Caching of Interactive Multiple Choice MPEG-4 Presentations

Carsten Griwodz, Frank T. Johnsen[1], Simen Rekkedal, Pål Halvorsen
Department of Informatics
University of Oslo
Postbox 1080 Blindern
N-0316 Oslo, Norway

Abstract

On-demand access to remote multimedia content via the Internet is becoming increasingly popular. Applications like Video on Demand and News on Demand are increasingly based on structured multimedia presentations, which give users more options and freedom to interact with the content than just VCR-like operations. For example, authors provide multiple choices for parts of the presentation and the user can select the most interesting one. Distributing such presentations in hierarchical distribution systems comprised of origin server and proxies can lead to new challenges for proxies since entire presentations are large but only smalls parts of them are actually consumed by each user. We have analyzed this problem for interactive presentations encoded in MPEG-4 and found that we can use the internal structure of MPEG-4 objects to improve performance in a distribution system based on proxy caching.

1 Introduction

On-demand access to remote multimedia content via the Internet is becoming increasingly popular. One of the main reasons for this trend is the technological development. End-user machines are sufficiently powerful and the bandwidth on the last mile from the content provider to the end-users is continuously increasing such that sufficient resources for high quality presentations are available. Video on Demand (VoD) [10] is an example of one such popular application and News on Demand (NoD) [7] is another. While these applications are today either presentations with linear timelines or hypertext presentations connecting such linear pieces, digital media enable authors to create much more flexible presentations. In this paper we consider the use of hypermedia within a single presentation object. Instead of having one monolithic presentation object, a presentation object can be structured into several parts, which we call *chapters* throughout this paper. VoD and other media used for entertainment purposes can be structured into chapters in a multimedia presentation [10].

Streaming media applications have varying degrees of interactivity, ranging from very high interactivity in the case of Learning on Demand, to medium and low interactivity for the previously mentioned NoD and VoD, respectively. Interactivity introduces unpredictability into on-demand systems, and the greater the degree of interactivity, the greater the amount of unpredictability will be. An investigation on the interactivity of various on-demand applications can be found in [12]. The performance of many distributed system ideas suffers when users interact with the presentation. Simple VCR-like functions need special consideration for the origin servers in VoD systems, and likewise for the intermediate proxy. If further interactive patterns are available, such as in [1, 13] and [15], the objects that are referenced by the links in the presentation must be considered separately by the on-demand system and proxy.

Our goal is to investigate the effects that varying cache granularity has on byte hit ratio when caching interactive multimedia presentations. In this paper we focus on applications with medium to low interactivity, and how such applications can benefit from proxy caching. Here, we consider exclusively MPEG-4, and how knowledge of this format can be used to perform efficient caching in a proxy server. The contribution of this paper is an analysis of MPEG-4, and how a minimal understanding about the stream at a proxy needs can be exploited to increase the byte hit ratio when caching VoD or NoD presentations encoded in that format. We present a series of simulations showing the effectiveness of caching based on this knowledge.

The rest of the paper is structured as follows:

[1]Frank T. Johnsen's contribution was made in the context of the INSTANCE II project, which is funded by the Norwegian Research Council's IKT-2010 Program, Contract No. 147426/431

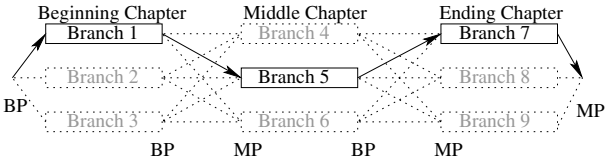


Figure 1: Caching of multiple choices

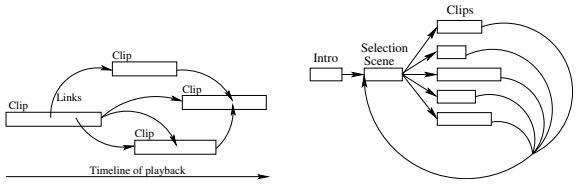
In section 2, we describe caching of multiple choice presentations encoded as MPEG-4, and derive the knowledge the proxy needs to be able to perform the caching. In section 3, we present our evaluation results of the caching scheme, finally we conclude this paper with an outlook to future work in section 4.

2 Caching of multiple choices

In this section, we explain in detail how knowledge about the encoding can enable a proxy to cache interactive streams more efficiently. First we briefly describe parts of the ISO-14496-1 MPEG-4 systems encoding structure [5] and give an overview of how our scheme works. We emphasize on the elements which are important for caching interactive streams with multiple choices, followed by a description of the metadata structure for tracking cache object replacements in proxy caches. Then we describe the mechanism for such caching in detail.

2.1 Branching

The key idea is that the proxy exploits the structure of requested presentations, recognizing the alternative choices, i.e. branches, for each chapter and keeps track of their popularity. Figure 1 illustrates a sample presentation structure with three chapters and for each chapter the user gets three choices. The users can tailor the presentation to their particular needs and interests. The example in Figure 1 illustrates that a user selected Branch 1 for the beginning chapter, Branch 5 for the middle chapter, and Branch 7 for the ending chapter. At each branching- and merging point (BP/MP), the user is offered interactive choices. Navigating through a presentation from beginning to end traces a *path* through it.



(a) Hypervideo (b) Cyclic Timeline

This presentation structure is only one specific type of interactive multimedia, and the only one that we use for the experiments described in this paper. Basically we assume that the only possibility for choosing a different playback point is at predetermined branching points, and that any such alternative object is completed before the merging point. More general structures can of course be used to implement the same type of presentation, see Figures 2(a) and 2(b). Both hypermedia presentations and presentations that include cyclic timelines provide basically the same amount of information about individual branches as the simple branching example. However, access probabilities for individual branches may vary more widely.

2.2 MPEG-4

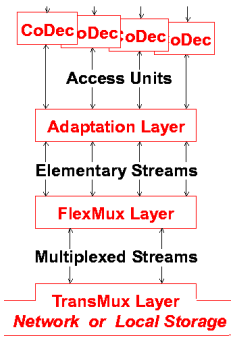


Figure 2: MPEG-4 layers [11]

To understand the amount of knowledge that a proxy needs for considering branches for caching, we look in more detail at our encoding format of choice, MPEG-4. Functionality in MPEG-4 is organized in layers. The relation of the various layers is shown in Figure 2. Coding of data is done by the codec, which produces access units (AUs). The AUs will, through an adaptation layer, form elementary streams (ESs). The ESs can then be multiplexed into streams through the FlexMux layer and then be either stored locally or transferred over the network via the TransMux layer. They are thoroughly covered in the full ISO-14496 standard, and further details are beyond the scope of this paper. However, the implications of the layers for caching can be briefly described as follows:

- Between the TransMux layer and the FlexMux layer, the smallest visible unit is the complete stream, and caching at this level leaves little flexibility. For example, adding subtitles to or removing them from a movie results in an ES being either added to or removed from the multiplexed stream, thus appearing as two completely different sets of data to a proxy.

- Between the FlexMux layer and the Adaptation layer, the proxy can consider the ESs themselves. Each ES can encode one piece of information, be it a video, an audio clip, a subtitle or similar. Thus, we loose some relations among strongly related information such as audio and video streams that are to be played backed concurrently, or video streams that are to be played back-to-back.

- Above the Adaptation layer, we can consider individual AUs. This adds complexity but not usability. Since AUs provide a granularity of individual frames or audio samples, they are highly flexible and can be used to efficiently cache bits of an ES that is frequently cancelled by users after playing only part of it back. However, considering separate AUs leads to complex book-keeping of cache objects without adding flexibility – an AU is meaningless on its own. It needs to be combined with other AUs into an ES and then further multiplexed for a client to be able to view the presentation.

So, considering the structure of presentations as described in [10], it suffices to consider the ESs for caching. We do therefore focus on the ESs in the following, which provides sufficient flexibility for caching in a proxy server while still keeping the cache object granularity to minimum and limiting cache maintenance metadata overhead. We will also see that partial caching does not conflict with making caching decisions at ES granularity.

If a branched video is encoded in MPEG-4, we can not trivially assume that every branch is encoded in one video and one audio object each. It is possible that one medium of a branch is encoded in several ESs, and it is also conceivable that a branch is encoded as temporal subrange of a larger ES. However, considering coding and addressing efficiency, it appears most convenient for editing tools that can be used to create branched videos to employ ESs or sequences of ESs to build a graph. It is then always at the end of an ES that the user can choose between multiple branches for the next chapter. For each chapter, there are one or more alternative branches that can be selected by the user. One branch includes all the media objects necessary to play that portion of the presentation, for instance a video ES an audio ES and an animation ES.

2.3 Proxy design

It does then become an issue to efficiently identify ESs and keep metadata about them. Following the discussion above, the proxy cache must be able

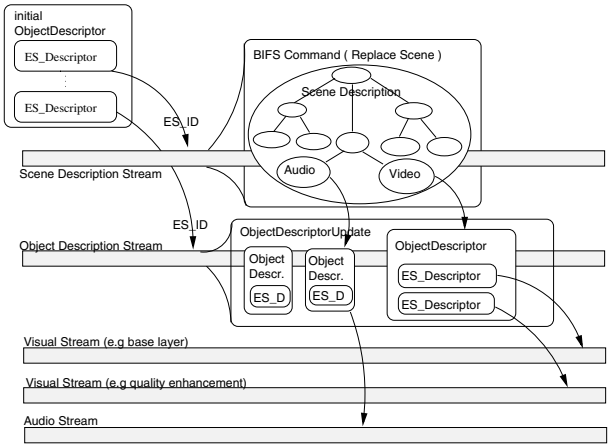


Figure 3: Elementary Streams and the Object Descriptors [5]

to identify the ESs that the MPEG-4 presentation consists of to perform caching effectively. In our design, the proxy intercepts requests from the user to the source server, and content going from the source server to the users.

Under the assumption made in the previous section, the proxy is able to recognize the ESs and keep track of branches' popularity per presentation. For every presentation that is requested through the proxy, the proxy tracks the popularity of every ESs of that presentation. All data that a client might request is encapsulated in ESs. The proxy can look up the requested ES in the cache and serve it to the client if it is present. With all media content stored in identifiable ESs, the proxy can keep metadata attributed to every ES in the cache. This can be used to count accesses, enabling the use of a suitable cache replacement algorithm.

In our approach, we see no need for the proxy to have further knowledge of the structure to assure that the alternative branches of a chapter are handled correctly, i.e. the proxy does not need to know which ESs belong to the same chapter and are an alternative to each other.

Figure 3 shows that locating ESs in a data stream does not require the proxy to implement the MPEG-4 standard in great detail, if the MPEG-4 Initial Object Descriptor (IOD) of a presentation contains a complete list of all the ESs that the presentation is made of, and there are no external references. According to the MPEG-4 14496-1 Standard, the IOD may or may not contain such a complete list, depending on the author of the media. Any ES that does appear within the IOD is said to be in the same name scope.

Parsing of the meta descriptors provided by MPEG-4 does then allow the proxy to locate the individual ESs is as follows:

> The proxy recognizes the ES, first by getting hold of the IOD. Next, the ES is packetized in the synchronization layer as a set of sequential Access Units (AUs), and the beginning of an ES is marked in the first AU of that ES. The type of ES may be a scene description stream, object description stream, or a media stream such as video or audio. The object descriptor stream provides a set of ES descriptors for the ES, which determines the types and all information needed for decoding, digital rights management and resource allocation.

Each ES has a unique identifier within the name scope of one IOD. In the scene description stream the layout of the presentation is coded using a set of descriptors. These descriptors provide the possibility of more general interactive presentations. In our proxy design, a presentation's IOD is parsed first, and all ESs of a presentation are initially considered as caching candidates, meaning that the proxy stores metadata about all the ESs that the presentation consists of. This provides the proxy with a handle for all metadata that it needs to track the popularity of each ES in a presentation. For every ES requested by the user, the proxy notes an access for that ES in the metadata structure, and retrieves the ES from the source server and serves it to the user. If there is room in the cache, the proxy stores the ES, and tags this ES as local in the cache content list. Subsequent requests may be for ESs that are already in the cache. The proxy searches through the cache content list, and serves the ES from cache when possible, noting a cache hit by updating the access counter for that ES. When the cache is full, the new caching candidates must contend for space with the ones already there. Note that this way of caching MPEG-4 streams, which we call ES-caching, works best when the presentations have multiple choices. It gracefully reduces to full object caching if all the presentations are monolithic in structure.

3 Simulations and results

We evaluate the benefits of ES-caching over full object caching by simulation. It consists of a three-step hierarchy consisting of clients, proxy and server. Since we focus on the caching efficiency, the limiting factor in our simulation is exclusively the size of the cache. We assume that the network connecting them is perfect, i.e. there is no congestion, loss or latency. The origin server has all the interactive MPEG-4 presentations. We assume that all branches are of equal size and consist of two ESs, audio and video, respectively. We have a total of 1000 unique presentations, with 100 000 requests for starting a new presentation. For each presentation that is started, the client navigates through one possible path of the multitude of choices. At each branching point the client sends an interaction message to the proxy. The internal structure of each presentation various among experiments.

To choose among the available presentations, every client makes a independent choice every time it requests a content. The choice is made according to the Zipf distribution, which is representative for the popularity of the interactive multimedia streams in both Video on Demand [3] and News on Demand [7] scenarios. Thus, we use the Zipf distribution in all of our experiments to choose the probability for selecting a particular presentation in our simulations. Following the request for a presentation there will be subsequent requests for chapters. This request pattern within a presentation, i.e. the tracing of a path through the presentation, is varied in our experiments. In the experiments we ignore the temporal development of popularities that we advocated in [3]. We can make this simplification only because we can assume that the large number of requests that we simulate arrives on a timescale that is considerably smaller than the interval between the insertions of new presentations into the origin server. If the difference of timescales is large, the re-ordering of popularities will only very sporadically be the reason for a miss and thus, have no visible effect on the hit ratio. In both, NoD and VoD scenarios, the installation of cache servers is viable only when the number of clients is so big that this is not a problem.

3.1 Metric

The metric for evaluation of ES-caching is the byte hit ratio that can be achieved for various configurations of interactive MPEG-4 content. We choose byte hit ratio as the only performance metric because it gives an indication of performance gains that is independent of the characteristics of the network. It provides a general idea of the efficiency one can expect from this scheme, rather than measuring latency improvements that would be valid for just one particular network configuration. Below, we present further details and assumptions for the simulations, followed by a presentation and discussion of some of the most important results.

3.2 Replacement Strategy

For the simulations presented in this paper we have chosen LRU with a minor variation since it is a good all-purpose cache replacement algorithm often used in proxies [8]. The ESs are cached in their entirety, and likewise replaced entirely. The metadata, such as the scene descriptors and object descriptors are small compared to the media data and is kept in the proxy cache forever. We do not need a more elaborate algorithm than LRU, since we are only trying to establish the effect varying cache object granularity has on byte hit ratio. For our simulations we assume static content; transfer of live streams cannot be enhanced by our scheme, since they are transmitted only once. For more considerations on caching streaming media objects, see [2].

The proxy implements the ES-caching algorithm and tracks accesses to ESs of MPEG-4 content, without realizing that they can be the branches of an interactive video. Due to the nature of the scheme, it will also work for variable size branches and different numbers of ESs. Thus, our results are still valid even under these assumptions.

3.3 Experiment overview

In this paper, we present three experiments: in figure 4, we illustrate how different probabilities for following paths through a presentation affect the performance of the scheme; in figure 5, we show the impact of increased interactive choices in relation to the byte hit ratio; finally, in figure 6, we show how ES-caching performs when it is combined with quality partial [9] caching. Throughout the simulations we vary the following parameters:

- The number of interactive choices in each presentation is varied to see how increasing interactivity granularity can affect the byte hit ratio.
- The cache size is varied to see how well ES-caching performs when storage space is scarce.
- Selection patterns for the interactive choices within each presentation are varied also, as this is important to illustrate performance variances that may arise from different ways the users may interact with the material.

Variable branch popularity

Figure 4 illustrates the linear access pattern for our scenario, in which the presentations are requested according to the Zipf distribution, but the way in which interactive choices are made, i.e. the branches are accessed, is varied. In the case of cyclic branches, the

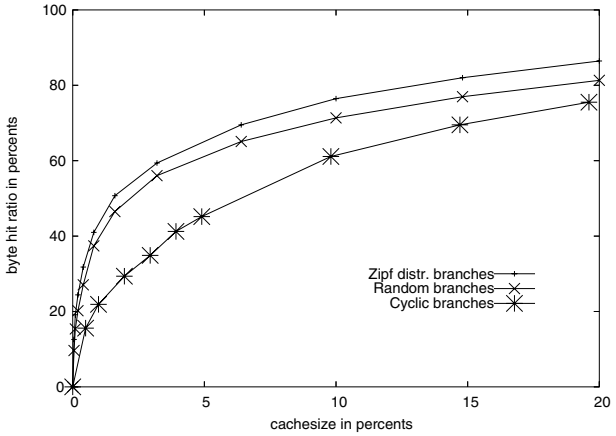


Figure 4: Variable branch popularity

branches are accessed repeatedly in consecutive order. This implies that every ES has exactly the same probability of being accessed as all other ESs in the same presentation, and that branches of the same chapter are accessed in a cycle. The number of branches within a chapter as well as the number of chapters in the figure is 3.

The other two approaches shown fix the probability of being accessed for branch beforehand. In the random branches case, the popularities within a chapter are distributed randomly, in the Zipf case, the popularities of branches within a chapter are chosen according to the Zipf distribution. The latter does obviously favour individually branches in a chapter heavily, providing popular branches from unpopular presentations with a high chance of staying in the cache in favour of unpopular branches of popular presentations. The effect that this has on the overall hit ratio can be seen clearly. The random popularity assignment has a similar, but weaker effect; some branches are more popular than other, but the distribution of popularities is uniform. Thus it is less likely for a branch to be cached in favour of a more popular title from an unpopular presentation.

Number of Multiple Choices

In figure 5 we illustrate the effect that granularity has on the effectiveness of the ES-caching scheme. We vary the number of branches within each chapter as well as the number of chapters (and thereby the number of interaction points). The popularity assignment for the branches is the Zipf-distribution approach from the first experiment.

We investigate the impact varying the number of interactive choices has on the byte hit ratio. The overall trend is that increasing the number of branches,

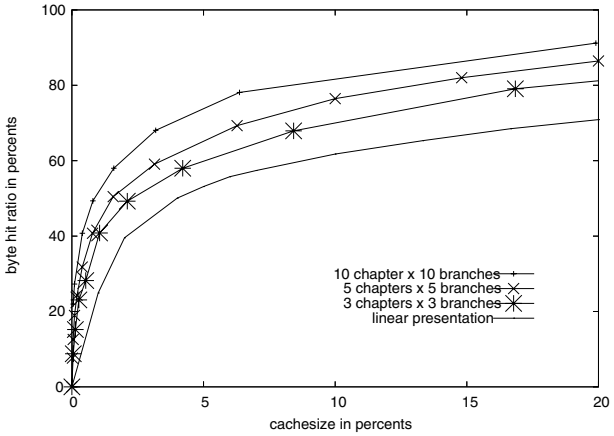


Figure 5: Number of Multiple Choices

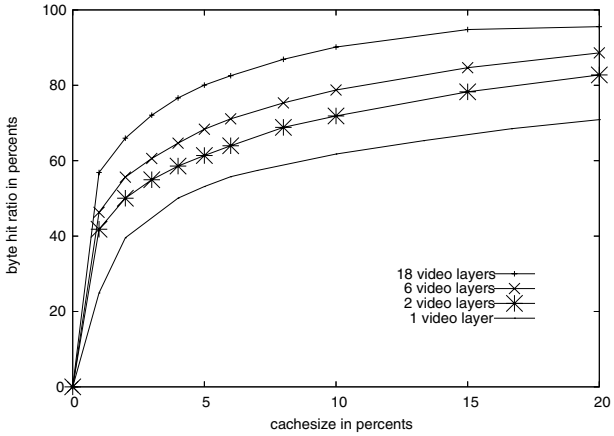


Figure 6: Combination with partial caching

and thus the number of objects that are eligible for caching, has a positive impact on the byte hit ratio. Also, by increasing the number of branches and keeping the total size of the presentation constant, the overall branch size decreases, leading to finer granularity of the objects eligible for cache replacement. This is where ES-caching shows its strength by efficiently handling presentations with a large degree of interaction.

Combination with quality partial caching

In this experiment, we have combined ES-caching with layering of content, which is one of various approaches for quality partial caching ([14, 16, 6]), in contrast to the more typical temporal partial caching. Quality partial caching does actual fit well with the selection of MPEG-4 ESs as caching granularity as well. Quality partial caching depends on structured content where some parts provide an optional quality enhancement to a mandatory part that provides only basic quality. The mandatory part is frequently called the base layer, the optional parts are called enhancement layers. Since MPEG-4 support the idea of optional enhancement through the addition of optional ESs, it is appropriate to encode each quality layer in its own ES.

Our experiment shows the orthogonality of ES-caching to that of quality partial caching. The combination shown in figure 6 is based on the assumption that the decision for retrieving layers is made by clients for there own reasons, not by the proxy. This is a major difference from, e.g. the temporal caching approach prefix caching, where startup latency is minimized by actively replicating the initial portions of content to the proxy.

We see in figure 6 that applying quality partial caching to the scheme yields an increase in byte hit ratio when caching only lower layers, and that this further raises as one increases the number of layers. This is due to the fact that an increase in layers for a branch when caching only the lower layer will mean that a smaller part of the entire stream is cached, thus effectively leaving room for even more of the low layers of other branches in the cache. Having more layers will lead to more signalling overhead, as each layer cache miss will require the corresponding data to be transmitted from the origin server. Our main concern here is the byte hit ratio, though. The requests for quality will for each user be for a random amount of interdependent ESs, to a maximum of 6 layers, and a minimum of 2. The lowest graph, one video layer, is the reference graph of a monolithic movie with no branching and no layering.

3.4 Further considerations

The presentation structure we have assumed so far and based the implementation of ES-caching on, is only one specific type of interactive multimedia. If the proxy can not assume that an ES is a complete object, the granularity of the cache object will be different than the granularity of the interactivity, which will lead to inefficiency at best, or faulty service at worst. In this case, the proxy would have to keep a score of how big a portion of the ES is used, and cache only that part.

Also, if other formats, for instance MHEG, Hytime, Quicktime, SVG or SMIL are used, the proxy needs the assurance that the elementary building blocks are referable through meta-descriptors, and can be stored on their own as cache objects. If these building blocks cannot be referred to through meta-descriptors, the proxy must parse the data looking for branching points

and merging points. This requires that the proxy has knowledge about the encoding of the media data. In MPEG-4 the proxy can be encoding-format agnostic, since the MPEG-4 meta-descriptors already have done the job.

4 Conclusion and future work

We have presented a new caching technique for interactive MPEG-4 multimedia presentations which have multiple choices in selection scenes. This technique is ES-caching, since knowledge of the presentations' structure, i.e. ESs, is utilized by the scheme to cache various interactivity choices structured in chapters as branches. The branches are implemented as a set of ESs in MPEG-4. ES-caching works by keeping the most popular ESs in the cache. Our experiments show that interactive MPEG-4 presentations benefit from the scheme, and that the gains are proportional to the degree of interactivity. ES-caching works well both alone and also in conjunction with orthogonal partial caching schemes like quality partial caching. By using this in addition to pure ES-caching, we can further increase the overall byte hit ratio.

Future extensions to the ES-caching scheme include quality adaptation [14] for different user terminals and fine grained prefix caching [4] of each ES, for more general types of interactive presentations. For instance, featuring reuse of content from other presentations, and jumping into and out of ESs at other points than their beginning and end. Along with these variations of the content that is eligible for caching and the differentiation of the ESs' roles, we will investigate more complex caching strategies that make more efficient use of the eternal caching history of partial objects.

References

[1] Glorianna Davenport, Thomas Aguierre Smith, and Natalio Pincever. Cinematic primitives for multimedia. *IEEE Computer Graphics and Applications, vol. 11 issue 4, pg. 67 - 74.*, 1997.

[2] Carsten Griwodz. *Wide-area True Video-on-Demand by a Decentralized Cache-based Distribution Infrastructure.* PhD thesis, Darmstadt University of Technology, Darmstadt, Germany, April 2000.

[3] Carsten Griwodz, Michael Bar, and Lars C. Wolf. Long-term movie popularity models in video-on-demand systems: or the life of an on-demand movie. In *Proceedings of the fifth ACM international conference on Multimedia*, pages 349–357. ACM Press, 1997.

[4] Stephane Gruber, Jennifer Rexford, and Andrea Basso. Design considerations for an rtsp-based prefix-caching proxy for multimedia streams. *Tech. Rep. 990907-01, AT&T Labs Research*, 1999.

[5] ISO/IEC JTC1/SC29. Information Technology – Coding of Audio - Visual Objects – Part 1: Systems ISO/IEC 14496-1 International Standard, 2001.

[6] Jussi Kangasharju, Felix Hartanto, Martin Reisslein, and Keith W. Ross. Distributing Layered Encoded Video through Caches. *IEEE Transactions on Computers*, 51(6):622–636, 2002.

[7] Espen Nilsen. Analysis of News-On-Demand Characteristics and Client Access Patterns. Master's thesis, University of Oslo, April 2005.

[8] M. Rabinovich and O. Spatscheck. *WEB Caching and Replication.* Addison & Wesley, 2002.

[9] Reza Rejaie, Haobo Yu, Mark Handley, and Deborah Estrin. Multimedia proxy caching for quality adaptive streaming applications in the Internet. In *Proceedings of the Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, pages 980–989, Tel-Aviv, Israel, March 2000.

[10] Simen Rekkedal. Caching of Interactive Branching Video in MPEG-4. Master's thesis, University of Oslo, July 2004.

[11] ISO Rob Koenen. Overview of the MPEG-4 Standard. *http://mpeg.telecomitalialab.com /standards/mpeg-4/mpeg-4.htm*, 2002.

[12] Marcus Rocha, Marcelo Maia, Italo Cunha, Jussara Almeida, and Srgio Campos. Scalable media streaming to interactive users. *ACM MM November 6-12, 2005, Singapore.*, ACM 1-59593-044-2/05/0011.

[13] Nitin Sawhney, David Balcom, and Ian Smith. Hypercafe. *Seventh ACM Conference on Hypertext*, 1996.

[14] Peter Schojer, Laszlo Boszormenyi, Hermann Hellwagner, Bernhard Penz, and Stefan Podlipnig. Architecture of a quality based intelligent proxy (QBIX) for mpeg-4 videos. *ACM 1-58113-680-3/03/0005*, mar 2003.

[15] Guy Vardi. Navigation scheme for interactive movies with linear narrative. In *Proceedings of the tenth ACM Conference on Hypertext and hypermedia : returning to our diverse roots*, pages 131–132. ACM Press, 1999.

[16] Michael Zink, Oliver Künzel, Jens Schmitt, and Ralf Steinmetz. Subjective impression of variations in layer encoded videos. In *Proceedings of the International Workshop on Quality of Service (IWQoS)*, Monterey, CA, USA, June 1993.