Journal of Zhejiang University SCIENCE A
ISSN 1009-3095 (Print); ISSN 1862-1775 (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

Efficient video-on-demand services over mobile datacast channels

JENKAČ Hrvoje[†1], STOCKHAMMER Thomas[†2], XU Wen[3], ABDEL SAMAD Wissam[3]
([1]*Institute for Communications Engineering, Munich University of Technology, Munich 80333, Germany*)
([2]*Nomor Research, Bergen 83346, Germany*)
([3]*BenQ Mobile, Munich 81675, Germany*)
[†]E-mail: h@tum.de; stockhammer@nomor.de
Received Dec. 1, 2005; revision accepted Feb. 28, 2006

Abstract: The integration of reliable Video-on-Demand (VoD) broadcasting schemes in the DVB-h transmission system is studied, exemplary for Pyramid Broadcasting (PB). Sophisticated VoD broadcasting schemes such as PB allow receivers to tune into the ongoing transmission of a video-stream at arbitrary time, while still being able to receive the multimedia sequence from the beginning to end, after a short initial playout latency. Raptor coding, integrated in the FLUTE protocol, is combined with the traditional PB scheme in order to provide high service and presentation reliability. We give a short overview on the DVB-h transmission system as well as on the FLUTE protocol. We present and discuss options for the integration of VoD broadcasting schemes in combination with Raptor coding. We achieve backward-compatibility even with terminals not supporting Raptor coding. Simulation results show the benefits of the discussed VoD scheme compared to existing carousel approaches in DVB-h.

Key words: Video-on-Demand (VoD), DVB-h, Pyramid broadcasting, FLUTE protocol, Raptor code
doi:10.1631/jzus.2006.A0873 **Document code:** A **CLC number:** TN919.8

INTRODUCTION

Mobile TV and wireless streaming services to mobile users are recently experiencing rejuvenation due to advanced terminal capabilities, new network infrastructure and significant improvements in media coding efficiency. Moreover, to deliver such services efficiently to a multitude of mobile terminals in parallel, point-to-Multipoint (p-t-M) transmission schemes are favored and have been adopted in recent standardization efforts. For example, 3GPP has recently enhanced the existing mobile cellular systems EGPRS and UMTS by introducing additional p-t-M transmission bearers by creating Multimedia Broadcast/Multicast Services (MBMS). Furthermore, the terrestrial Digital Video Broadcasting (DVB) system DVB-T has been enhanced to meet the requirements of moving receivers and the DVB-h (Digital Video Broadcasting for Handheld) standard was released.

Whereas DVB-T, DVB-S and DVB-C (Terrestrial, Satellite, Cable) mainly aim at Digital TeleVision (DTV) broadcast over MPEG-2 transport streams, the DVB-h system is IP-based. Furthermore, DVB-h not only enables DTV services, but also IP-Datacasting services by the definition of appropriate content delivery protocols (CDP).

Traditional video broadcast services such as plain TV services allow that all receivers retrieve an identical media stream synchronously. Thereby, arbitrary tune-in of receivers into the ongoing program is commonly supported by random access points in the encoded media streams. However, despite the capability to almost instantaneously join an ongoing program, this does not prevent the miss of a significant amount of earlier content, e.g., first part of a movie. For mobile users the arbitrary access of content is even more desirable. Video-on-Demand (VoD) services aim at providing the possibility for receivers to tune-in (request the video sequence) at arbitrary times without missing the start of the program. To accomplish this feature, trivial schemes offer the service over conventional point-to-point (p-t-p) con-

nections, e.g., as provided by GPRS or UMTS packet-switched data channels. However, with increasing number of subscribers, the system may not scale appropriately. Then, the service will use a significant amount of transmission bandwidth. To eliminate the drawbacks of multiple p-t-p connections a huge variety of VoD broadcasting schemes have been proposed and evaluated (Hu, 2001; Engebretsen and Sudan, 2002; Xu, 2001). All of them have in common that media can be "requested" asynchronously by receivers without the necessity for individual p-t-p connections. Usually, these schemes are based on some media stream pre-processing. The pre-processing is in general performed such that only some acceptable bandwidth expansion is achieved, independent of the number of receivers, and that the worst-case start-up latency is still in an acceptable range.

The most prominent VoD broadcasting schemes are known as Harmonic Broadcasting (HB) (Engebretsen and Sudan, 2002) and Pyramid Broadcasting (PB) (Hu, 2001). These schemes have been studied in detail and have recently been extended to operate in packet loss and noisy environments (Horn *et al*., 2001; Jenkač and Stockhammer, 2005; Huang *et al*., 2004). However, in these prior works only conceptual analysis had been performed. Protocol issues and the proper integration of the VoD broadcasting schemes into transmission systems are rarely addressed in these papers. In this work we consider practical aspects for the integration of PB in conjunction with Raptor coding (Shokrollahi, 2003; Luby *et al*., 2006) into the DVB-h datacasting framework.

VIDEO-ON-DEMAND OVER BROADCAST CHANNELS

Problem formulation

On-demand services should provide specific user experience. Whenever a user decides to join the service, it expects that the content is played from the beginning with as low startup latency as possible. In p-t-p connections this is easily realized by issuing a request to streaming server which then starts a streaming session to the requesting user. However, with increasing number of users, unicast delivery is very inefficient and broadcast solutions are preferable.

In this case the users cannot explicitly issue a request to a streaming server, but the broadcast server distributes content without receiving any explicit requests. The time period between tuning into the broadcast and the time when the playout is started at the receiver, is usually referred to as initial playout latency. To formalize the concept of asynchronous media reception in broadcast environments, we assume that receivers tune into an ongoing broadcast distribution without any coordination between transmitter and receivers, and, without any coordination among receivers, respectively. Especially in wireless environments, where user mobility is one of the key aspects, receivers will enter a broadcast area asynchronously, i.e., at arbitrary time instants. So in such hostile transmission environments with costly resources, the provision of high-quality on-demand services poses some challenges to be solved.

Trivial solutions

An obvious solution to provide video delivery with the option of asynchronous access is simply carouseling the entire video stream, i.e., the transmission of the video sequence restarts after it has been completed. The Content Delivery Protocol (CDP) specification of DVB-h (ETSI, 2005), already foresees carousel services. The upper part of Fig.1 visualizes the concept of a data carousel. However, in the worst case, the corresponding receiver just missed the first data unit, so that in the worst case, the initial latency Δ can be as large as $\Delta=T$, with T denoting the duration of the media stream (For simplicity, we assume that the transmission rate and the media bit-rate are equal).

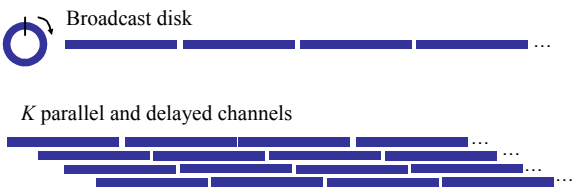

Fig.1 Trivial solution: data carousel

If such a long waiting time is not acceptable, the transmission bandwidth must be expanded by some factor ω, e.g., by broadcasting the carousel with a bit-rate R_c higher than the corresponding bit-rate of the media-sequence R_m. The bandwidth expansion in this

case is $\omega=R_c/R_m$, but the worst-case playout delay is reduced to $(R_m/R_c)\cdot T$. Alternatively, one would broadcast the same media stream on K parallel channels, as shown in the lower part of Fig.1. On each channel a delayed version of the video stream is offered periodically with a delay shift of T/K. The worst-case playout latency is reduced to $\Delta=T/K$, but the overall required bandwidth increases by a factor of $\omega=K$. Due to the significant bandwidth expansion, both trivial schemes are rather unattractive for service providers.

Pyramid broadcasting

More sophisticated broadcast schemes (Hu, 2001; Engebretsen and Sudan, 2002) have in common that the media bit-stream is segmented into S segments $\mathcal{G}_i$, with each segment being periodically broadcast on different parallel channels, with i=1, …, S. Each segment is assigned a segment size $|\mathcal{G}_i|$, representing the size of the segment in bytes, as well as a latest decoding time (for the segment) $t_{DTS}(\mathcal{G}_i)$ representing the earliest decoding time of any contained media unit in this segment.

The most prominent schemes are Harmonic Broadcasting and Pyramid Broadcasting (PB). Optimizations and generalization of these schemes to Variable Bit-Rate (VBR) encoded media have been introduced, but for ease of exposition we concentrate on a simple but practical example and study PB for Constant Bit-Rate (CBR) encoded video. Our findings can be extended straight forward to other types of segmentation schemes, e.g., for those adapted to VBR.

PB suggests segmenting the video stream into S segments of increasing size or duration, with the length $|\mathcal{G}_i|$ being selected according to $|\mathcal{G}_i|=2^{i-1}|\mathcal{G}_1|$. Each segment is broadcast periodically on an individual channel with channel bit rate equal to the video bit rate. This results in bandwidth expansion $\omega=S$. Without loss of generality, we assume $\sum_i|\mathcal{G}_i|=T\cdot R_M$, with R_M denoting the bit-rate of the media stream. In the following explanation we restrict ourselves to the case of equal bit-rate of each channel with the bit rate of each channel being identical to the media rate R_M. The initial startup latency Δ is equal to the time required to entirely receive the first segment $\mathcal{G}_1$, i.e., $\Delta=|\mathcal{G}_1|/R_M$. By carefully selecting the overall number of segments S, initial startup latency Δ can be traded against bandwidth expansion ω. A shorter waiting time yields an increase in bandwidth. However, compared to the trivial solutions the bandwidth expansion is quite moderate.

Fig.2 sketches the principle of PB. In this illustrative example the video bit-stream is segmented into S=4 segments of different lengths. Each segment $\mathcal{G}_i$ is broadcast periodically on individual channels (Ch1~Ch4). Two sample receivers (Rx1, Rx2) tune into the broadcast session at different subscription times, i.e., asynchronously. The highlighted areas within the channels indicate data exploited by receiver Rx1. After the first segment $\mathcal{G}_1$ has been completely received, the receiver starts the playout of the first video segment and immediately stops listening to the first channel. As can be observed, the structure of the scheme is such that after finishing the playout of segment $\mathcal{G}_i$ it is guaranteed that segment $\mathcal{G}_{i+1}$ is completely available at the receiver buffer.

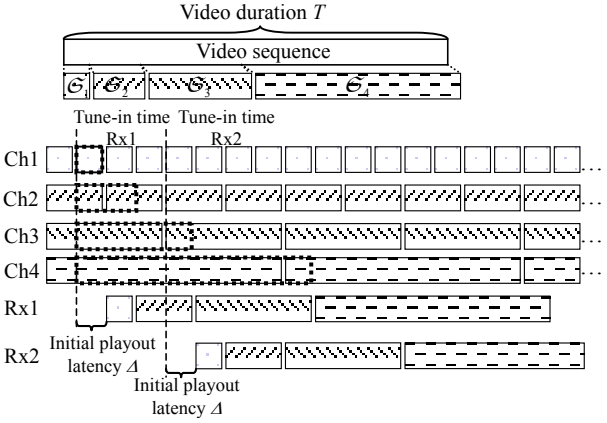

Fig.2 Pyramid broadcasting

Pyramid broadcasting with FEC

Originally, VoD broadcasting schemes were designed to operate over lossless transmission channels only, e.g., delivery over Local Area Networks (LANs). However, since specially wireless networks suffer from losses we will in the following discuss how PB can be extended by Forward Error Correction (FEC) to operate over lossy packet channels. Assume that data contained in individual segments $\mathcal{G}_i$ are

appropriately packetized into k_i packets of equal size prior to the transmission over the channels. Transmitted packets are assumed to be either received correctly at the receiver or are entirely lost.

By waiting for the next replica of a lost packet to be received this periodic transmission of the segments allows compensating for packet losses. However, this strategy results in an undesirable increase in playout latency. Therefore, it has been proposed to not just carousel the packets of the segments, but to transmit a certain amount of parity packets in addition to the original information packets (Xu, 2001; Jenkač and Stockhammer, 2005). These parity packets are usually obtained by some FEC code.

In general, FEC codes are employed in order to produce a set of encoding packets from a set of "information packets". For ease of exposition we will consider systematic codes in the following. Let k denote the number of information packets and g the number of produced parity packets, such that in total $n=k+g$ packets are available after FEC encoding, resulting in a code rate $r \triangleq k/n$. From coding theory it is known that so called Maximum Distance Separable (MDS) codes can be constructed which allow the reconstruction of all information packets from any k out of n received packets. Unfavorable properties in terms of flexibility, code rate and block length restrictions as well as decoding complexity has led to the development of flexible and low complexity erasure codes for which $k'=(1+\varepsilon)k$ of encoding packets are required at the decoder in order to achieve decoding success. Raptor codes for example have been thoroughly optimized towards $\varepsilon \to 0$, especially for large enough $k \geq 1000$.

Let k_i denote the number of information packets obtained from segment $\mathcal{S}_i$. Then, for each segment $\mathcal{S}_i$ an FEC code with rate r_i is applied in order to produce n_i encoding packets. Then, all n_i encoding packets are broadcast periodically on channel Chi, rather then only the k_i information packets. This allows receivers to recover segment $\mathcal{S}_i$ by receiving any k_i packets out of n_i, assuming an MDS code, and only slightly more for Raptor-like codes. However, with each lost packet the playout latency is increased at the receiver. Receivers can compensate for packet loss by just increasing their individual playout latency.

PROTOCOL ARCHITECTURE AND IPDC IN DVB-H

Content Delivery Protocols (CDP)

The DVB-h IP Datacast (IPDC) protocol stack (ETSI, 2005) is designed to transport different types of media such as audio, video, text, pictures, and binary files. The CDP protocol stack is depicted in Fig.3. Bearers provide the mechanism by which IP data is transported in DVB-h. Subsections 3.2 and 3.3 are dedicated to this aspect. The DVB-h IP broadcast bearer may be used jointly with p-t-p bearers in offering complete service capabilities. When delivering content to a receiving application one or more "delivery methods" may be used. Two delivery methods are defined, namely "download" and "streaming" delivery. The delivery layer provides functionality such as security and key distribution, reliability control by means of FEC techniques and associated delivery procedures such as file-repair and reception reporting. Finally, the IPDC "User service" enables applications. Different applications impose different requirements when delivering content to receivers and may use different delivery methods. Typical examples are that a software package update would use the download delivery while a TV broadcast application would use the streaming delivery based on RTP/RTCP.

For the case of VoD services, we propose to use the download delivery service rather than streaming delivery. The proposed extension is indicated by the shaded box in the Fig.3. File delivery in DVB-h builds on the FLUTE protocol (Paila *et al*., 2004). FLUTE allows reliable delivery of files and other discrete binary objects over unicast channels without back channels. File-based video delivery in FLUTE enables reliable VoD services and will be discussed in more detail in Section 4.

DVB-h time-slicing

The modulation scheme applied in DVB-T and DVB-h is OFDM. In DVB-T, TV channels are multiplexed to different OFDM sub-carriers. In order to demodulate and decode a certain channel, all sub-carriers have to be demodulated in parallel. Since this process of demodulating and decoding the total received signal consumes significant battery power, a time-slicing mechanism has been introduced for

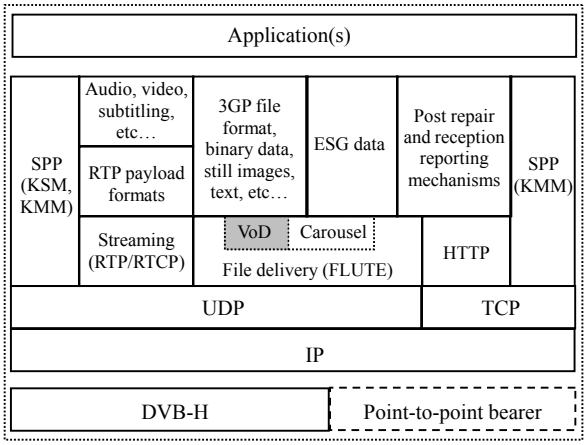

Fig.3 IP Datacast (IPDC) protocol stack in DVB-H. The gray box shows our proposed extension discussed in the next sections

DVB-h. High data rate bursts are transmitted in a short period of time, enabling the receiver to be power-off for most of the time. Each burst contains notification to the receiver of when to wake for the next burst.

In this manner, other DVB services can also be time-sliced and multiplexed in the transmitted stream. Each receiver can tune into its intended service(s). At the same time, a non-time-sliced MPEG-2 DVB-T stream can be multiplexed with the time-sliced DVB-h streams as shown in Fig.4. Time slicing also enables receivers to support efficient cell-to-cell transfer. During the off-time between bursts, the receiver can monitor the signal from the neighboring cells and switch between them if necessary. This smooth handover could not be achieved in the non-time-sliced case without some interruption of the service or without multiple receivers in the terminal.

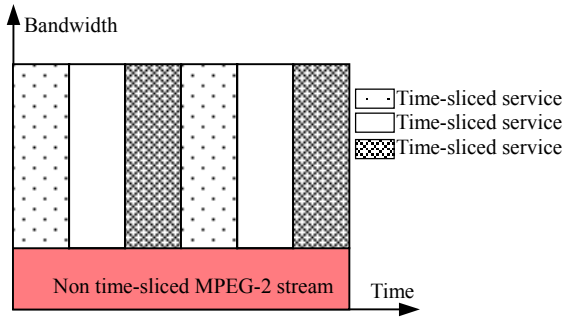

Fig.4 Time slicing in DVB-H

Multi-Protocol Encapsulation (MPE)

DVB-h introduces MPE-FEC, which is an additional FEC scheme based on Reed-Solomon codes. The objective of MPE-FEC is to compensate for the performance degradations due to Doppler effects in mobile channels. It is integrated in DVB-h in such a way that MPE-FEC ignorant receivers, e.g. stationary ones, can also receive the service. It is important to note that DVB-h only supports IP-datagrams or other network layer datagrams as its payload, not MPEG-2 transport streams.

An MPE-FEC frame can be envisioned as a matrix of 255 columns and a variable number of rows, as shown in Fig.5. Row sizes of 256, 512, 768, and 1024 are supported. Each entry in the matrix is 1 byte. The first 191 columns of the matrix are reserved for IP-datagrams and so this part of the frame is called the Application Data Table (ADT). Received datagrams are written into the frame column-wise one after another. The last 64 columns of the matrix are reserved for Reed-Solomon parity bytes, and so this part of the frame is called the RS Data Table (RSDT). RS (255,191) correction is performed on the ADT row-wise and thus providing interleaving. By either padding some broadcast data with zero bytes or by puncturing some parity columns, the additional error correction may be flexibly strengthened or weakened according to broadcast operator's needs for service specific adaptation.

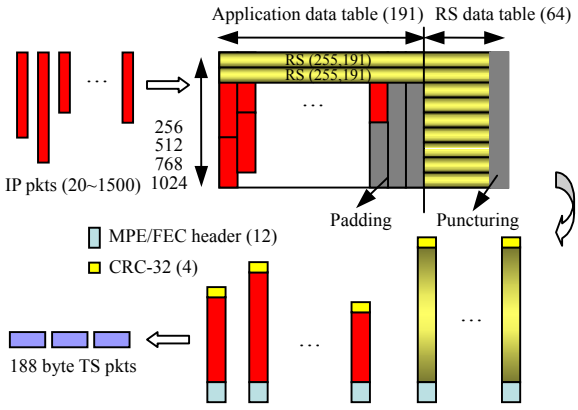

Fig.5 Multi-protocol encapsulation of IP-packets and MPE-FEC Reed-Solomon encoding

After the MPE-FEC frame is constructed, MPE-sections are built from the ADT table according to the DVB standard. Each IP-datagram is preceded by a 12 byte header containing information at its start and end position in the MPE-FEC frame. Additional

4 bytes of CRC-32 are appended. If the CRC of some section fails at the receiver, all the bytes of this section are marked as erroneous and thus the RS parity bytes are exploited to perform erasure correction instead of error correction. Similarly, FEC-sections can be built from the RSDT, where each RS column constitutes a section.

After the MPE and FEC sections are formed, they can be mapped directly to Transport Stream (TS) packets. Each TS packet has 183 bytes of payload, 5 bytes of header information, and 16 bytes resulting from yet another Reed-Solomon (n=204, k=188) error protection. Note that MPE-FEC ignorant receivers can still be supported. They can simply neglect TS packets that contain parts of RS parity sections, of course, at the expense of lower error resilience.

Reliable download delivery with FLUTE

File Delivery over Unidirectional Transport (FLUTE) (Paila *et al*., 2004) is a protocol for the delivery of files and other binary objects, e.g., images, text, documents, executables, video and audio, download services, from progressive download to background opportunistic download. FLUTE was originally designed for transmission over the Internet on top of UDP/IP and is particularly suited for one-to-many delivery in transmission environments without feedback channel. FLUTE is adopted as a content delivery protocol for download delivery services in DVB-h, see protocol stack in Fig.3. It builds on top of the Asynchronous Layers Coding (ALC) (Luby *et al*., 2002a) protocol instantiation. ALC combines the Layered Coding Transport (LCT) (Luby *et al*., 2002b), the Congestion Control (CC) and the Forward Error Correction (FEC) (Luby *et al*., 2002c) building blocks.

In general, files or, more generally, transport objects might span several kbytes or Mbytes, i.e., the file size is usually in magnitudes larger than the Maximum Transmission Unit (MTU) of the underlying network. Therefore, an appropriate packetization before delivery is required. FLUTE basically supports a two-step segmentation of transport objects to be delivered. Assume that the transport object size is denoted as L in bytes. In a first step the binary representation of the transport object can be segmented into a certain number S of smaller blocks referred to as "source blocks" by a blocking algorithm. The blocking algorithm determines the "source block structure", i.e., number of source blocks S and assigns to each source block a certain source block size K_i. The maximum source block length may be determined by some buffer constraints in receivers, as well as some limitations of the applied FEC. However, in general the blocking can be quite flexible in the sense that the minimum and maximum source block size is not significantly restricted in FLUTE or DVB-h CDP specifications.

Source blocks are further fragmented into K equal sized source symbols with source symbol length T in bytes, where K denotes the source block size (in number of encoding symbols). Source symbols are the smallest data units to be transmitted over the network. Usually, the source symbol length is selected such that in combination with all additional headers the MTU size of the underlying network is not exceeded. In total the entire transport object is fragmented into E source symbols, where $E=L/T$. Fig.6 visualizes the blocking and symbol encoding encoding employing a generic FEC scheme.

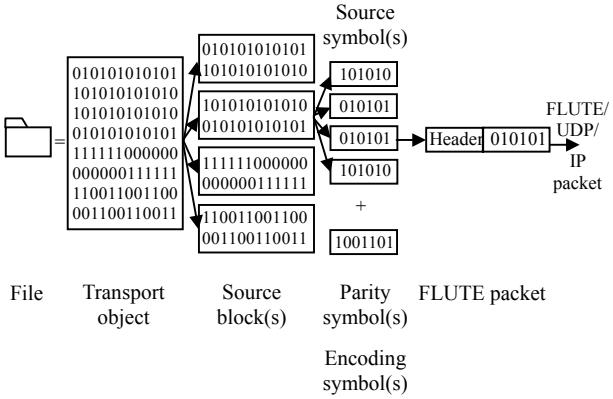

Fig.6 Blocking of transport objects into encoding symbols and mapping to FLUTE packets (Peltotalo *et al*., 2005)

FLUTE supports the usage of an optional symbol encoding algorithm, i.e., an FEC code, in order to provide reliability. FEC encoding is performed for each source block individually. Encoding symbols have the same length as source symbols, T. In case that a systematic FEC code is applied, the first K encoding symbols are identical to the source symbols and the remaining $N-K$ symbols are parity symbols. The selection of T, K, and N is quite flexible and is up to the transmitter. The encoding symbols are encap-

sulated in FLUTE packets and mapped to UDP/IP packets before being forwarded to the MPE protocol or any other IP-based bearer.

To allow appropriate reconstruction at the receiver, all parameters along with the position of each encoding symbol within the encoding blocks, the affiliation to encoding blocks and the Transport Object Identifier (TOI) have to be communicated to the receiver. For this, two methods are foreseen within FLUTE: Either the FLUTE header, which precedes each encoding symbol or a group of encoding symbols, or a special transport object, i.e., the File Delivery Table (FDT). The position of each encoding symbol within the encoding block, the corresponding source block number as well as the TOI are communicated within the FLUTE header, which allows the receivers to use the received packets. Additional and insightful information on signaling can be found for example in (ETSI, 2005; Paila *et al*., 2004) and references therein.

The receivers might reconstruct lost encoding symbols by applying FEC decoding. According to (ETSI, 2005), DVB-h receivers shall support the Compact No-Code FEC scheme and it is recommended that Raptor FEC scheme should be supported by DVB-h receivers. Note the Raptor code is mandatory for MBMS and might therefore be included in many handheld devices anyway.

FLUTE allows establishing multiple FLUTE channels for transmission (Paila *et al*., 2004) in parallel. FLUTE channels are logical channels defined by the pair of specific "address" and "port" number, as well as a transmission rate. Sets of encoding symbols of a transport object might be mapped to multiple FLUTE channels concurrently, allowing receivers to only subscribe to a sub-set of offered channels. Receivers subscribing to multiple channels might be able to receive the transmission object in shorter reception time. The CDP specification of DVB-h specifies that single FLUTE channels shall be supported within DVB-h, but multiple FLUTE channels may be supported by terminals. However, it is required that for terminals supporting only a single FLUTE channel, it should be possible to complete the reception of the transport object from the first channel.

In summary, FLUTE allows the transmitter significant freedom to distribute transport objects to receivers. However, thereby it is necessary to understand that receivers with different receiving capabilities might exist in the sense that multiple channels and/or Raptor FEC are only supported by a subset of terminals. These constraints should be taken into account by the transmitter.

INTEGRATION OF VOD BROADCASTING INTO FLUTE AND DVB-H

Preliminaries

The trivial approach for providing asynchronous tune-in by receivers through a carousel mechanism is already foreseen and included in the current DVB-h CDP specification (ETSI, 2005). As shown in Fig.3, carouseling is realized as a certain type of file delivery mechanism with appropriate utilization of FLUTE. Moreover, the DVB-h CDP specification already foresees carousel mechanisms enhanced by the Raptor FEC coding scheme.

Rather than simply reducing the download time of the entire video file by expanding the transmission bandwidth, the objective of our work is to reduce the initial start-up latency Δ regardless of the actual download time. For this, we consider the integration of the pyramid broadcasting scheme, as discussed in Section 3, into FLUTE and DVB-h. We already indicated our proposed extension, which we discuss in the remainder of this section, in the protocol stack shown in Fig.3.

As explained in Section 2, a key component of PB with and without FEC is the segmentation of the video bit-stream and broadcasting resulting segments in combination with their corresponding parity data on different channels. Fortunately, FLUTE already provides appropriate means to accomplish these features. FLUTE allows opening multiple transmission channels simultaneously, supports segmentation by a blocking algorithm, and allows FEC by the usage of, e.g., the Raptor FEC scheme. However, the blocking and mapping to channels in (ETSI, 2005) neither proposes PB, nor does it prevent it. In the following we introduce and discuss a modified blocking algorithm as well as an appropriate mapping of segments to FLUTE channels. The proposed extensions are transmitter only modifications. Our strong objective is to propose a backward-compatible scheme which

allows a state-of-the-art DVB-h/FLUTE receiver to participate in the VoD services. In addition, receivers with optional extensions of multiple channels, Raptor FEC, and possibly specific signaling should benefit from the modified scheme.

Proposed algorithm

First an appropriate source symbol size E in bytes is selected such that the MTU is not exceeded. Note, for each source/encoding symbol to be transmitted an overall FLUTE/UDP/IP header overhead of H bytes has to be taken into account. Given the video-stream length (file size) L in bytes, the overall number of resulting source symbols T is obtained by $T=L/E$. In contrast to traditional parameterization of PB, where a desired initial start-up latency is selected and from which the number of channels is calculated, we fix the number of channels to N_{ch}, resulting in bandwidth extension $\omega=N_{\text{ch}}$. This approach is selected for fair comparison of the proposed PB scheme with a carousel approach having the same bandwidth expansion.

We propose a modified "blocking algorithm" to determine source blocks $\mathcal{U}_i$ of (different) source block size K_i in symbols corresponding to the calculation of segments sizes $|\mathcal{G}_i|$ according to the PB scheme presented in Section 2. Rather than employing the recommended algorithm in (ETSI, 2005) or (Paila *et al*., 2004), which outputs equal length source blocks, we propose source block lengths $K_i=|\mathcal{G}_i|/E$ (in source symbols) according to

$$K_i = T \cdot 2^{i-1} \Big/ \sum_{j=1}^{S} 2^{j-1}, \tag{1}$$

where by $S=N_{\text{ch}}$. A trivial solution for the integration into FLUTE opens multiple FLUTE channels and maps the source symbols from source block i, indexed with $k_i=1, \ldots, K_i$, periodically to the FLUTE channel i. This is one possible PB realization as already discussed in Section 3.

However, this approach involves several problems. First, no FEC is applied and, second, as stated earlier, the CDP specification of DVB-h requires that receivers must be able to reconstruct transport object from the first channel only. To overcome these problems, we propose a solution which (1) allows incorporating FEC, (2) allows receivers to reconstruct the entire transport object (video-stream) by only listening to the first channel, and (3) allows Raptor FEC ignorant receivers to receive the transport object as well.

To address (1) each source block $\mathcal{U}_i$ of size K_i source symbols is encoded independently applying the Raptor FEC scheme. The FEC encoder outputs encoded blocks $\mathcal{E}_i$ containing N_i encoding symbols indexed by $n_i=1, \ldots, N_i$. We concentrate exclusively on systematic Raptor in the following, i.e., the first K_i encoding symbols of $\mathcal{E}_i$ are identical to the source symbols $\mathcal{U}_i$. Similar to Fig.4, encoding symbols from $\mathcal{E}_i$ are periodically mapped to FLUTE channel i. The FLUTE packet headers are appropriately utilized to convey the required information, i.e., the encoding block number i and the encoding symbol index n_i. All further parameters are communicated to the receivers within the FDT. In combination with each FLUTE header a receiver knows where to insert the received encoding symbols into the receiver's decoder buffer. Still, requirement (3) is realized only poorly, since a receiver is required to wait exclusively for the systematic encoding symbols. Consequently, the initial start-up latency especially for receivers in bad conditions is significant. Moreover, this scheme does not address requirement (2) at all. A receiver which is restricted to listen only to the first channel can only reconstruct the first segment.

Therefore, we propose an alternative to the previous scheme, or to be more specific we change the mapping of encoding symbols to FLUTE channels, such that all the above stated requirements are met. In the previous scheme, encoding symbols n_i form source block (segment) $\mathcal{U}_i$ have only been mapped to FLUTE channel i. Fig.7 shows examples for $S=4$ our proposed modifications to PB, referred to as FLUTE-PB in the remainder of this section.

In order to meet requirement (2), i.e., reception must be possible exclusively from the first channel, and in order to meet in addition requirement (3), the only solution is to periodically broadcast all source symbols on the first channel, which actually is identical to the trivial carousel approach. However, on the remaining channels, i.e., $i=2, \ldots, N_{\text{ch}}$, parity

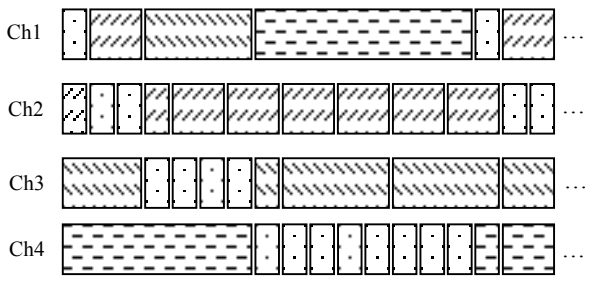

Fig.7 Proposed FLUTE-PB scheme: on the first channel simple carouseling of the entire video sequence is performed. The other channels carry only repair symbols

(repair) symbols of $\mathcal{E}_i$ are transmitted periodically except for the time instants where on the first channel the source symbols of the corresponding source block $\mathcal{U}_i$ are transmitted. At these time instants instead of parity symbols of $\mathcal{E}_i$ parity symbols of $\mathcal{E}_1$ are mapped to channel i. Note, receivers can still tune-in at arbitrary time, since the encoding symbols are only mapped to different channels. The receiver does not need to be aware of the mapping performed at transmitter. The FLUTE entity at the receiver just receives encoding symbols on any subscribed channel. How to use a received encoding symbol is clarified after evaluation of the FLUTE header.

PERFORMANCE EVALUATION

Simulation environment

We will briefly explain our simulation environment to evaluate the integration of VoD services in DVB-H download delivery. For this purpose different existing implementations have been used and modified. Fig.8 shows the simulation environment which basically implements the protocol stack as shown in Fig.3. The simulator consists of two main modules: for the CDP/FLUTE part, we used the open-source implementation of FLUTE from the MAD project (MAD/TUT, http://www.atm.tut.fi/mad). Modifications were integrated in MAD to support our new proposed blocking algorithm. In addition, Raptor FEC according to the specification in DVB-H CDP was added. For the DVB-h IPDC part, we implemented a DVB-h simulator according to the DVB-h standards based on the description of the simulator implementation in (TM-CBMS1361, 2005). The simulator includes MPE-FEC as well as time-slicing. The DVB transport stream is simulated using traces generated from offline simulations (TM-CBMS1361, 2005). The error traces include effects such as fading in a typical urban environment, shadowing, and different Doppler frequencies. We concentrate in the following on Doppler frequency of 1 Hz. Traces are provided for carrier-to-interference ratios (C/I) of {9, 12, 15, 18, 21} dB which reflect different reception conditions.

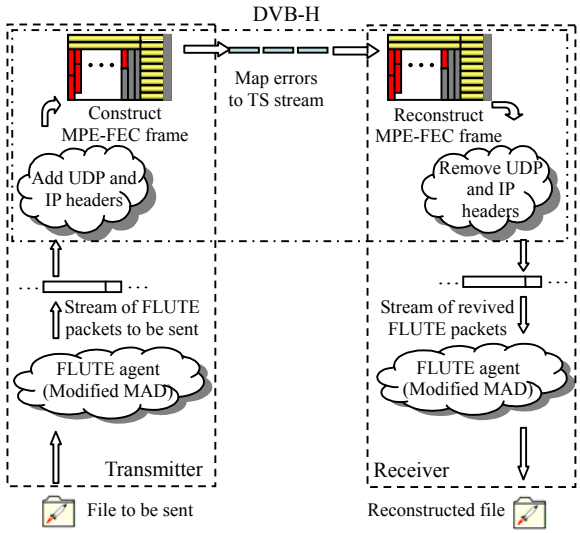

Fig.8 Block diagram of the simulation environment

The DVB-H IPDC simulator allows us to simulate the delay and loss characteristic of each inserted IP packet. The MPE-FEC strength can be selected, for the following simulations we apply two modes: "with MPE-FEC" refers to an MPE-FEC rate of RS (191,255) and "without MPE-FEC" uses the entire available bandwidth for the transmission of the IP stream. Different bearer bitrates can be controlled by appropriate access and time-slicing schemes. The applied bitrates will be discussed in more detail in the simulation results.

Fig.8 shows the file-based simulator operation. Assume a file is to be broadcast. The MAD software takes care of partitioning this file into source blocks, adding FEC, and producing the FLUTE packets. Then, DVB-h emulator reads these packets, adds UDP and IP headers, constructs the MPE-FEC frames, and segments the resulting sections into TS packets. This represents the transmitter side. Error patterns corre-

sponding to different users and receiving conditions are then mapped to the TS stream. At the receiver side, the TS stream is used to reconstruct the MPE-FEC frame. If applicable, MPE-FEC erasure correction is performed, and only correctly received FLUTE packets are delivered. MAD receiver software then receives these packets and attempts to reconstruct the original file applying FEC decoding.

At the receiver it is evaluated at which time after the receiver starts listening to the ongoing broadcast session a certain source block can be reconstructed. This depends on the source block size, on the observed loss patterns, on the FEC applied, and obviously on the receiving conditions. For worse receiving conditions the time to recover the source block is expected to be longer. In order to provide early playout functionality and to guarantee smooth playout it is necessary that all source blocks are available at the time they have to be played out. We evaluate the necessary initial start-up latency such that all segments can be played out smoothly. Note that the determination of this value is non-trivial, and rather uncritical. Details are discussed in (Jenkač and Stockhammer, 2005).

Simulation results

In order to show the benefits of our proposed method compared to an FEC carousel system, we performed extensive simulations. We simulated the broadcast of a video clip of duration T_d=294 s, with bitrate R_M=192 kbps, including audio and video. The video and audio was CBR encoded. We selected a source/encoding symbol size of T=1404 bytes to not exceed the MTU of 1500 bytes.

For fair performance comparison of the FEC carousel mechanism and our proposal (FLUTE-PB), both schemes are evaluated for ω={4,5}, i.e., the reference case (FEC carousel) is given equivalent bandwidth as the FLUTE-PB scheme, but only utilizing a single FLUTE channel. For both schemes we selected the lowest possible Raptor code rate supported by the FEC encoder, i.e., the maximum encoding symbol length N=65536. This reduces frequent reception of already received encoding symbols. More background on the suitability of this fountain approach is for example discussed in (Peltotalo *et al*., 2005). Moreover, we also investigate both schemes with MPE-FEC and without MPE-FEC. To have a fair comparison, the resources consumed on the air interface (bandwidth) are identical for both MPE schemes. Specifically, we fixed the rate of TS packets. Switching off MPE-FEC will therefore result in increased FLUTE channel bitrate, in our case 192 kbps×(255/191)=256 kbps in the no MPE-FEC case. In addition, we evaluated the start-up latencies for a receiver that only listens to the first channel in order to receive the file.

Fig.9 presents our simulation results for bandwidth expansion ω=4. The figure shows the average required initial start-up latency Δ vs the C/I in dB. First consider the reference case (middle 2 curves), i.e., the FEC carousel mechanism. As expected, for good channel conditions, i.e., high *C/I*, the initial start-up latency Δ was about T_d/ω, since no early playout can be supported with this scheme, but only plain download-and-play. With decreasing *C/I*, i.e., worse channel conditions, the initial start-up latency Δ was increasing, since losses have to be compensated for by listening to more encoding symbols. Interestingly, switching-off MPE-FEC shows the same behavior but reduces Δ slightly. At first glance, this is not intuitive. However, in contrast to the Raptor FEC, where the receiver benefits from each additionally received encoding symbol, with MPE-FEC, a receiver cannot benefit from additional parity Reed-Solomon if it already received enough of them. Replacing the band-

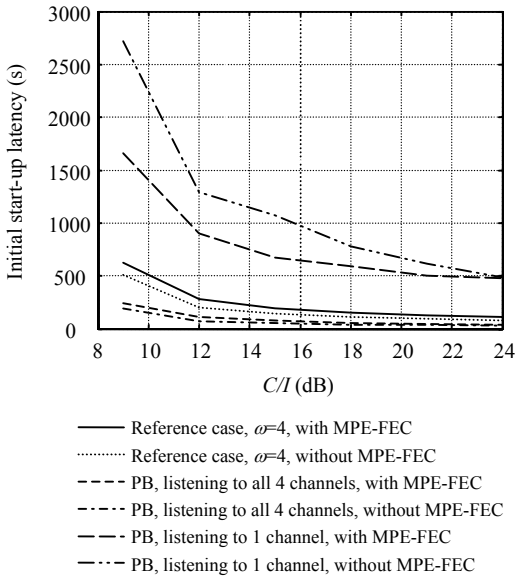

Fig.9 Simulation results for ω=4 and stream duration T_d=294 s

width occupied by the MPE-FEC with Raptor symbols proved advantageous. Moreover, note that without MPE-FEC the FLUTE channel can be operated at higher rate. These two effects explain the observed gains.

The bottom 2 curves show the performance of our proposed scheme, i.e., FLUTE-PB, for a receiver subscribing to all offered channels. As can be observed significant gains can be expected in the entire C/I region, compared to the carousel approach. For receivers with bad channel conditions the initial playout delay Δ is halved. Again, cancelling MPE-FEC gives further improvements.

The top 2 curves show the evaluation of our proposed FLUTE-PB scheme with a receiver subscribing only to the first FLUTE channel of the session. This might have several reasons, e.g., insufficient computational power, limitations in processing power, etc. This shows the worst performance of all investigated cases. However, download is still possible and the requirements are met. However, since on the first channel no Raptor FEC is applied, in this case MPE-FEC proved advantageous compared to the case without as proved application layer FEC was available.

Fig.10 shows the results for the same experiments but for ω=5. Basically the same tendencies can be observed like in the ω=5 case, but with reduced Δ, showing again the superiority of our proposed broadcasting scheme.

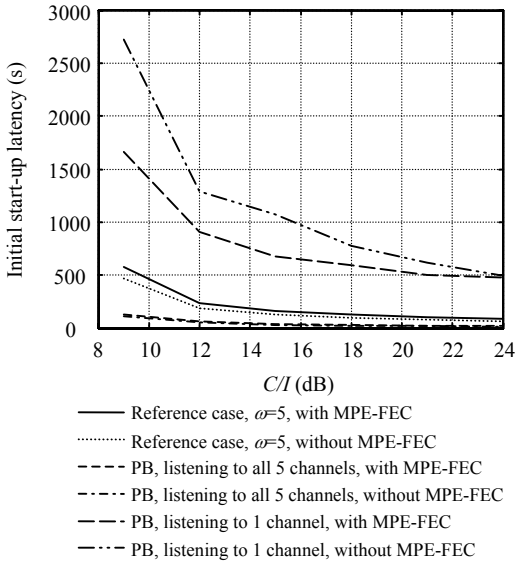

Fig.10 Simulation results for ω=5 and stream duration T_d=294 s

CONCLUSION

In this work we investigated the integration of reliable Video-on-Demand (VoD) broadcasting schemes into the DVB-h IP datacasting in order to reduce the initial start-up latency compared to state-of-the art carouseling mechanisms. The integration of Pyramid Broadcasting (PB) into the FLUTE protocol was studied. We proposed a modified PB scheme in combination with Raptor coding, which meets the requirements of the current DVB-h CDP specification and is therefore backward-compatible. Significant benefits of our scheme compared to carousel approaches have been shown, in a fair comparison, where both schemes utilize the same bandwidth. Our scheme was shown to be scalable in terms of receiver capabilities. Raptor FEC ignorant receivers and receivers restricted in the number of FLUTE channels are also able to retrieve the video-sequence, although with increased start-up latency. Moreover, our scheme was shown to require transmitter modifications only, while a state-of-the art DVB-h receiver can immediately benefit from the transmitter modifications. Future work will consider appropriate signaling of transmitter behavior to the receivers, the integration of more sophisticated VoD broadcasting schemes in order to reduce the start-up latency further, as well as cross-layer issues which allow joint optimization of the parameter selection like MPE-FEC, Raptor rate, segmentation, etc. Currently we are implementing this scheme in a real-time emulator. Preliminary tests show that these types of schemes pave the road to more quality-of-service and user satisfaction.

References

Engebretsen, L., Sudan, M., 2002. Harmonic Broadcasting is Bandwidth-Optimal Assuming Constant Bit Rate. Proc. Annual ACM-SIAM Symposium on Discrete Algorithms. San Francisco, CA, USA.

ETSI, 2005. IP Datacast over DVB-H: Content Delivery Protocols. ETSI Standard, Draft, V0.0.9.

Horn, G.B., Knudsgaard, P., Lassen, S.B., Luby, M., Rasmussen, J.E., 2001. A scalable and reliable paradigm for media on demand. *IEEE Computer*, **34**(9):40-45.

Hu, A., 2001. Video-on-Demand Broadcasting Protocols: A Comprehensive Study. Proc. IEEE Infocom. Anchorage, Alaska.

Huang, C., Janakiraman, R., Xu, L., 2004. Loss-Resilient Media Streaming Using Priority Encoding. Proc. ACM

International Conference on Multimedia (MM'04). New York, USA.

Jenkač, H., Stockhammer, T., 2005. Asynchronous Media Streaming over Wireless Broadcast Channels. Proc. of International Conference on Multimedia and Expo (ICME). Amsterdam, The Netherlands.

Luby, M., Gemmel, J., Vicisano, L., Rizzo, L., Handley, M., Crowcroft, J., 2002a. Asynchronous Layered Coding (ALC) Protocol Instantiation. RFC 3450, IETF.

Luby, M., Gemmel, J., Vicisano, L., Rizzo, L., Handley, M., Crowcroft, J., 2002b. Layered Coding Transport (LCT) Building Block. RFC 3451, IETF.

Luby, M., Vicisano, L., Gemmel, J., Handley, M., Crowcroft, J., 2002c. Forward Error Correction (FEC) Building Block. RFC 3452, IETF.

Luby, M., Watson, M., Gasiba, T., Stockhammer, T., Xu, W., 2006. Raptor Codes for Reliable Download Delivery in Wireless Broadcast Systems. Proc. Consumer and Communications Networking Conference (CCNC). Las Vegas, NV, USA.

Paila, T., Luby, M., Lehtonen, R., Roca, V., Walsh, R., 2004. FLUTE—File Delivery over Unidirectional Transport. RFC 3926, IETF.

Peltotalo, J., Peltotalo, S., Harju, J., 2005. Analysis of the FLUTE Data Carousel. Proc. 10th EUNICE Open European Summer School. Colmenarejo, Spain.

Shokrollahi, A., 2003. Raptor Codes. Tech. Rep. DR2003-06-001, Digital Fountain.

TM-CBMS1361, 2005. Proposal for Simulations for Evaluation of Application Layer FEC for File Delivery.

Xu, L., 2001. Efficient and Scalable on-Demand Data Streaming Using UEP Codes. Proc. ACM International Conference on Multimedia (MM'01). Ottawa, Ontario, Canada.