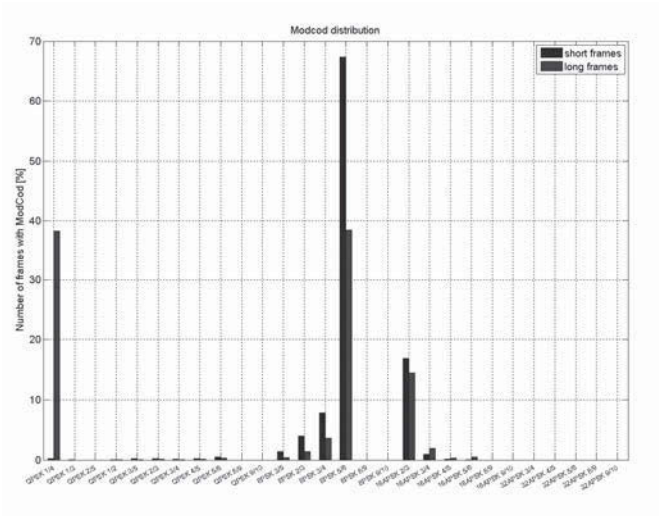

FIG 17. Forward link ModCod distribution (reference scheduler).

FIG 17 shows the ModCod distribution when using the reference scheduler (note, that the 32APSK ModCods and 8/9 and 9/10 coding rates are disabled). For each ModCod two bars are shown: the left is the percentage of short frames and the right the percentage of long frames. The absolute peak in ModCod usage is at 8PSK and 5/6 with some usage (around 17%) of 16APSK rate 2/3. The ModCod 0 frames carry signaling information.

The figure changes when the pool scheduler is used (both simulation use exactly the same channel conditions) as shown in FIG 18.

There is a big shift to more efficient ModCods. Now the peak is 16APSK rate 2/3, over 20% 16APSK rate 3/4 and some use of 16APSK rate 4/5 and 5/6.

The reason for this shift is that the reference scheduler often needs to put IP packets that could be sent at a more efficient ModCod into a BBFRAME that has a less efficient ModCod just because there is also a packet to a terminal in worse conditions (ModCod downgrading). The pool scheduler will put these packets into different BBFRAMES, each with the optimum ModCod. It downgrades packets only in the optimization step and there it makes the efficiency larger, because the packet will be transmitted instead of padding which would be even worse from the efficiency point of view and one BBFRAME needs not to be sent altogether.

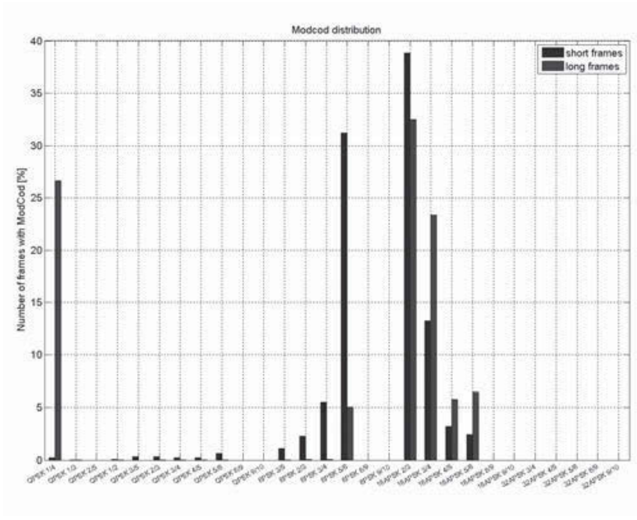

FIG 18. Forward link ModCod distribution (pool scheduler).

Another interesting measure is the packet delay jitter. Especially for real-time applications like VoIP the jitter should be as low as possible because a larger jitter means that a larger playout buffer is needed on the receiver leading to more delay. FIG 19 shows the delay jitter in the forward link for the VoIP, the audio/video stream (avstream) and the video conferencing (vconf) application measured as the absolute difference between the expected and the real arrival time of a packet. It must be noted, that VoIP traffic is classified into the EF class while avstream and vconf is transferred as BE traffic.

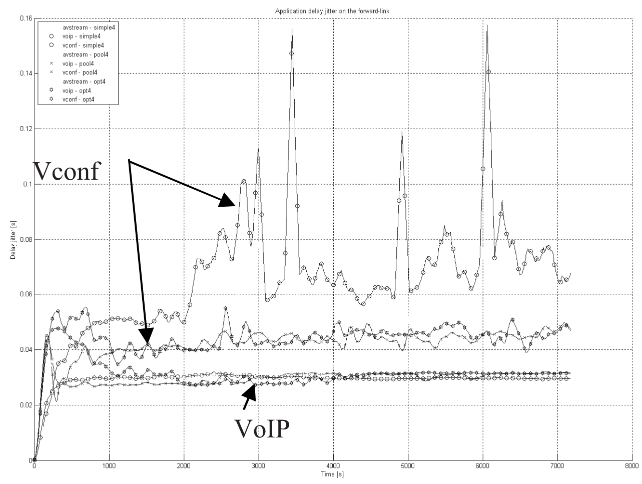

FIG 19. Delay jitter in the forward link

For VoIP it can be seen that there is almost no difference between the reference scheduler and the pool scheduler even in overload situations (the simulation was done with GSE, ACM enabled and the extreme load) – the reference scheduler is even slightly better (35ms instead of 36ms). For the video conference the situation is entirely different, though. With the reference scheduler the jitter is always larger than 60ms having peaks going up to 160ms (upper of the three lines for vconf). These peaks are related to extreme bursts in total traffic. With the pool scheduler the situation is more stable – the jitter is always kept between 40ms and 50ms. It is less and more stable.

5.2. Return link ACM

In the return link the gain from using advanced technologies is even more impressive than in the forward link.

FIG 20 shows the terminal scheduler cumulative (over all terminals) input and output rates for three scenarios:

1) MPEG sized bursts, MPE encapsulation and no ACM with high load (105% of the link capacity).
2) MPEG sized bursts, GSE encapsulation and no ACM with high load.
3) Fixed sized bursts (400 and 800 symbols), GSE encapsulation and ACM. Twice the number of terminals as in scenarios 1 and 2.

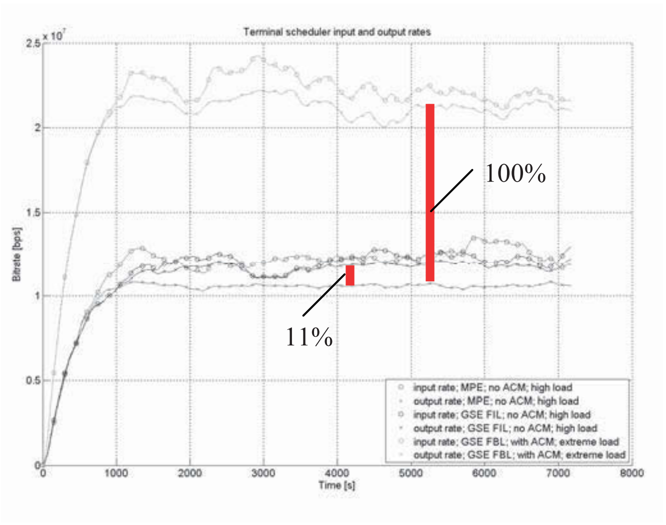

FIG 20. Return link performance

The lowest straight line is the output rate for case 1 – the system is obviously in overload. Just by replacing the encapsulation protocol with a more efficient one, the output rate goes up by around 11% and there is almost no loss anymore.

With switching to fixed-burst-length segmentation on the link, enabling ACM and exploiting the full dynamic burst allocation a gain of 100% in output rate is reached.

A very interesting performance figure is the delay in the scheduler which is the biggest contributor to the delay jitter in the return link.

In FIG 21 the upper line is the delay (measured as the time it takes a packet to travel through the scheduler) for scenario 1. It jitters between 0.4s and 1.3s which means that the total delay on the return link may be more than 1.5s with a very large jitter that gives additional delay due to the large required playout buffer. By going to a more efficient encapsulation (GSE) the delay is reduced 0.2 and 0.7s but this is due to the fact the the overload is reduced. With the same overlead the delay would also be the same as for MPLE.

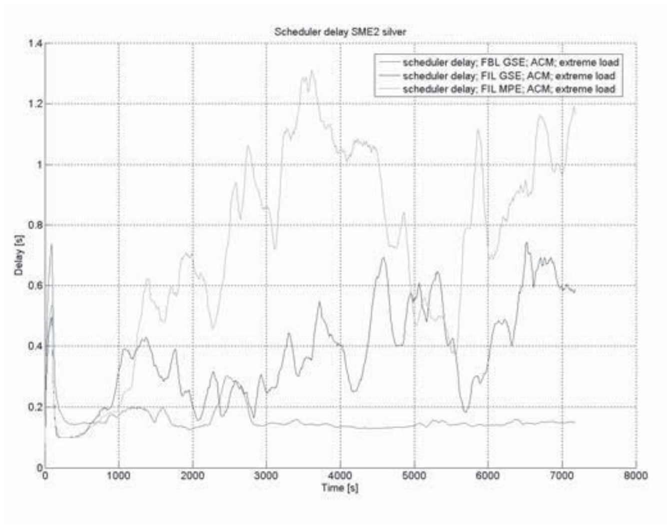

FIG 21. Return link scheduler delay

The situation, however, drastically changes by switching the FBL and ACM. Even though the overload is the same as in the first scenario (albeit with twice the number of terminals) the delay is constant and well below 0.2s (lowest line). This is due to the fact that the scheduler has much more freedom when allocating bursts to terminals with different channel conditions.

5.3. Conclusions

A modular simulator has been developed that can be used for performance evaluation and tuning of satellite communication systems based on the DVB-S family of standards. The simulator allows also developing and evaluating new protocols and algorithms.

The simulator has been used to show the benefits from using ACM and advanced algorithms and protocols in such a system. The total gain in capacity in the forward link from using DVB-S2 with enhanced schedulers and encapsulation protocols compared to a DVB-S-like system is around 48%.

The benefit of using better suited return link segmentation, a more efficient encapsulation protocol and ACM is 100% - the available bandwidth is doubled. At the same time the delay jitter is reduced and stabilized even for overload situations.

5.4. Further work

Several topics are currently in the works which enhance both the simulator and the simulated communication systems:

– Terminal login and logout. This allows varying the load during a simulation run. Currently several runs are necessary to simulate different loads.
– A MySQL backend for the results produced by the simulator. This allows more flexibility in the postprocessing and also would allow displaying results while the simulator is still running.
– A TCP implementation for the applications (currently only IP is fully implemented).

System related enhancements:

– Better schedulers for both the forward and the return link that integrate the policing function with the scheduler and provide early overload detection.
– A new return link encapsulation protocol based on GSE but with less overhead.
– A better return link resource manager which can allocate very low rates (one burst per N superframes).
– TCP-awareness of the system.

References

[1] www.omnetpp.org
[2] C. Párraga Niebla, “Scheduling Techniques for Satellite Systems with Adaptive Coding and Modulation”, in Proc. of ASMS Conference, Herrsching, Germany, 29 – 31 May 2006.
[3] R. Toegl, C.Párraga Niebla, U. Birnbacher, “Framing Efficiency Optimization for DVB-S2 Systems with QoS Guarantees,” in Proc. of 12th Ka and Broadband Communications Conference, Naples, Italy, September 27 – 29, 2006.
[4] R. Toegl, C. Párraga Niebla, U. Birnbacher, “Framing Efficiency Optimization for DVB-S2 Systems,” in Proc. of IEEE Globecom 2006, San Francisco, California, USA, 27 November – 1 December 2006