Telecommunication Systems manuscript No.
(will be inserted by the editor)

Delivering scalable video with QoS to the home

Chris Develder · Peter Lambert · Wim Van Lancker · Stefaan Moens · Rik Van de Walle · Jelle Nelis · Dieter Verslype · Steven Latré · Nicolas Staelens · Nick Vercammen · Brecht Vermeulen · Bart Masschelein · Tom Van Leeuwen · Jean-Francois Macq · Kris Struyve · Filip De Turck · Bart Dhoedt

Received: 15 May 2009 / Accepted: 16 November 2009

Abstract User satisfaction is a key factor in the success of novel multimedia services. Yet, to enable service providers and network operators to control and maximize the quality (QoS, QoE) of delivered video streams, quite some challenges remain. In this paper, we particularly focus on three of them. First of all, objectively measuring video quality requires appropriate quality metrics and methods of assessing them in a real-time fashion. Secondly, the recent Scalable Video Coding (SVC) format opens opportunities for adapting video to the available (network) resources, yet the appropriate configuration of video encoding as well as real-time streaming adaptation are largely unaddressed research areas. Thirdly, while bandwidth reservation mechanisms in access/core networks do exist, service providers lack a means for guaranteeing QoS in the increasingly complex home networks (which they are not in full control of). In this paper we offer a broad view on these interrelated issues, by presenting the developments originating in a Flemish research project (including proof-of-concept demonstrations). From a developmental perspective, we propose an architecture combining a real-time video quality monitoring platform, on-the-fly adaptation (optimizing the video quality) and QoS reservation in a heterogeneous home network based on UPnP QoS v3. From a research perspective, we propose a new subjective test procedure that revealed user preference for temporal scalability over quality scalability. In addition, an extensive study on optimizing HD SVC encoding in IPTV scenarios with fluctuating bandwidth showed that under certain bandwidth constraints (prohibiting sufficient fidelity) spatial scalability is a better option than quality scalability.

Keywords QoS · SVC · H.264 · adaptation · monitoring · UPnP · PQoS

The work presented in this paper was supported by the Flemish government through the Interdisciplinary Institute for Broadband Technology (IBBT), via the Video Q-SAC project. C. Develder is supported by the Research Foundation – Flanders (FWO–Vl.) as a postdoctoral fellow.

C. Develder, J. Nelis, D. Verslype, S. Latré, N. Staelens, N. Vercammen, B. Vermeulen, F. De Turck, B. Dhoedt
Ghent University - IBBT, Dept. of Information Technology (INTEC), IBCN
G. Crommenlaan 8 bus 201, BE-9050 Ghent, Belgium
Tel.: +32-9-3314900
Fax: +32-9-3314899
E-mail: chris.develder@intec.ugent.be

P. Lambert, W. Van Lancker, R. Van de Walle
Ghent University - IBBT, Dept. of Electronics and Information Systems, Multimedia Lab,
G. Crommenlaan 8 bus 201, BE-9050 Gent, Belgium
E-mail: peter.lambert@ugent.be

B. Masschelein
IMEC, Kapeldreef 75, BE-3001 Heverlee, Belgium

T. Van Leeuwen, J.-F. Macq, K. Struyve
Alcatel-Lucent Bell NV, Copernicuslaan 50, BE-2018 Antwerp, Belgium

1 Introduction

Despite increasingly widespread adoption of broadband access networks, ever-increasing use of multimedia (especially video), and advances in coding technology, the achieved delivery of video streams frequently is still unsatisfactory. The major cause is that there is only limited support of Quality of Service (QoS) in the current networks. This is particularly true of home networks, where for instance the presence of wireless channels may impair the video quality.

This paper addresses some of the key issues in solving the qualitative delivery of video streams. The results presented were obtained in the frame of the IBBT

Video Q-SAC project, addressing some key issues opening the way to qualitative broadband multimedia services. This project's scope and contributions are briefly summarized in subsection 1.2 after the overview of related work in the following subsection 1.1. A basic requirement to assess video quality is exactly monitoring and measuring the achieved quality, as discussed in Section 2. When network parameters do not suffice to deliver the video in its original format, a possible workaround is to adapt the video to the available network resources (bandwidth). The latter is made easier thanks to the recently standardized Scalable Video Coding (SVC) standard: the encoding issues and adaptation are described in detail in Section 3. The techniques for allocating network resources to ensure parameterized QoS, focusing in particular on the home network, is discussed in Section 4. The overall conclusions of the paper are summarized in Section 5.

1.1 Related work

A prerequisite for any QoS-enabled delivery of video is a means to assess the video quality. This challenges the operator/service provider to offer scalable measuring and monitoring tools to determine possible QoS problems, including those caused by the home network which he has no full control over. Moreover, existing main metrics as designed for data traffic delivery will not suffice, given the growing importance of (latency sensitive and bandwidth consuming) video traffic.

The most reliable assessment of visual quality can be achieved with subjective tests, involving human observers, based on standardized methodologies [14,16, 15,4]. However, since such tests are expensive and time consuming, a lot of research is currently ongoing towards the construction of objective metrics which are capable of measuring quality as perceived by the end users in an automated manner. Two of the most widely used objective metrics are Peak Signal-to-Noise Ratio (PSNR) and Structural SIMilarity (SSIM) [38]. These are examples of Full-Reference pixel-based quality metrics: they are on frame-by-frame comparison between the original and the degraded video sequences. Recently, the International Telecommunications Union (ITU) released two new recommendations for multimedia video quality measurement in the presence of a full reference [17] or reduced bandwidth reference [18], which are again pixel-based quality metrics involving decoding the degraded video sequence. However, in order to estimate visual quality in real-time, new metrics are needed which do not require the full decoding of the received video signal. Furthermore, it is clear that the original video sequence is not available during real-time video streaming.

Therefore, Ries et al. [31] propose a bitstream-based metric estimating video quality for mobile H.264/AVC streaming: the original video sequence is first classified based on its content into one of five predefined content classes (which have a different impact on user perception). This information is then used as input for the quality metric. Results indicate a good correlation with the subjective Mean Opinion Score (MOS) in case no network impairments occur. Kanumuri et al. [20,21] provide models for predicting the visibility of packet loss in MPEG-2 video, and as such offer a very rudimentary bitstream-based indication of quality.

To correctly assess the video quality, statistics of its delivery need to be collected at several demarcation points in the network. To aggregate this, a monitoring platform is required for distributing this abundance of information in (quasi) real-time. In the past, several monitoring platforms have been proposed. The perfSONAR framework [12] is a service-oriented monitoring architecture developed in the context of Geant2 [1] and intented to monitor the performance in multi-domain networks. The perfSONAR framework provides a set of services that facilitate the exchange of information between multiple domains. These services include monitoring services (e.g. SNMP based) as well as visualization and authentication services. As perfSONAR is targeted at multi-domain networks, it is primarily used in large-scale environments and less suited for service monitoring where fine grained information is important (such as for video quality monitoring). Another successful monitoring platform is SCAMPI [7], developed in the context of the IST SCAMPI project and continued in IST LOBSTER. SCAMPI is intended as a scalable monitoring platform for performing measurements in Gigabit networks. While concepts of SCAMPI are certainly useful for monitoring video quality, the intention of the platform itself diverges too much from the video monitoring case. Also in the context of grids several monitoring platforms have been proposed [26,6]. Again, these monitoring frameworks are intended for a specific platform (i.e. grids) and often lack desired functionality (e.g. QoS service monitoring) or provide functionality (e.g. job monitoring) that is useless for video quality monitoring.

With respect to providing QoS guarantees to video streams, the major challenge resides in the home network, since this is not entirely under control of the network operator. Up to the home gateway, well-known mechanisms such as (G)MPLS ((Generalized) Multi-Protocol Label Switching) can be used to set-up the necessary reservations, e.g. based on Carrier Ethernet

[9]. Providing these reservations in both backbone and access have been successfully addressed, even considering variations in the required bandwidth [3,10]. In the home however, the provider has only limited control, and the complexity of dealing with a heterogeneous home network comprising multiple layer-2 technologies (e.g. WLAN, HomePlug, etc.) arises. In [5], the authors present a possible solution based on UPnP-QoS v1 and Remote Management in Diffserv (RMD). Non-standardized extensions towards parametric QoS, providing absolute guarantees rather than (relative) prioritization, are proposed in [34]. Lee et al. [22] propose extensions to UPnP-QoS v2 for monitoring and also consider temporal scaling (frame rate reduction) as video adaption technique, which they assume is provided by the media server providing the video stream (from within the home).

With the advent of Scalable Video Coding [33], a single video file/stream can simultaneously contain multiple quality levels. This enables more efficient content delivery that targets a heterogeneous group of consumer devices. In [25] a number of plausible applications and their requirements are presented, including some that concern SDTV and HDTV. However, no optimal configurations were suggested to achieve this. Wien et al. do look into this matter [39], but its scope does not exceed the spatial resolution of SDTV.

1.2 Our contributions

The work presented in this paper discusses advances beyond the aforementioned state-of-the-art in the following domains: (i) quality measurement methodology, (ii) optimization of scalable encoding (SVC) and its adaptation, (iii) architecture and proof-of-concept implementation for streaming video to the home, incorporating UPnP QoS v3.

With respect to measuring and monitoring video quality, our main contribution in this field is the development of an automatic test framework for deploying, testing and validating new video quality metrics, described in Section 2.2. Using this test framework, both network related (loss rate, delay, jitter) and video related statistics can be gathered. (The latter is important for the construction of new video quality metrics since it is generally known that the type of video content plays an important role in the visibility of visual impairments.) A major benefit of our test framework is that it can run different experiments simultaneously and thus reducing overall experiment duration. In addition, we propose a new methodology to assess the perceived QoE by the end users: subjective tests based on offering full length movies to the test subjects revealed that users, in general, favor temporal scalability over quality scalability [35].

With respect to encoding optimization, we investigated how SD and HD could be combined in one SVC stream while maintaining optimal quality, and how to optimize quality by appropriate SVC encoding under certain target bandwidth constraints. Extensive test results are explained in Section 3.1. Next to optimizing the encoding, we also implemented a so-called bitstream Adaptation Engine (BAE), offering a powerful adaptation mechanism for SVC. Section 3.2 elaborates on the latter.

The aforementioned BAE is included in the architecture we propose for delivering video with QoS into the home, as discussed in Section 4. For this, we contributed to the definition of UPnP QoS v3 (thus advancing beyond v1 used in [5], or v2 in [22].) The major difference with the proposal of e.g. [22] is that v3 (i) offers parameterized QoS, allowing for explicit bandwidth reservations, and that it (ii) can rely on layer-2 (L2) capabilities for detecting network problems rather than relying exclusively on explicit monitoring by UPnP components. We also developed a QosDevice framework to facilitate UPnP QoS v3 implementation on various L2 technologies. With respect to adaptation of the video to cope with e.g. bandwidth constraints, we propose a generic BAE. In contrast with [22] (which only offers temporal scalability with H.264/AVC, assumed to be realized directly by each and every media server within the home), we support all forms of scalability in SVC and moreover in streaming scenarios both from within the home (media servers) and from streaming servers. For the latter, our architecture also enables control of the local QoS reservation processes (based on UPnP) by service providers through remote management. By exploiting the plug-and-play discovery of services typical of UPnP, both the user and service provider (SP) are relieved from extensive (manual) configuration, while the SP nevertheless is able to control QoS reservations without requiring explicit knowledge of the home network constellation.

The above contributions have been achieved within the frame of the Video Q-SAC project. Within Video Q-SAC, we developed and demonstrated (in a proof-of-concept) a framework for configuration, enforcement, monitoring and measurement of end-to-end QoS delivery from service/content provider to the user, focusing on video streaming. For this, we combined the above contributions (monitoring framework to measure video quality, optimized encoding of the offered SVC streams, on-the-fly adaptation, and QoS reservation in the heterogeneous home network) to realize the architecture

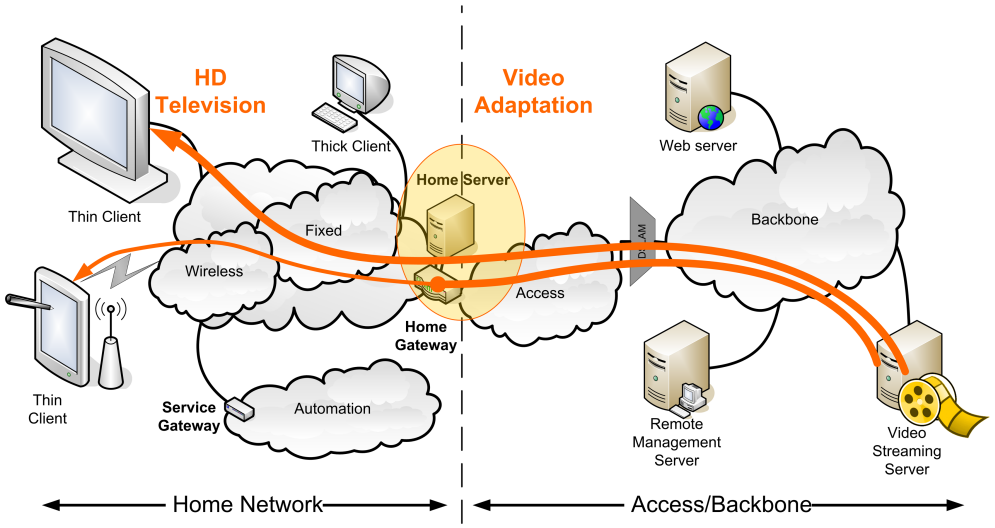


Fig. 1 The IBBT Video Q-SAC project focuses on qualitative delivery of video to the home network, involving bandwidth reservation and video adaptation.

outlined in Fig. 1. We consider the backbone and access network interconnecting the home networks to various service/content providers. A Remote Management Server (RMS) makes the necessary configurations to allow successful service delivery (e.g. configuration of the Home Gateway, HGW). A Video Streaming Server (VSS) will stream the actual content towards the home users. One of the challenges addressed is to achieve the ideal solution where the VSS has to store/offer only a single encoded video stream, optimized to maximize video quality even when (dynamically) adapted (see Section 3). The latter can be realized using the aforementioned BAE, which could be part of a Home Server close to the Home Gateway (HGW, interconnecting the home with the access network). Thus, we realize QoS-enabled delivery of both video streams from service/content provider to a broad range of user devices (and possibly vice versa, cf. surveillance cameras in the home) and between user devices within the home network (e.g. alarm-triggering surveillance camera preempting a digital television broadcast signal).

2 Quality monitoring and measuring

To take appropriate actions in the case of severe visual degradations (see Section 3.2), service providers require (quasi) real-time monitoring tools. In this context, the achieved Quality of Experience (QoE) is of particular interest. This calls for a monitoring platform providing a video quality metric, rather than more traditional Quality of Service (QoS) measures such as packet loss etc. Solutions proposed in literature range from translating IP packet loss to a quality metric [30], to more in-depth analysis on higher layers [28,29].

We developed such a real-time video quality monitoring platform, as detailed in Section 2.3. In addition, we developed a test framework outlined in Section 2.2 to construct or validate video quality metrics in an automated manner. But first, in order to get more information on the way end-users react on and tolerate visual degradations during real-life streaming, we conducted subjective tests. For this, we used a novel methodology based on full length movies, as explained next.

2.1 Real-life quality of experience assessment

As mentioned in Section 1.1, a variety of standardized subjective video quality assessment methodologies exist [14,16,15,4], specifying the way to set-up and conduct a subjective video test. They pose some stringent demands on the length of the video sequences to be displayed to the users and on the overall test duration: short video sequences must be presented to the subjects and the overall test duration should be limited to 30 minutes in order to avoid viewer fatigue. Prior to the start of such subjective tests, users also receive specific instructions on how to evaluate the individual sequences. As a consequence, subjects are actively evaluating visual quality. Since the aim of the Video Q-SAC project was to outline the problems and challenges in realizing video delivery to a home user with maximal QoS and QoE, we wanted to get more insight in the way users react on and tolerate visual degradations when watching content in their home environment.

There are significant differences between the way subjects evaluate video sequences during a standard test and while they are watching a movie at home. Firstly, when users are watching a movie they are watching for the content and are not concentrated on visual quality evaluation. Secondly, the content that is watched at home (e.g. movies or television programs) often has a duration of more than 30 minutes. And last, the home environment is a highly uncontrolled environment as opposed to the test room conditions when conducting standardized test. As a result, none of the existing methodologies can be used for performing real-life QoE assessment.

We therefore constructed a new subjective methodology based on full length DVD movies. Using this approach, the DVD can be taken home by the subjects and watched in their typical home environment. By using DVDs, we also encourage subjects to watch for the content and not to be focused on visual quality evaluation. Finally, in order to assess the influence of impairments on the perceived visual quality, we impaired the movies prior to writing them on a compliant DVD disc.

Using this methodology, two subjective tests were conducted. In [36] we conducted a test in order to assess the influence of packet loss and frame freezes on the perceived visual quality. Our results indicated the frame freezes are less noticed during full length movies. However, in the case where our subjects perceived both blockiness and frame freezes, the freezes were rated as more annoying compared to random blockiness. We conducted a similar test in [35] to investigate the influence of quality and temporal scalability on end-users QoE. This test revealed that subjects in general favor temporal scalability over quality scalability but this preference also depends on video content and on user's expectations. Our results also indicate the importance of playback fluidity and the movie flow experience. It is important to mention that the latter cannot be created using one of the standardized methodologies.

2.2 Video quality test framework

To assess the effect of network impairments such as packet loss, delay, etc., on visual quality we constructed a test framework. The basic setup comprises a streaming server (using various protocols, e.g. MPEG-TS over RTP or UDP), an impairment node and a video client. The impairment node emulates the network in a controllable way (varying random packet loss and delay, types of packets to impair). The client receives the video stream and captures it. Since each video client typically has its own error concealment techniques, it is crucial to include the client (possibly a customized one with controllable concealment techniques) in the experiment set-up. After running the streaming experiment, the captured image quality is assessed using one of the metrics outlined in Section 1.1.

Since running a single test can be time consuming (cf. streaming of video in real time), we set up a test platform to parallelize multiple video quality evaluation test. Therefore, we used the iLab.t Virtual Wall testbed (`http://ilabt.ibbt.be`), based on Emulab [13]). To configure, manage and execute video experiments in an automated manner we developed a Java management application, illustrated in Fig. 2(a).

The graph in Fig. 2(b) illustrates the significant advantage of using our scalable test framework compared to sequentially running all tests on a single set-up. (Note that the slight discrepancy between theoretical execution times and the actual measurements indicates the limited overhead of our management software to control the experiments). Results indicate a significant drop in overall experiment duration when running multiple quality evaluation tests simultaneously.

2.3 A quality monitoring platform

Within the Video Q-SAC project, we developed the monitoring platform illustrated in Fig. 3(a), which comprises three main components: (i) the monitoring probe, (ii) the management node, and (iii) the measurement archive. The latter stores historical information of both the probes and management node.

A probe monitors the network at a specific demarcation point (distributing them across the network allows for localization of quality impairments, thus detecting possible bottlenecks in the delivery chain) and measures network related metrics (packet loss, jitter) as well as video related metrics (frame rate, GOP structure, etc). To allow end-to-end Quality of Experience (QoE) estimations, a probe exchanges its monitoring information with the central management node via a web service offering access to its measurement data. We developed various probe instances with increasing complexity: some are only capable of basic network monitoring, others perform more in-depth[1] analysis of the actual video streams (e.g. extracting macroblock and motion vector information). The management node acts as a central controlling entity and (i) configures the monitoring probes, (ii) stores the monitored information and quality metrics in the measurement archive, and (iii) offers a global network view by collecting the information

[1] It must be pointed out that no full decoding is being performed by the monitoring probes.

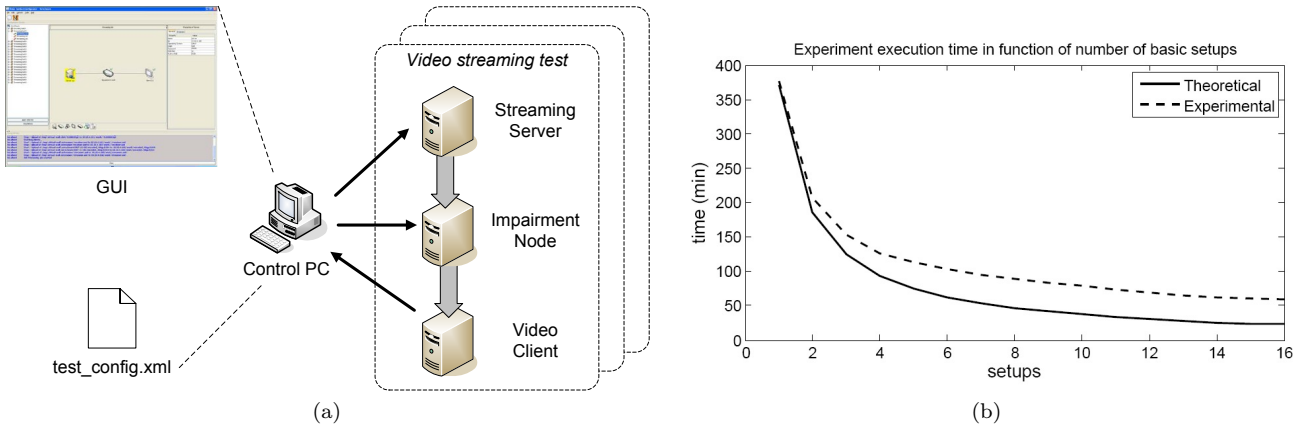


Fig. 2 Video quality test framework: (a) Video quality tests involve streaming video from a Streaming Server, through an Impairment Node, to a Video Client. Multiple instances are run in parallel on the iLab.t Virtual Wall and are controlled through Java management software using Remote Method Invocation (RMI) on the experiment machines. (b) Execution time per experiment in function of the number of simultaneous setups.

from all the available probes. The latter involves combining the probe information to derive a visual quality metric. In addition, the management node provides a topological view of the network and its probes. This is all combined in a graphical user interface (see Fig. 3(b)).

The goal of our test framework is to simplify the construction, testing and validation of existing and new objective video quality metrics. In order to demonstrate video quality monitoring during real-life streaming, the EPSNR algorithm from the Alliance for Telecommunications Industry Solutions (ATIS) was implemented in the monitor probes and the management node. Currently, we are improving and tuning the EPSNR algorithm since we found that the current implementation lacks correlation with the standard full-reference PSNR metric. Additionally, we are also looking towards the construction of a new no-reference bitstream based objective video quality metric for real-time detection of visual impairments.

3 Video coding and adaptation

As discussed above, we also realized real-time adaptation of SVC video. Before detailing the bitstream adaptation engine (BAE) in Section 3.2, we first address the issue of appropriately encoding SVC streams.

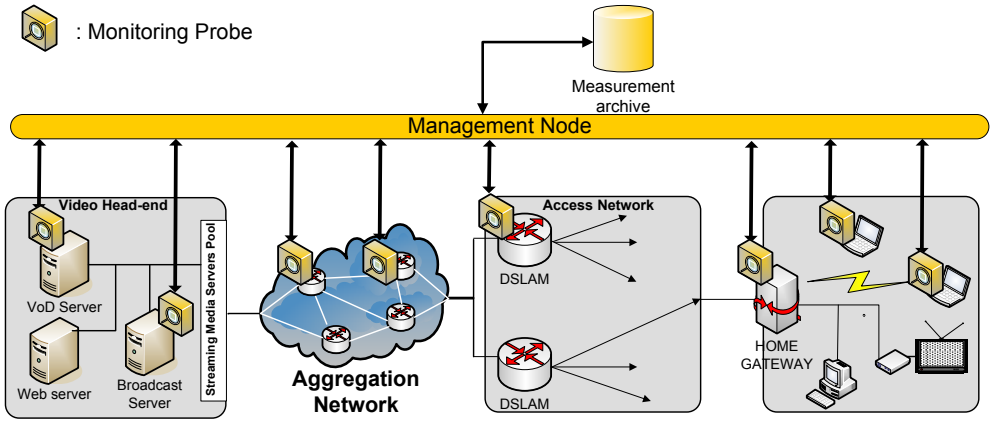
3.1 Scalable video and optimal SVC encoding configuration

3.1.1 Introduction to SVC

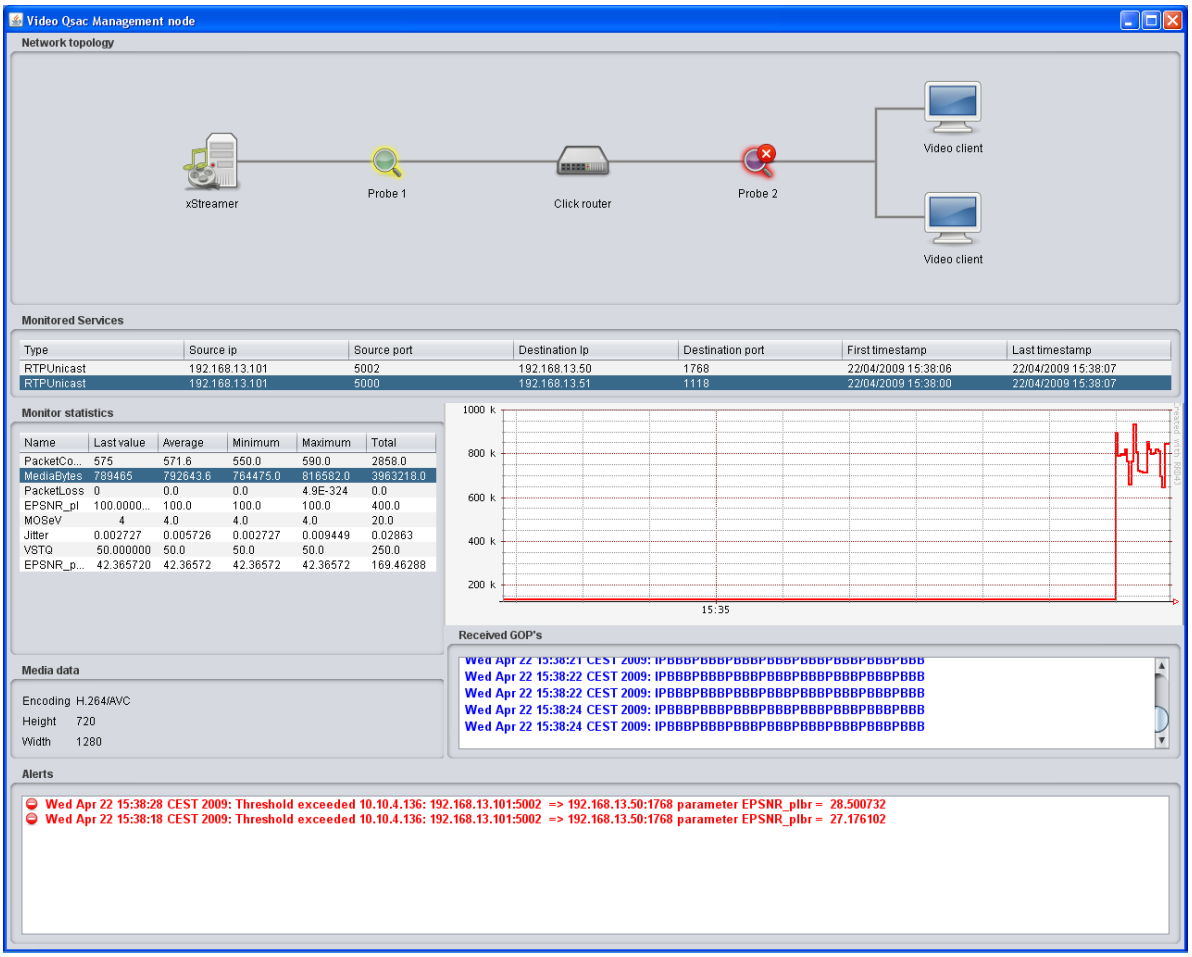
Scalable Video Coding (SVC) has been standardized in November 2007 as an extension of H.264/AVC [33, 19]. It provides a scalable coding format that, in order to facilitate market adoption, meets some important requirements: (i) similar coding efficiency for the individual layers compared to single-layer coding; (ii) little increase in decoding complexity compared to single-layer decoding; (iii) backward compatible base layer; (iv) support for temporal, spatial and quality scalability; (v) support for simple bitstream adaptations after encoding.

The three forms of scalability can be seen as three orthogonal axes. Scaling along the temporal axis results in lowering the frame rate (the temporal resolution). Spatial scaling reduces the resolution of the video in terms of pixels per frame. Finally, dropping the quality means that the properties of the video are maintained but that the fidelity of the video signal is reduced. The latter kind of scalability is enforced in SVC through the use of Medium Grain Scalability (MGS) layers or Coarse Grain Scalability (CGS) layers (with MGS allowing more flexible switching between quality layers).

The advantage of SVC is that by omitting some of the layers, the encoded stream can be tailored to target a specific device or available network bandwidth, without having to store all different versions separately. However, given the multiple scalability axes, this tailoring can occur in numerous ways and hence determining an optimal adaptation (for given target requirements) is not straightforward. This issue is addressed in Section 3.2. First, we address the issue of optimally (with respect to video quality) configuring SVC encoding in a case study on creating an HD video stream that can deal with varying network bandwidth.



(a)



(b)

Fig. 3 Video quality monitoring: (a) Monitoring framework architecture, (b) Real-time monitoring of network parameters (jitter, packet loss, etc.) with our proof-of-concept implementation.

Table 1 Choice of SVC encoding parameters for a two-layer HD stream. Note that the stated resolutions for the base layer are applicable only in the spatial scalability case (for MGS and CGS it is 1280×720).

	Bit rate	Resolution
Base layer	5, 5.5, 6, 6.5, 7 Mbit/s	704×400, 768×432, 848×480, 1024×576
Total	7.5, 8, 8.5, 9 Mbit/s	1280×720

3.1.2 Coping with fluctuating bandwidth in IPTV

We consider an IPTV use case for HD (1280×720) with varying bandwidth, and evaluate various scalability options to create a bitstream that is adaptable to varying bandwidth. We consider the frame rate to remain constant, thus leaving only spatial and quality scalability options. To gain insight in the resulting (objective) quality measures that can be achieved, we explore the results of SVC encoding with only two layers (a base layer and an enhancement layer) targeting a full stream bandwidth of 7.5-9 Mbit/s and 10-30% reduction by keeping only the base layer. The configuration parameters are summarized in Table 1, resulting in 120 different versions for each test sequence ($4 \times 5 = 20$ bit rate combinations for MGS, idem for CGS, and $4 \times 5 \times 4 = 80$ combinations for spatial scalability).

We evaluated the objective quality of the resulting video streams, both the full version and the scaled down one. For assessing the scaled down stream in case of spatial scalability, we assumed that the decoder will upsample the base layer video to the original HD resolution (resulting in lower visual quality). This upsample algorithm is assumed to be the same as the one used in the JSVM reference software of SVC.

For the encoding of the SVC video streams we used version 8.9 of the JSVM reference software (GOP size of 16 pictures, IDR period of 32, and a dyadic hierarchical coding pattern for B frames). The target bit rates were achieved using the *FixedQPEncoderStatic* tool, allowing a mismatch of 5%. In our experiments, we used seven test sequences[2]: *blue_sky*, *pedestrian_area*, *riverbed*, *rush_hour*, *station*, *sunflower*, and *tractor*.

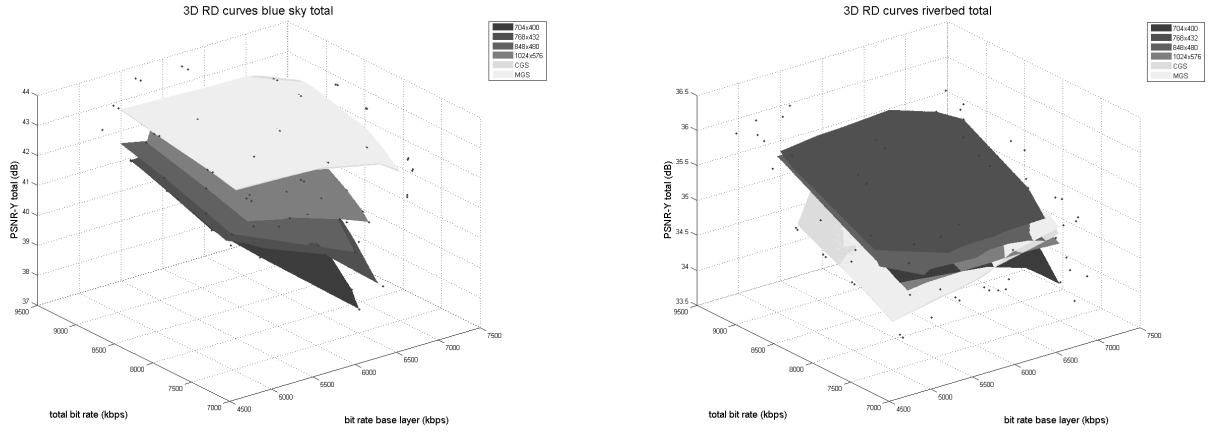
The results shown in figures 4(a) and 5(a) are representative for video sequences with low complexity. It should be noted that the 3D graph shows the quality of the total stream only, while the 2D graph shows the results for both base layer and the total bitstream when allocating 6 Mbps for the base layer. This indicates that the best quality (highest PSNR-Y) of both base layer and total stream is achieved for the highest base layer resolution, and for a lower bit rate encoding of the base layer. Comparing the quality scalability options MGS and CGS, we note they are very similar (with a slight advantage for MGS). Hence, we recommend using MGS, which allows for more flexible adaptation.

The conclusions for low complexity scenes thus are in line with SVC's design goals. Yet, for high motion complexity scenes, the above conclusions do no longer hold, as shown in the comparable Fig. 4(b) and 5(b) for the *riverbed* scene. We observe that for MGS and CGS the PSNR-Y improves considerably for increasing bit rate of the base layer (contrary to Fig. 4(a) and 5(a), where the PSNR-Y was more stable). Also, for the base layer quality, we found that spatial scalability with a base layer resolution of 768×432 achieves the best PSNR-Y: higher fidelity is achieved when encoding the sequence at a lower resolution (combined with upsampling after decoding) than when the sequence is encoded with MGS or CGS at the native resolution. This observation is clearly visible in Fig. 5(b). Also for the total PSNR-Y, this 768×432 case works best. For this optimal resolution, only a minor decrease of the PSNR-Y value can be noted as the bit rate of the base layer increases. Using other resolutions in the base layer (720p for example) causes the PSNR-Y value to increase with increasing bit rates of the base layer.

Comparing the total PSNR-Y values, that of the *riverbed* sequence (about 34-36 dB) is considerably lower than that of the *blue_sky* sequence (about 40-42 dB). This indicates that the *riverbed* sequence is significantly more complex to encode, and therefore requires more bandwidth to get the same fidelity. Therefore, we also encoded the *riverbed* sequence at a higher bandwidth, using (i) 16, 18, 19.5, 21, and 22.75 Mbit/s for the base layer; and (ii) 24, 26, 28, and 30 Mbit/s for the total bit rate (base and enhancement layer). These experiments lead to similar qualitative observations as for the lower complexity scenes like *blue_sky*: again the highest resolution (720p) achieves the best quality for the base layer alone, and the PSNR-Y value does not change as much with increasing bit rate for the base layer as it did for lower bandwidths. With respect to the quality of the entire video stream, we found that the optimal resolution was 720p (or 1024×576, depending on the bit rate configuration). Also, the PSNR-Y value now also decreases with increasing bit rate for the base layer (as in Fig. 4(a)).

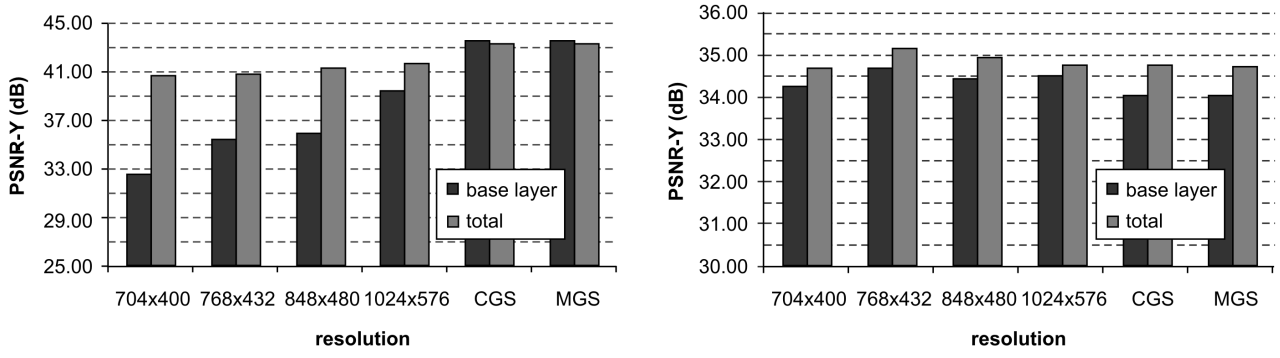
In conclusion, we can summarize that spatial scalability is only a better option than quality scalability if the total bandwidth is limited. In other words, if the available bandwidth does not allow a video stream to have a sufficient fidelity (ca. 40 dB)—which is often the

[2] Obtained from `ftp://ftp.ldv.e-technik.tu-muenchen.de/pub/test_sequences/1080p/`



(a) (b)

Fig. 4 Measured video quality of the total video stream for the (a) *blue_sky* and (b) *riverbed* test sequences. (Note: the dots represent actual measurement values).



(a) (b)

Fig. 5 Measurement results allocating 6 Mbps for the base layer, for the (a) *blue_sky* and (b) *riverbed* test sequences.

case in current HD IPTV systems—spatial scalability is preferred as it delivers a quality at both the base layer and the enhancement layer compared to quality scalability. In the other cases, SVC's quality scalability should be used. This holds true for all test sequences in our extensive test.

3.2 Adaptation of scalable video

To effectively adapt an SVC stream (e.g. keeping only the base layer of the HD stream analyzed above), we developed a bitstream Adaptation Engine (BAE). It was conceived as a node that can be placed anywhere in the network between the source and destination, adapting the video in a transparent manner. Thus, both the content provider (i.e. the streaming server) and the client will be unaware of its presence. The BAE can be triggered by the monitoring platform of Section 2 (e.g. when located close to the streaming server, under control of the provider) or by a UPnP control point when located in the home network (see Section 4).

To accomplish these goals, we have designed the adaptation node with a layered architecture, as depicted in Fig. 6. The top layer is the application layer which functions as a web service towards the other components that wish to communicate with the BAE (e.g., the home QoS management). The bottom layer is the network layer and will perform bitstream adaptations. This layer is implemented by using and implementing Click elements in the Click Modular Router platform [24]. This component will further on be referred to as the bitstream Adaptor (BA). The middle layer contains the Optimizer, whose main functionality is translating network statistics and device properties to adaptation configurations. These three components are linked to each other by the Controller that functions as a gateway between the different components.

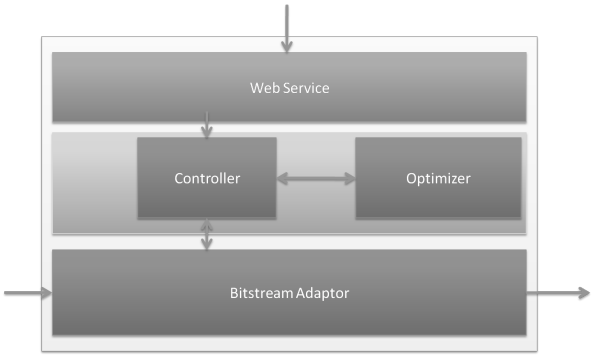


Fig. 6 Architecture of the bitstream Adaptation Engine (BAE).

3.2.1 Optimizer

Whenever an update is issued (e.g., by the QoS framework's QoSManager or Control Point) via the web service, the device and/or network descriptions are passed on to the Optimizer. The Optimizer interprets and translates these descriptions to a specific adaptation configuration which is sent to the bitstream adaptor (BA). Indeed, when multiple scalability layers (e.g. along various scalability axes) are present, various choices may be available. The streaming details of these scalability features are discussed next. We also outline how the Scalability Information SEI message can be used to gather information about the bitstream. Finally we explain the optimization process.

Scalability features of an SVC stream An SVC stream is, just like an AVC stream, constructed out of NAL (Network Adaptation Layer) units. These NALUs are decoded in order, and the result of the decoding process is the reconstructed video sequence. Unlike AVC however, each NALU has three identifiers in their NALU header, defining to which scalable layer the NALU belongs to:

(i) *dependency_id:* Increasing values for dependency_id can indicate two different scalability aspects, either a change in resolution, or a change in quality using CGS.
(ii) *quality_id:* Changes in the value for quality_id denotes a changes in quality using MGS.
(iii) *temporal_id:* Changes in the value of temporal_id denotes a change in the frame rate.

A scalable layer represents a set of coded slice NAL units with the same values of *dependency_id*, *quality_id*, and *temporal_id* and associated non-VCL NAL units. The bitstream subset that is required for decoding the scalable layer is referred to as the scalable layer representation or the representation of the scalable layer.

Scalability Information SEI message The characteristics of the complete bitstream can be extracted from the Scalability information SEI message [2]. The Scalability information SEI messages contain scalability information for subsets of the bitstream. Each of these subsets is referred to as a scalable layer and is associated with a layer identifier. Information such as bit rate, frame rate, and spatial resolution, among others, are signaled for the representation of each scalable layer as specified in [2].

SEI messages constitute additional information, and are not required to have a compliant bitstream. However, they are required for the BAE to be able to easily extract the content of the bitstream without the need to parse additional NAL Units, such as Sequence Parameter Sets and Picture Parameter Sets, or to measure the bitrates per layer at run-time.

Optimization process The purpose of this process is to find the most appropriate layer to extract from the whole set of layers in the SVC stream, fulfilling certain restrictions. Once this layer is selected, the optimizer will decide which other layers are required for decoding this most appropriate layer, e.g. it will extract the scalable layer representation for the selected layer. The decision on what layer is selected depends on:

(i) *Network bandwidth:* The optimizer will choose from the complete bitstream the optimal subset that satisfies the limitation given by the available bandwidth in the network. The first candidate for adaptation is the temporal scalability, followed by the quality scalability (in line with subjective test results mentioned earlier). However, if the temporal downscaling exceeds the values of the cases that were evaluated in the subjective tests, further downscaling is performed along the quality scalability axis. Changing the resolution is less convenient, as this requires a dynamic upsampling process at the decoder side to accommodate for the resolution difference.
(ii) *Device resolution:* The optimizer will select from the complete bitstream the highest spatial scalability layers closest to the terminal's screen resolution.
(iii) *Profile and Levels:* The optimizer will assure that the selected bitstream and the terminal's decoder match in terms of supported profile.
(iv) *User preferences:* User preferences could be taken into account to help the optimizer in case several solutions exist for the given constraints. An example of this is the choice between video quality

versus frame rate[3]: the user can decide whether to keep the quality of the video stream, and reducing the frame rate, or vice versa, keep the frame rate with a reduced quality. As such it is possible to change the default behavior of the optimizer to meet bandwidth constraints as described above.

The simplified workflow of the optimizer is as follows: (1) Mark each layer using the bandwidth constraint; (2) Mark each layer using the resolution constraint; (3) Step 1 and 2 will result in a subset of unconstrained layers; (4) Select from this subset the highest layer; this is the first candidate; (5) Check whether based on the user preference, a more appropriate layer can be found in the subset; (6) Select the layers that build up the scalable layer representation.

In step 3 and 4 it is assumed that higher bandwidths and higher resolutions represent versions of the bitstream of higher visual quality (resulting in a Pareto-optimal subset of candidate versions). In case multiple resolutions are present in the subset, preference is given to the higher resolution as long as the average fidelity at that resolution is high enough (which could be known based on the received SEI messages, also see Section 3.1.2 for the choice between quality and spatial scalability). The user preferences (step 5) can include extra constraints (which are treated as any other constrain present in the system) or they can indicate that one scalability option is preferred above another one. In the latter case, the selection in step 4 is affected by selecting the highest layer in the order of the user's indicated scalability preference.

3.2.2 bitstream Adaptor

The final step in the adaptation process is the actual adaptation of the bitstreams by the bitstream adaptor (BA). For this, the BA will first filter out RTP packets from the rest of the network traffic. Second, the RTP packets are mapped to a session by its source and destination IP address and port. Each session will contain an adaptation configuration that denotes which packets need to be dropped. This adaptation configuration is calculated by the optimizer by means of the network and/or device descriptions and the scalability information SEI message as described in the previous section. Hence, these SEI messages need to be filtered out of the network traffic as well and passed on to the Optimizer. Last, the RTP packets are parsed to retrieve the scalability information about the RTP packet currently being processed. For this, the NAL Unit header of the packet currently being handled is parsed and the three scalability syntax elements are extracted: *dependency_id*, *quality_id* and *temporal_id*. These three values are then mapped on the adaptation configuration and the packet is either forwarded or dropped, resulting in the requested adaptation.

As might be clear from this description, the BA introduces only a very limited overhead compared to a vanilla Click environment compiled in user space. Indeed, performance tests of the BA have shown that the throughput of the adaptation node is almost not affected by the BA and that the CPU usage is negligible. Also the memory usage is limited to less than 50 byte per session. Moreover, the BAE is scalable in the sense that the BA can be decoupled from other components depicted in Fig. 6 and deployed on a separate machine (introducing a small communication overhead).

[3] User studies carried out in the frame of the Video Q-SAC project showed that this preference is dependent on the end device (PC versus TV) and type of content.

4 Ensuring QoS in the home

One of the challenges in providing end-to-end QoS is that the home network is not entirely under control of the service provider or network operator. Hence, it is likely that the probes of the monitoring framework discussed above cannot be installed in the end user's home network (with the exception of the HGW, constituting the home boundary with the provider's network). Thus, there is a need for QoS mechanisms in the home network, while relieving the end user of troublesome configuration. Hence, plug-and-play functionality is desirable: this is the aim of the UPnP (Universal Plug-and-Play) Forum. In the following, we present an overview of the latest UPnP-QoS version 3, targeted at providing common interfaces to make the necessary QoS reservations. The QoS framework we propose also allows for remote configuration, thus granting a service provider access to the in-home network reservation processes.

4.1 End-to-end QoS enforcement in the home network

Recently various home network technologies emerged that support bandwidth reservation (e.g., HomePlug AV, IEEE 802.11e, HomePNA). To be able to control end-to-end Quality of Service in a heterogeneous home network comprising one or more segments of these technologies, a common framework is needed to provide an interface to actual physical properties of the network. This is exactly what the UPnP-QoS framework defines, as discussed next and provided in our proof-of-concept (PoC) implementation. It provides a way to stream video with the best quality possible while taking into account the limitations imposed by the network.

Depending on which level of QoS has been reserved for the current stream by UPnP-QoS (see Section 4.2), our Control Point (CP) triggers the bitstream Adaptor (see Section 3.2.2) on the HGW to adapt the stream if necessary (i.e. if not enough bandwidth can be reserved). It also takes into account the policies remotely configured by a Service Provider. The HGW, being the entry point of the home network, plays a central role in that it holds an interface for remote configuration (see Section 4.3) of the in-home UPnP-QoS services.

4.2 The UPnP-QoS framework

The UPnP-QoS working committee recently has finalized version 3 of the UPnP-QoS framework. Version 1 provided a framework for policy-based prioritized QoS, while version 2 extended this framework with a rotameter service to measure network performance in order to be able to diagnose and react to network problems. The latest version, version 3, introduces support for parameterized QoS and admission control. For the latter, it relies on functionality of the underlying layer-2 technology (see further, subsection 4.4.1), as opposed to e.g. proposals for explicit monitoring components in UPnP [22].

UPnP-QoS [37] defines three new UPnP [27] services, namely the QosDevice (QD), QosPolicyHolder (QPH), and QosManager (QM) service. Setting up QoS for a particular traffic stream needs to be initiated by a QoS-enabled UPnP Control Point (CP), which simultaneously can initiate the video streaming e.g. from a UPnP MediaServer to a MediaRenderer (see the UPnP AV framework [32]). Thus, the CP is aware of the source and sink (within the home network boundaries) of the stream, as well as the stream's traffic specification (TSpec, this includes e.g. bandwidth and tolerable delay). As outlined in detail in Fig. 7, the CP contacts the QM that takes care of the actual admission and set-up of end-to-end QoS in the home network. The QDs act as an abstraction layer towards the physical devices (i.e. the specific layer-2 technologies), maintain the state of the various network devices and manage the actual resources. The QPH provides an interface to a database of policies for priority and conflict resolution. It is called by the QM whenever conflicting QoS requirements arise (e.g. to preempt lower-priority streams when a new request comes in).

The predefined steps followed by the QM upon a request for QoS are depicted in Fig. 7. The QM entity first queries the network to know which QPH service to use. To learn which QDs to contact for the actual request, the QM calculates the path to follow from the path information it gathers from all QDs (either by explicitly calling *QD:GetPathInformation*, or via evented state variable information asynchronously sent by the QDs). The QM then reserves the QoS as given in the TSpec on the QDs on the calculated path. If no sufficient resources are available, the QPH is contacted, and if conflict resolution is possible by preempting other streams, their QoS reservations are released before trying once more to setup the QoS for the new stream. Note that in our implementation, when dealing with SVC streams, multiple alternative TSpecs can be considered: if the TSpec for the highest quality cannot be admitted, a lower quality's TSpec will be tried. If a lower quality TSpec is chosen, the CP will interact with the bitstream adaptation engine (BAE), as discussed in Section 3.2 (note that this is not indicated on Fig. 7).

4.3 Remote QoS management

Since UPnP is a networking technology that operates over a single subnet, UPnP devices are only controllable in the local network. To manage the QoS from outside the home networkwe need some technology that makes the UPnP QoS actions remotely accessible. This can be achieved through TR-069, a remote management protocol designed for controlling in-home devices. TR-069 manipulates a tree structure of parameters, modeling the state of the controlled device, using SOAP/HTTP. This makes the HGW remotely manageable. Modeling every UPnP-QoS action in the parameter tree, and locally calling the actions when the QoS parameters are changed trough TR-069, solves the remote management problem.

4.4 Proof-of-concept demonstration

We implemented the UPnP-QoS v3 framework, as well as the remote configuration components based on TR-069. For our implementation we used OSGi [11]: a modular component-based framework with which the lifecycle of services can be controlled. For basic UPnP functionality, such as service and device announcements and discovery, we used the Domoware UPnP driver [8]. This base driver acts as a bridge between the UPnP network and OSGi.

4.4.1 QoSDevice Framework

To address the (parameterized) QoS capabilities of a particular layer-2 technology, it needs to expose a UPnP QosDevice (QD) service (exposing state information, and a QoS enforcing interface using XML descriptions).

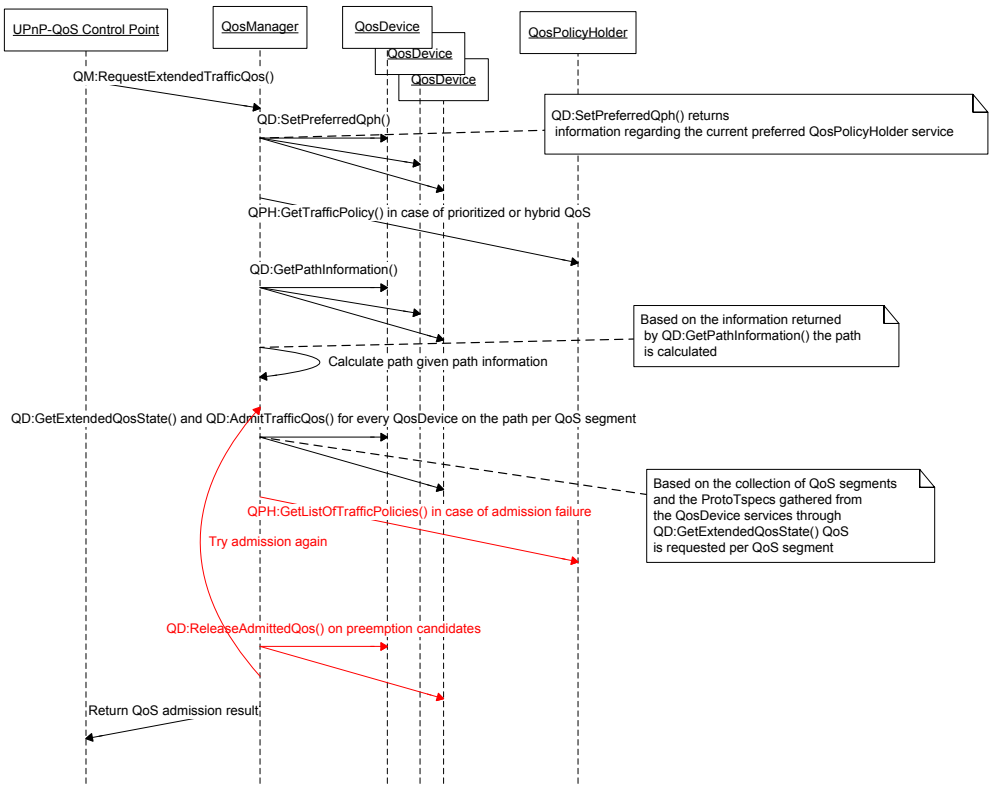


Fig. 7 Sequence diagram for a UPnP-QoS request in UPnP-QoS v3.

Since most of the QD logic is independent of the underlying network device, we developed a QD software framework that provides an easy interface between the UPnP-QoS layer and the network device, hiding all UPnP-QoS and XML. This reduces the implementation effort to deploy a new QD to implementing a few Java interfaces, gathering the network state information and translating the TSpecs in the (parameterized) QoS requests to the device's L2 capabilities. (This enabled us to provide the first demo of UPnP QoS v3 with MoCA devices[4].)

4.4.2 Demonstration scenarios

Figure 8 outlines our demo setup. The home server runs the components controlling the home network: the UPnP QosManager (QM) service, the UPnP QosPolicyHolder (QPH) service, the TR-069 component managed by the Remote Management Server (RMS) and the bitstream Adaptation Engine (BAE). The home network is divided in a wired segment (an emulator built on top of Click Modular Router [24]) and a wireless segment (an emulator built on top of NS-2 [23], with accurate modeling of the physical layer and 802.11e), each controlled by a specific UPnP QosDevice (QD) service. On this demonstrator, we successfully tested three scenarios discussed below: (i) parameterized QoS set-up, (ii) preemption, and (iii) QoS sensing.

The *parameterized QoS request* scenario tested the basic bandwidth reservation process for streaming to the laptop connected to the wireless network. The Control Point (CP, integrated with the video player) requests the QM service to make a QoS admission for the stream (see Section 4.2), thus contacting all QosDevices on the path of the stream. Once the QoS reservation is acknowledged (by the QDs to the QM, and by the QM to the CP), the video player application is notified and starts playing. We thus successfully achieved streaming with QoS, even in the presence of background traffic (that did not reserve any bandwidth).

The *preemption* use case was aimed at testing conflict resolution by the QosPolicyHolder (QPH). A first stream was granted QoS reservation. Subsequently, the security camera was activated (via motion detection), issuing its parameterized QoS request. Given the limited bandwidth, the QPH had to resolve the conflict

[4] See `http://www.lightreading.com/document.asp?doc_id=180284`

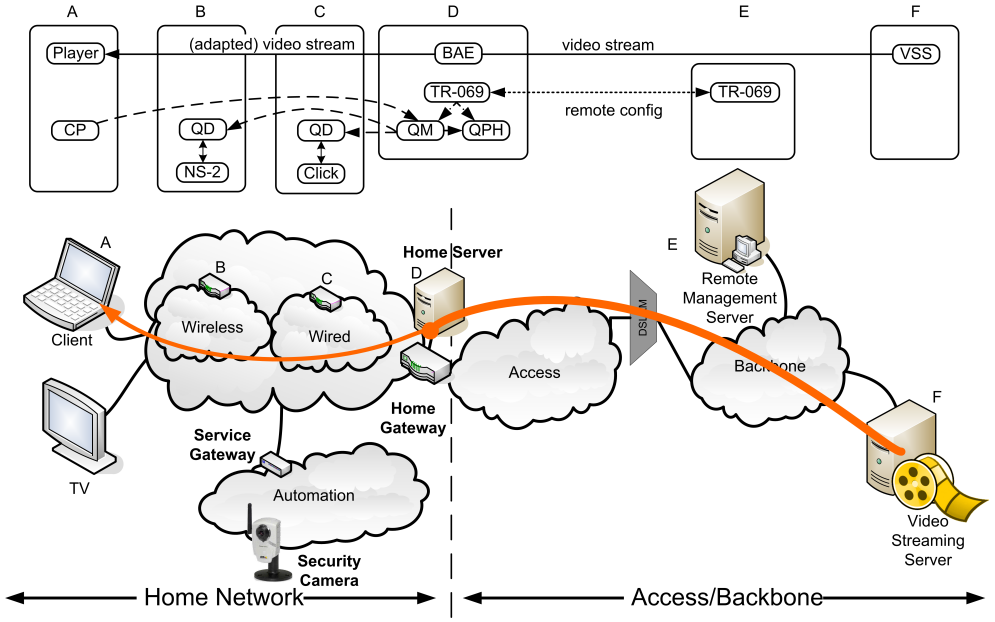


Fig. 8 Proof-of-concept demonstration setup.

in favor of the camera stream. We successfully demonstrated this feature, showing video degradation for the original stream when the security camera is activated. This also involved automatic remote configuration (TR-069) of the UPnP QPH (as well as non-UPnP configuration settings for the security camera).

The final *QoS sensing* demo illustrates the detection[5] of fluctuating network characteristics, e.g. on the wireless segment. (Since we emulate that segment, we are able to introduce a controllable bandwidth drop.) Such bandwidth drop is detected by the QD responsible for that network segment, and the QD informs the QM that admitted reservations on this QD. In our implementation, this leads the QM to contact the BAE which will scale down the affected video streams (see Section 3.2) based on the modified QoS characteristics.

4.5 Optimizations in implementing UPnP-QoS v3

Next to the functional verification, we also analyzed the performance of our PoC implementation of UPnP-QoS v3 framework. Fig. 9 shows the response time measurements of the various UPnP-QoS actions called by the QosManager on each of the QosDevices (see sequence diagram in Fig. 7). As expected, the actions involving interaction with the layer-2 access protocols (*AdmitTrafficQos* (ATQ) and *ReleaseAdmittedQos* (RAQ)) take longer. Also the times for parsing the XML data structures for *GetPathInformation* (GPI) and *GetExtendedQosState* are shown: these operations are carried out on the QoSManager (QM), and can be non-negligible for large output arguments produced by the UPnP actions (see GEQS + parsing).

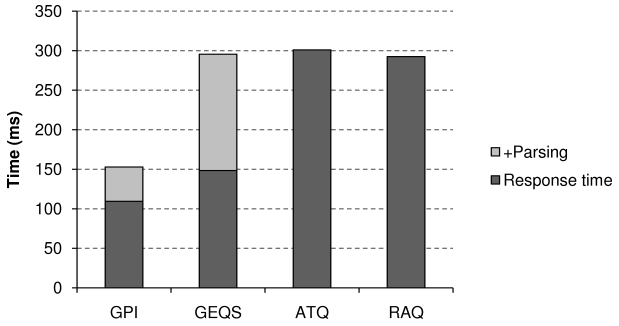


Fig. 9 Measurements of the UPnP-QoS action response times in the proof-of-concept testbed and parsing the output arguments. (GPI: *GetPathInformation*, GEQS: *GetExtendedQosState*, ATQ: *AdmitTrafficQos*, RAQ: *ReleaseAdmittedQos*.)

A basic implementation of the QM without any optimization will call all UPnP actions (described in Section 4.2) on the QD services sequentially. Say there are n_{net} QD services in the home network and n_{path} QDs on the path of a particular QoS request. The total time for a sequential implementation of the QoS request can be written as t_s (assuming successful ATQ calls) in equation 1, summing the times t_i^X for each action X over QDs i. This call of X can be subdivided in the

[5] Note that where e.g. [22] had to rely on explicit monitoring by a UPnP component, in UPnP QoS v3 the layer-2 info can directly propagate to the QoS framework via *UnexpectedStreamChange* events in the QD.

response time ($resp_i^X$) of the QD and processing the answer ($proc_i^X$) by the QM (see eq. 2).

$$t_s = \sum_{i=1}^{n_{net}} (t_i^{SPQ} + t_i^{GPI}) + \sum_{i=1}^{n_{path}} (t_i^{GEQS} + t_i^{ATQ}). \quad (1)$$

$$t_i^X = resp_i^X + proc_i^X \quad (2)$$

Assuming at least two QDs per layer-2 segment, our measurements (for a single QD) suggest that even for reasonably simple home networks (a couple of layer-2 segments) the whole procedure could already take a couple of seconds. To cope with user impatience, this can be limited by parallelizing each series of QD action X invocations. Note however that the processing of the output results (taking times $resp_i^X$) still all needs to be done by the QM. We analyzed the performance of each series of QD action calls by measuring both response (*resp*) and processing times (*proc*) in the testbed, and comparing parallel invocation times (t_p) with the basic sequential implementation (t_s). Results are summarized in Fig. 10. The relative performance improvement, the gain $1 - t_p/t_s$, is given in Table 2. This shows that by parallellizing the QD action calls, even for complex home networks (up to 20 QDs, hence in the order of 5-10 different layer-2 segments) the response times remain within an acceptable range. The decrease in response time ranges from roughly 50% to 90% for a large number (more than 5) QDs.

The performance measurements indicate that processing (esp. parsing) the PathInformation and ExtendedQosState output arguments (see GPI and GEQS) can take a relatively long time (i.e. $proc^X$ approaches $resp^X$). Thus, a further optimization is caching the PathInformation at the QM, since the QDs send this information upon changes (via evented UPnP state variables). Furthermore, if we assume that most QoS requests succeed, another optimization can be applied. Indeed, the invocations of *GetExtendedQosState* and *AdmitTrafficQos* are not entirely dependent on one another: the QM Entity can already admit QoS on one QD service before having information about another. However, these optimizations require more housekeeping in case of failure to admit Quality of Service on a particular QosDevice service. A sequential implementation would typically just release all previously admitted resources and fail without requesting the resources on the next QosDevice service in line. When any of the parallel admissions fail the QM Entity should wait for all QosDevice services to return and release each resource that has been admitted successfully before failing, and subsequently make extra release calls (RAQ)—which can again be issued in parallel—to each QD where ATQ requests succeeded. While sequential implementation will always perform a less than or equal number of ATQ invocations than the parallel implementation, the latter's performance in case of failing ATQs is however not compromised (since subsequent RAQ calls can be again issued in parallel).

5 Conclusions

In this paper, we outlined some major challenges in realizing video streaming with QoS. A primary requirement to deploy video streaming with quality guarantees is to have metrics to assess that quality. For this, we presented the current state-of-the art in objective video quality metrics. We developed a scalable testing platform to analyze the effects of network impairments on the observed quality measures. We also implemented a video quality monitoring platform. A novel subjective quality test methodology revealed users prefer temporal over quality scalability.

While a service provider can deploy such a platform in his own managed network, the home network is not under his full control. This home network is increasingly complex and heterogeneous in terms of technologies. To cope with this, we proposed a QoS framework based on UPnP and a remote management interface accessible by service providers. We successfully implemented the UPnP-QoS v3 framework and demonstrated on-the-fly adaptation of SVC video streams (triggered by the UPnP-QoS components in response to varying network conditions). We also assessed the performance of the UPnP QoS v3 framework and showed how parallellizing calls to QoS Devices improves performance compared to a naive sequential implementation.

The issues in scalable video (SVC) encoding to create an adaptable stream were addressed. In particular, our case study demonstrated that spatial scalability can achieve better visual quality than pure quality scalability under certain bandwidth constraints for scenes with high motion complexity. Additionally, we detailed the implementation of a bitstream adaptation engine capable of real-time and transparent adaptation of SVC streams. This was successfully demonstrated in a proof-of-concept setup.

Acknowledgements The authors would like to thank all partners in the IBBT Video Q-SAC project for the fruitful collaboration and discussions.

References

1. The Géant2 project. URL `http://www.geant2.net`
2. Joint draft ITU-T rec. H.264 / ISO/IEC14496-10 / amd.3 scalable video coding. Joint Video Team (JVT) of ISO/IEC

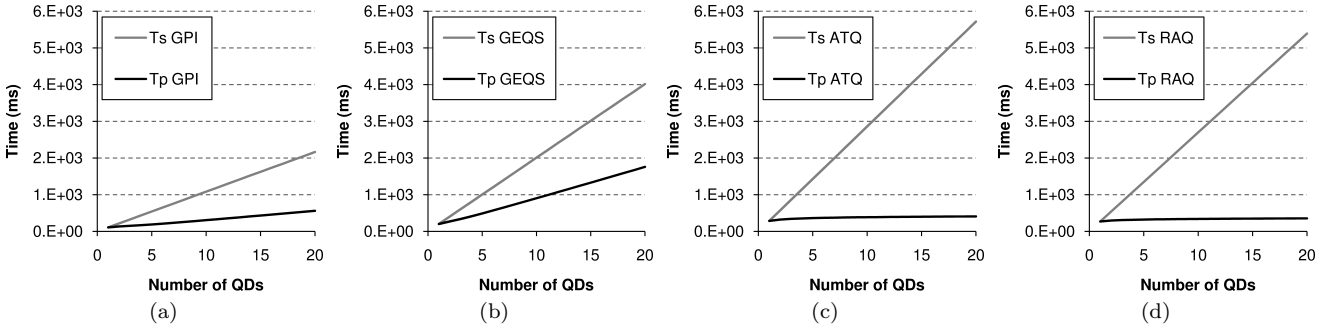


Fig. 10 Parallelization of the calls to each QoSDevice action keeps the response times of each series of calls below 1-2 seconds: (a) *GetPathInformation*, (b) *GetExtendedQosState*, (c) *AdmitTrafficQos*, (d) *ReleaseAdmittedQos*.

Table 2 The parallel execution of QosDevice action calls by a QosManager achieves considerable gain ($= 1 - t_p/t_s$).

	number of QosDevices						
	2	4	6	8	10	15	20
GEQS	32.46%	48.54%	52.46%	54.03%	54.84%	55.79%	56.10%
GPI	38.05%	60.80%	67.60%	70.32%	71.71%	73.40%	74.06%
ATQ	43.39%	69.16%	78.48%	83.41%	86.50%	90.69%	92.88%
RAQ	45.24%	70.76%	79.80%	84.52%	87.45%	91.40%	93.45%

MPEG & ITU-T VCEG, 24th meeting, Geneva, Switzerland (2007)
3. Anjali, T., Bruni, C., Iacoviello, D., Scoglio, C.: Dynamic bandwidth reservation for label switched paths: An on-line predictive approach. Comput. Commun. **29**(16), 3265–3276 (2006). DOI 10.1016/j.comcom.2006.05.004
4. Blin, J.L.: SAMVIQ – subjective assessment methodology for video quality. Tech. Rep. EBU BPN 056, EBU Project Group B/VIM (Video In Multimedia) (2003)
5. Chen, J.L., Chen, M.C., Chian, Y.R.: QoS management in heterogeneous home networks. Comput. Netw. **51**(12), 3368–79 (2007). DOI 10.1016/j.comnet.2007.01.032
6. Cooke, A., Gray, A., Ma, L., Nutt, W., Magowan, J., Oevers, M., Taylor, P., Byrom, R., Field, L., Hicks, S., Leake, J., Soni, M., Wilson, A., Cordenonsi, R., Cornwall, L., Djaoui, A., Fisher, S., Podhorszki, N., Coghlan, B., Kenny, S., O'callaghan, D.: R-GMA: An information integration system for grid monitoring. In: On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE, *LCNS*, vol. 2888/2003, pp. 462–481 (2003). DOI 10.1007/b94348
7. Coppens, J., Van den Berghe, S., Bos, H., Markatos, E., De Turck, F., Oslebo, A., Ubik, S.: Scampi: A scalable and programmable architecture for monitoring gigabit networks. In: Management of Multimedia Networks and Services, *LCNS*, vol. 2839/2003, pp. 475–487 (2003). DOI 10.1007/b13721
8. Domoware: UPnP basedriver. URL `http://sourceforge.net/projects/domoware`
9. Fang, L., Zhang, R., Taylor, M.: The evolution of carrier ethernet services - requirements and deployment case studies. IEEE Commun. Mag. **46**(3), 69–76 (2008). DOI 10.1109/MCOM.2008.4463774
10. Hachimi, M.E., Tremblay, B., Kadoch, M., Bennani, M.: Dynamic bandwidth allocation in SIP based MPLS. In: 21st IEEE Int. Conf. on Advanced Information Networking and Applications (AINA'07), pp. 917–923. Los Alamitos, CA, USA (2007). DOI 10.1109/AINA.2007.61
11. Hall, R.S., Cervantes, H.: An OSGi implementation and experience report. In: Proc. 1st IEEE Consumer Commun. and Networking Conf. (CCNC 2004), pp. 394–399 (2004)
12. Hanemann, A., Boote, J., Boyd, E., Durand, J., Kudarimoti, L., Łapacz, R., Swany, D., Trocha, S., Zurawski, J.: PerfSONAR: A service oriented architecture for multi-domain network monitoring. In: Proc. 3rd Int. Conf. Service-Oriented Computing (ICSOC 2005). Amsterdam, The Netherlands (2005). DOI 10.1007/11596141_19
13. Hibler, M., Ricci, R., Stoller, L., Duerig, J., Guruprasad, S., Stack, T., Webb, K., Lepreau, J.: Large-scale virtualization in the Emulab network testbed. In: Proc. USENIX Annual Technical Conference. Boston, MA (2008)
14. Int. Telecommun. Union: Methodology for subjective assessment of the quality of television pictures. ITU-R Recommendation BT.500-11, Geneva, Switzerland (1992)
15. Int. Telecommun. Union: Subjective audiovisual quality assessment methods for multimedia applications. ITU-R Recommendation P.911, Geneva, Switzerland (1998)
16. Int. Telecommun. Union: Subjective video quality assessment methods for multimedia applications. ITU-R Recommendation P.910, Geneva, Switzerland (1999)
17. Int. Telecommun. Union: Objective perceptual multimedia video quality measurement in the presence of a full reference. ITU-R Recommendation J.247, Geneva, Switzerland (2008)
18. Int. Telecommun. Union: Perceptual audiovisual quality measurement techniques for multimedia services over digital cable television networks in the presence of a reduced bandwidth reference. ITU-R Recommendation J.246, Geneva, Switzerland (2008)
19. ITU-T and ISO/IEC JTC 1: Advanced video coding for generic audiovisual services (2007). ISO/IEC 14496-10 AVC, version 8
20. Kanumuri, S., Cosman, P., Reibman, A.R.: A generalized linear model for MPEG-2 packet-loss visibility. In: Proc. 14th Int. Packet Video Workshop (PV2004). Irvine, CA (2004)
21. Kanumuri, S., Cosman, P.C., Reibman, A.R., Vaishampayan, V.A.: Modeling packet-loss visibility in MPEG-2 video. IEEE Trans. Multimedia **8**(2), 341–355 (2006). DOI 10.1109/TMM.2005.864343
22. Lee, H., Moon, S., Kim, J.W.: Enhanced UPnP QoS architecture for network-adaptive streaming service in home net-

works. IEEE Trans. on Consumer Electronics **53**(3), 898–904 (2007). DOI 10.1109/TCE.2007.4341563
23. Mahrenholz, D., Ivanov, S.: Real-time network emulation with ns-2. In: Proc. 8th IEEE Int. Symp. on Distributed Simulation and Real-Time Applications (DS-RT 2004), pp. 29–36. Budapest, Hungary (2004). DOI 10.1109/DS-RT.2004.34
24. Morris, R., Kohler, E., Jannotti, J., Kaashoek, M.F.: The click modular router. SIGOPS Oper. Syst. Rev. **33**(5), 217–231 (1999). DOI 10.1145/319344.319166
25. MPEG-4: Applications and requirements for Scalable Video Coding. MPEG-document ISO/IEC JTC1/SC29/WG11 N6880, Moving Picture Experts Group (MPEG), Hong Kong, China (2005). Available on `http://www.chiariglione.org/mpeg/working_documents/mpeg-04/svc/requirements.zip`
26. Newman, H.B., Legrand, I.C., Galvez, P., Voicu, R., Cirstoiu, C.: MonALISA: A distributed monitoring service architecture. In: Proc. Conf. Computing in High Energy and Nuclear Physics (CHEP03). La Jola, CA (2003)
27. Presser, A., et al.: UPnP Device Architecture (2008). URL `http://www.upnp.org/resources/documents.asp`
28. Reibman, A.R., Kanumuri, S., Vaishampayan, V., Cosman, P.C.: Visibility of individual packet losses in MPEG-2 video. In: Proc. Int. Conf. on Image Processing (ICIP '04), vol. 1, pp. 171–174 (2004). DOI 10.1109/ICIP.2004.1418717
29. Reibman, A.R., Kanumuri, S., Vaishampayan, V.A., Cosman, P.C.: Network monitoring for video quality over IP. In: Proc. Picture Coding Symposium (PCS2004). San Francisco, CA (2004)
30. Reibman, A.R., Vaishampayan, V.A., Sermadevi, Y.: Quality monitoring of video over a packet network. IEEE Trans. Multimedia **6**(2), 327–334 (2004). DOI 10.1109/TMM.2003.822785
31. Ries, M., Nemethova, O., Rupp, M.: Video quality estimation for mobile H.264/AVC video streaming. J. Communications **3**(1), 41–50 (2008)
32. Ritchie, J., Kühnel, T.: UPnP AV Architecture:1 (2002). URL `http://www.upnp.org/specs/av/default.asp`
33. Schwarz, H., Marpe, D., Wiegand, T.: Overview of the Scalable Video Coding extension of the H.264/AVC standard. IEEE Trans. Circuits Syst. Video Technol. **17**(9), 1103–1120 (2007). DOI 10.1109/TCSVT.2007.905532
34. Seepold, R., Madrid, N.M., Fernández, J.M.: QoS management for distributed multimedia services. In: Proc. 10th IFIP/IEEE Int. Conf. on Management of Multimedia and Mobile Networks and Services (MMNS 2007). San Jose, CA (2007). DOI 10.1007/978-3-540-75869-3_17
35. Staelens, N., Moens, S., Van den Broeck, W., Marien, I., Vermeulen, B., Lambert, P., Van de Walle, R., Demeester, P.: Assessing the perceptual influence of h.264/svc signal-to-noise ratio and temporal scalability on full length movies. In: Proc. Int. Workshop on Quality of Multimedia Experience (QoMEx 2009), pp. 29–34. San Diego, CA, USA (2009). DOI 10.1109/QOMEX.2009.5246982
36. Staelens, N., Vermeulen, B., Moens, S., Macq, J.F., Lambert, P., Van de Walle, R., Demeester, P.: Assessing the influence of packet loss and frame freezes on the perceptual quality of full length movies. In: Proc. 4th Int. Workshop on Video Processing and Quality Metrics for Consumer Electronics (VPQM 2009). Scottsdale, AZ, USA (2009)
37. UPnP Forum: UPnP QoS. URL `http://www.upnp.org/specs/qos`
38. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. IEEE Trans. on Image Processing **13**(4), 600–612 (2004). DOI 10.1109/TIP.2003.819861
39. Wien, M., Schwarz, H., Oelbaum, T.: Performance analysis of svc. IEEE Trans. Circuits Syst. Video Technol. **17**(9), 1194–1203 (2007). DOI 10.1109/TCSVT.2007.905530