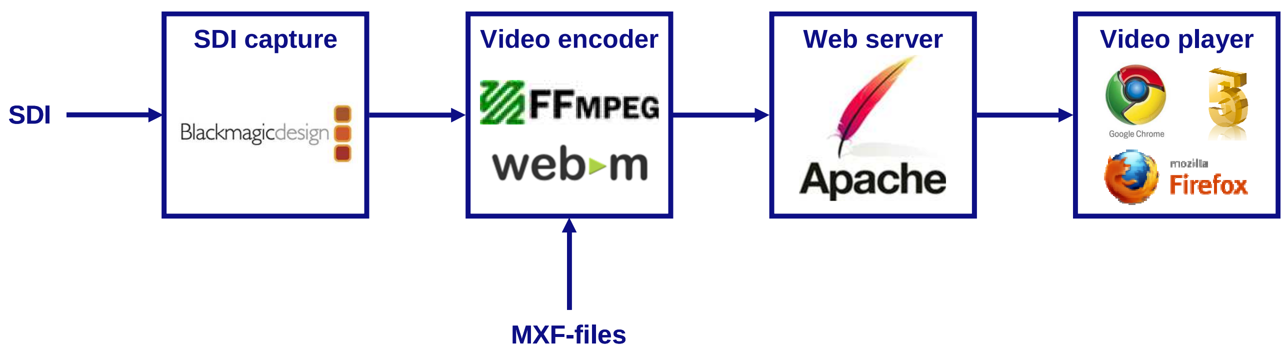
WebM as an alternative to H.264?

Investigation of the usage of open
source software and open standards



Co-operation of

CodeShop

Authors
Mark van Staalduinen
Martin Prins

TNO report

Technical Sciences
Brassersplein 2
2612 CT Delft
P.O. Box 5050
2600 GB Delft
The Netherlands

www.tno.nl

T +31 88 866 70 00
F +31 88 866 70 57
infodesk@tno.nl

WebM as an alternative to H.264?

Investigation of the usage of open source software and open standards

Date	August 11, 2011
Authors	Dr. Ir. Mark van Staalduinen, Martin Prins MSc
Reviewers	Ir. Kamal Ahmed Ir. Frits Klok Drs. Dirk Griffioen (CodeShop) Bram Tullemans (EBU) Dr. Ir. Egon Verharen (NPO)
Number of pages	35 (incl. appendices)
Number of appendices	3
Sponsors	Bram Tullemans (EBU), Egon Verharen (NPO), Jean-Louis Roso (TNO)
Project name	WebM Pilot
Project number	035.33967
Report number	35617

Preface

The introduction of WebM aroused more than a normal amount of interest at the Dutch public broadcast organization NPO. We were excited by both the open video format for web publications and the impressive VP8 codec performance at low bitrates. The expectations were high and we were curious if WebM could meet our standards and perhaps even more important, if it would also suit a broadcast production environment. We were tasked to investigate the consequences of the implementation of WebM for a broadcaster. The TNO report presented here is the result.

The questions we addressed are typically from a broadcasters perspective, for instance:
- Can live video be encoded from a high resolution SDI source to WebM?
- Can WebM be used effectively to publish video online?
- Is the codec fast and qualitative enough to compete with H.264 for both on-demand and live video?
- Can it be implemented in an automated media distribution workflow?
- Does it not consume more energy for encoding and distribution?
- Or in other words: Is WebM ready to be used on a big scale by the NPO?

The conclusion at this moment is that WebM is not ready for a large scale broadcast implementation. This is especially the case, given the quality of the online video at high speed encoding and the necessary tools to process the video format. At the time of NPO tests, these were not good enough. In addition to the technical and visual reasons, the openness of the format is still being disputed. That does not mean that WebM does not have a broadcast future! Because the development of WebM is an ongoing (open) process, all the code and content of this test are available for retesting purposes or under open source / creative commons license. The results are given back to the world with a question: Can you improve the test results?

Bram Tullemans

Project Manager,

Broadband Techniques & Online Services,

EBU Technical[1]

---

[1] At the time of broadcast evaluation of WebM resulting in the TNO report Bram Tullemans was working as a Senior Policy Advisor R&D at the NPO.

Management summary

This report presents the results of a WebM pilot, which was performed by the Dutch public broadcaster (NPO) in cooperation with CodeShop and TNO. These results were obtained while developing and using a live and VoD service based on the open video format WebM. The design choices within this pilot were based on the following principles:

- Focus on practical usage for broadcasters and their operational requirements,
- Use of open source software, open standards and open content, allowing third parties to reproduce the results of this pilot, and continuing development with the provided tools,
- Focus on reuse and configuration of existing open source components instead of developing new components.

Google is the driving force behind WebM. It is a free and open format for web-based video services allowing video to be played in web browsers without a plug-in. It allows anyone to implement WebM on any platform without licensing fees. Currently, most web video services use the H.264 video format in combination with proprietary plug-ins. This raises two issues: first, the functionality of these proprietary plug-ins is controlled by a single provider, such that broadcasters have limited influence on future developments. Second, these plug-ins are not available on every platform, such that the reachability of these video services is limited to the supported platforms. For example they are not fully available for Linux, Apple or mobile devices.

WebM could provide a solution to this problem, because Google provides WebM components that can be used freely by anyone. Therefore, WebM is a candidate as an open video standard. This is relevant for NPO, also because of the political debate with respect to open standards and financial cutbacks.

Since the introduction of WebM, numerous comparisons have been made between WebM and H.264 typically focusing on the video quality. For broadcasters, this is one of many criteria that determine whether WebM provides an alternative for H.264. This pilot investigates the practical usage of WebM. It assesses all relevant criteria, from video quality to its patent position, from encoding speed to deployment in the current operational setting.

With respect to usability of WebM the most important conclusions are:

- At this moment, it is impossible without extra development to deploy live and VoD services based on WebM using open source components that satisfy the operational requirements of NPO.
- The fact that Google is a driving force combined with the increasing support of industry partners gives confidence with respect to expected potential of WebM for broadcasters.
- The uncertain situation of patents regarding WebM could obstruct the roll out of WebM, because it introduces uncertainty with respect to future costs, but it also prohibits the possibility of WebM being an open standard.

Used abbreviations

DRM – Digital Rights Management
MXF – Material Exchange Format
NPO – Nederlandse Publieke Omroep
SD – Standard Definition
SDI – Serial Digital Interface
TRL – Technology Readiness Level
UZG – Uitzending Gemist
VoD – Video on Demand

Contents

1 Introduction

This report presents the results of a WebM pilot. These results were obtained while developing and testing a live and VoD service that was based on the open video format WebM. The design choices within this pilot were based on the following principles:
- Focus on practical usage for broadcasters and their operational requirements,
- Use of open source software, open standards and open content, allowing third parties to reproduce the results of this pilot, and continuing development with the provided tools[2],
- Focus on reuse and configuration of existing open source components instead of developing new components.

The goal of this pilot was to determine whether the open media format WebM provides an alternative for H.264, which is the most important video standard at this moment. WebM is developed by Google, and it is positioned as the free alternative for H.264. WebM has the potential to become a standard for “free” video on the internet. For this reason, WebM is important within the context of VoD services like “Uitzending gemist” (UZG), the Catchup TV service of the Dutch public broadcasters (NPO). This pilot investigates the maturity of WebM.

This pilot is performed by NPO, CodeShop and TNO. These parties cooperated by fulfilling these roles: access provider and controller of UZG (NPO), setup and configuration of video chains based on open source software (CodeShop) and qualitative assessment of the video quality and the delivery chain (TNO). Finally, TNO was responsible for the project management.

1.1 Background

In 2009 committee Brinkman[3] concluded that the Dutch public broadcasters should use more open source software and open standards, and the obtained knowledge by using these techniques should be shared, for example with publishers. Besides these conclusions, open source software and open standards are recognized as important by public broadcasters, because the techniques are often free, can be reused by others and can be altered as needed. On the other hand, its practical usage can become more complicated for different reasons, such that in the end the “open” solutions can be more expensive than the “closed” couterparts. For this reason, the investigations to use open source software and open standards focus on gaining insight in its practical usability.

Currently, most web-based video services use the closed H.264 video format in combination with proprietary plug-ins like Adobe Flash and Microsoft Silverlight. This raises two issues: first, the functionality of these proprietary plug-ins is controlled by a single provider, such that broadcasters have limited influence on future developments. Second, these plug-ins are not available on every platform,

[2] All pilot tools, content en configurations can be found: http://kenniscentrum.publiekeomroep.nl/pilot/WebM
[3] https://zoek.officielebekendmakingen.nl/blg-23864.pdf

such that the reach of these video services is limited to the supported platforms. In case of UZG, they are not fully available for Linux, Apple or mobile devices.

WebM is an open media format for web-based video services which was released by Google in 2010. WebM consists of three components:
– A container format for storage and distribution, which is based on the open Matroska format;
– the VP8 video format[4], which is developed by Google;
– The open Vorbis[5] audio format.

This project investigated the usage of open source software that encodes video to the WebM format, because:
– WebM has an active community (which is a good identifier for the available support and future software development),
– WebM could fit to the current operational requirements of a broadcaster (i.e. can the software be easily be deployed in the current operational environment,
– WebM could lead to cost savings.

1.2 Objectives

The main objective of this pilot was to give insight in the blocking factors of practical usage of WebM. This objective is achieved by answering the question whether WebM is an alternative to H.264 for web-based video services offered by a broadcaster. For the broadcaster, WebM is an alternative if two WebM-based video delivery chains can be developed which provide a live stream and a VoD service, with a quality of experience that is comparable to the current operational services. Figure 1 presents these delivery chains with their needed components:

– **SDI capture** (in Standard Definition (SD)format) using a SDI capture card for live input,
– **Video encoder** to encode WebM from the SDI input, as well as from files in MXF file format, in case of VoD,
– **A web server** to play out the live and VoD streams,
– **A video player** to watch WebM encoded content in a HTML5 enabled web browser.

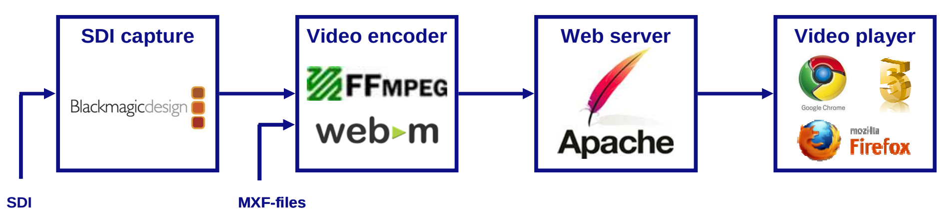


Figure 1: The investigated pilot setup, including the open source components that were used.

[4] VP8 is developed by On2 Technologies, and taken over by Google in 2010. Precursor VP6 was the video format of Flash before the introduction of H.264.
[5] http://www.vorbis.com/

To answer the question whether WebM provides an alternative, the following sub-questions are answered by this report:

- What are the blocking factors that hinder the setup of these two video delivery systems? To answer this question, these delivery systems were developed and chapter 2 presents the obtained observations.
- Can a WebM-based video service be deployed by a broadcaster in the near future by only using open source components? To answer this question the delivery chains are evaluated in its complete context, so from its patent position to play out options in different browsers. Chapter 3 presents these results.
- How does WebM perform regarding video quality and encoding speed with respect to H.264? To answer this question a subjective video quality assessment was performed and an encoder benchmark was executed. The observations and results are presented in chapter 4.

Finally, chapter 5 presents the most important conclusions with respect to the practical usage of WebM in comparison to H.264.

2 Observations WebM pilot

This chapter presents the most important observations that were obtained during the development of the WebM-based live- and VoD-services. These observations have a strong relation with the possibilities of WebM, but also with the barriers of open source software.

An important argument for many choices in the pilot setup was that this setup should fit to the current operational requirements for video services of a broadcaster. For instance, the open source solution Flumotion, which is developed by Fluendo, was rejected for this reason. This solution provides a live and VoD-service, but as a totally separated delivery chain, which is not preferable from an operational and manageability perspective.

2.1 SDI video capture

For the live-delivery system, a live-signal in SD format should be captured using a capture card, and thereafter the captured video signal should be encoded in the WebM format. This process requires two components:

1. SDI card to capture the SD signal
2. Software to provide this signal to the encoder

2.1.1 SDI capture-card

At first, the Centaurus LT II capture card[6] was chosen. The software provided with this card did not provide the possibility to stream the captured video to an encoder. This software turned out to be focused on writing a captured stream to a file. For this reason, this card could not be used for live-streaming without extra development effort, because there was no open source software available that provided the required functionality for this card. For this reason, another card was selected: BlackMagic Decklink[7]. This card turned out not to be as stable as the Centaurus, but it was also available at lower cost. This is probably the reason that it had been adopted by the open source community, which has resulted in several tools, one with the purpose of capturing and encoding an SDI signal using FFmpeg software.

2.1.2 SDI to encoder

Within the open source community the toolkit Ingex was discovered, which was developed by BBC[8]. It is a toolkit that captures an input signal and stores the audiovisual content from the production process, where it uses a SDI capture card and a standard PC. During the pilot it was concluded that Ingex does not meet the requirements, as it could not output live-feeds; it was developed for writing to files. For this reason Ingex was not adopted.

[6] http://www.dvs.de/products/video-boards/centaurus-ii-lt.html
[7] http://www.blackmagic-design.com/products/decklink/
[8] http://ingex.sourceforge.net/index.html

2.2 Video encoding

For the live- and VoD-system two encoding processes are needed:

1. Live-encoding of the SDI signal to a WebM live stream,
2. Transcoding of MXF-files to WebM files.

For these two tasks, the open source software FFmpeg is chosen, which was already in use by the broadcaster for different multimedia processing tasks. This toolkit contains a program that converts multimedia files, a library with audio and video codecs (libavcodec) and a library with container and distribution formats for audio and video data (libavformat).

2.2.1 Live-encoding

For the live-service, the SDI signal should be encoded to a WebM stream. To serve the live-stream, WebM requires a different encoding format (bitstream format) than for WebM files (for VoD)[9]. Contrary to file-based encoding, this specification is not yet implemented in FFmpeg. It is implemented in Gstreamer[10], which could serve as a potential alternative for FFmpeg.

This issue gives insight to a relevant aspect of open source software. Because, (almost) all tools are freely available, it is rather simple to investigate whether other tools have relevant functionality and to test whether these tools satisfy the requirements. It shows that FFmpeg provides most required functionalities, but a crucial element is not (yet) implemented. Possibly, FFmpeg will be further extended, such that the functionalities for live streaming are available in the future as well. Finally, Gstreamer is not used, because FFmpeg is already used operationally by NPO.

2.2.2 MXF-encoding

The Dutch broadcasters deliver their programs for distribution in the MXF file format[11]. For VoD services these MXF files should be encoded to WebM. MXF provides functionality to support multi-channel audio, a feature that is used by the broadcasters. FFmpeg supports MXF file input and can encode the video in the MXF files to WebM files using the VP8 library correctly. The audio however could not be encoded correctly to WebM's vorbis audio format. This is related to FFmpeg's lack of support for down-mixing multi-channel audio to stereo, being the default audio configuration for broadcasters web services. This implies that it is difficult to use FFmpeg for MXF processing in an operational setting. The same audio-mapping issues where encountered with Gstreamer.

In 2008 a patch was developed within the FFmpeg community that provides the down-mixing from multi-channel audio to stereo. This patch was offered to the FFmpeg maintainers to be added to the main FFmpeg release. The maintainers however did not accept this patch. The reason to reject this patch, was related to the functionality offered by the patch, which would not work for all down-mixing cases. The developed patch can be used by anyone who is interested, but a user then must add this patch to his FFmpeg code. However, this is not attractive because compatibility of the patch with newer FFmpeg releases is not guaranteed.

---

[9] WebM pilot repport by CodeShop
[10] http://gstreamer.freedesktop.org/
[11] For an introductory article of the MXF file format, see: http://tech.ebu.ch/docs/techreview/trev_2010-Q3_MXF-2.pdf

This case is illustrative for the open source community. For many problems, (partial) solutions exist, but these are not integrated in official releases. An advantage of open source software is that anyone can adapt the source code to fit ones need, or integrate a required patch if needed. However, this implies that a developer continues to build on the functionality offered by the patch. This means that an alternative version (a branch) of the software is created, which loses its connection with the main releases.

Summarizing, for some cases, solutions (in the form of patches) are available, but it might be difficult to use them, as it requires expert knowledge to use them. If a solution does not exist, it is possible to develop a patch by yourself or a third party, but it might be hard to get such a new patch accepted by the community. In order to have a patch accepted in daily releases it should be developed as generic as possible.

2.2.3 VP8 encoder

To encode video in the WebM format, the VP8 encoder library libvpx is used, which was released by Google. The encoder benchmark (see chapter 4) shows that the libvpx encoder has a relatively low encoding speed, meaning that real-time encoding is only possible for low resolution content and bit rates lower than 200 kbps, for which the video-quality is limited. Different solutions were investigated to increase the encoding speed:

- Encoding VP8 using Gstreamer instead of FFmpeg. It was concluded that Gstreamer provided no advantage over FFmpeg: encodings tests showed comparable results[12].
- Using a newer version of the libvpx implementation that was released during the pilot. This newer version had some performance optimizations implemented. However, it was concluded that these were not sufficient to allow live streaming[13].

This case showed a few typical aspects of open source software:

- It is easy to investigate alternative open source solutions, also during the development phase. This has the advantage that it is still possible to switch to an alternative solution, for example to Gstreamer in this case.
- For some popular open source projects, updates are provided by the community on a regular basis. For closed source software, this can also be the case, but then the updates are typically provided by a single point-of-contact, who decides whether certain functionality will be developed or not. On the one hand a single point of content is preferable from a management point of view, on the other hand if the provider does not give priority to a certain wish or requirement, then it is a risk that it will not be developed at all. For open source software, it is possible to develop this solution by yourself or by third parties, such that there is at least the possibility to realize a solution.
- This pilot shows that the combination of libvpx and FFmpeg is not very mature with respect to the broadcaster requirements. For broadcasters, software can be used if its TRL (technology readiness level) is sufficiently high to integrate it as an operational service. This applies to closed source but to open source

[12] GStreamer release 0.10.30
[13] Old version: Webm release 0.9.2. Newer snapshot version: v0.9.5-286-g945dad2. The latter was used in the final pilot setup.

software as well, but in case of open source this might need to be determined by the broadcaster itself.

2.3 Web server

At the start of this pilot, the open source web server Lighttpd[14] was the most important candidate to be used as a web server. When the project was started it became clear that it was better to use the Apache[15] web server, being more scalable then Lighttpd for serving video concurrently. Additionally, Apache supported chunked-encoding, an important feature for http based live-delivery. Another reason to switch to Apache was the fact that activity in the Lighttpd community, especially regarding development of new functionality, had decreased significantly. For this reason, the ICT department of the broadcaster switched its focus to Apache, which has an active community.

This example shows another open source aspect. The developments within the Lighttpd community had stopped. Choosing an open source product for which activity in the community has halted comes with a risk: developers with the required knowledge to solve problems, extend functionality and maintain this solution can become scarce or might not be available at all.

2.4 Video player of the webbrowser

WebM encoded video can be watched in a web browser that supports HTML5. Table 1 presents an example of how a video-object is addressed in HTML 5. At this moment only Chrome, Opera and Firefox4 support WebM out-of-the-box, without requiring a plug-in. An evaluation of some quality of experience aspects regarding playback in these browsers are presented in chapter 3.

Table 1. Example of the video-object in HTML5 code

```
<video
  src="http://webm-pilot2.omroep.nl/video/
  big_buck_bunny_480p_stereo.ogg.webm"
  width="320" height="240" controls="yes" autoplay="yes">
  Your browser does not support the video tag
</video>
```

2.5 Conclusions

The development of the live- and VoD-systems lead to the conclusion that at this moment it is impossible to setup a live-service based on WebM using the investigated open source components, which fulfills the requirements of the broadcaster. To realize this delivery system, FFmpeg should be extended with

[14] www.lighttpd.net/
[15] http://www.apache.org/

support for WebM for live purposes and the encoding performance with respect to the encoding speed must significantly increase to provide a service with reasonable video quality.

Additionally, a robust VoD-service cannot be realized, because of the absence of correct mapping of multi-channel audio to stereo, such that proper audio support is not available for this service. To solve this issue different solutions exist, nevertheless it requires an extra investment. The encoded MXF files contain a proper video stream, so it is possible to evaluate the picture quality of the video stream (see Chapter 4).

Given the drive of Google to support WebM, it is probable that in the upcoming period the missing elements will be developed and become available. Another possibility is to develop these missing elements in-house or by a third party, if these features are not developed fast enough.

With respect to the development of the delivery chains, the following conclusions are relevant with respect to usage of open source software:

- Using open source software provides the possibility to investigate alternatives, like Gstreamer instead of FFmpeg, or Apache instead of Lighttpd.
- As soon as the TRL is sufficiently high, open source software can be used by a broadcaster to provide its services. This is the case for the open source web server Apache. This TRL aspect is also important in case of closed source software, but the difference is that open source software is sometimes published in an early stage.
- Functionalities that are not available in open source software can be developed in-house or by a third party. In order to get these solutions accepted by the maintainers of the open source project, such that they become available in the main releases of the open source software, these functionalities should be developed as generic as possible.
- With respect to maintenance and control, it is essential to have knowledge of the open source software available, possibly in the community, with an implementation partner or possibly within the broadcasters' organization. If this knowledge is not available, an alternative solution should be selected.

3 Evaluation of the VoD-platform

This chapter presents the evaluation of the WebM-based VoD-platform. For a broadcaster, many aspects can influence the success of a WebM-based video service. In advance of this pilot, a business case[16] was developed to get an overview of all these aspects. This chapter evaluates the developed delivery chain based on these aspects. The evaluation uses two episodes of the TV program “Tegenlicht”, which are published under an creative commons license. This content was chosen to keep this pilot as “open” as possible.

3.1 Costs

The idea that creating a service based on open source software[17] without any cost is not true. The advantage of open source is the fact that it results in a solution, without license fees, such that operational costs can be low. Typically, obtaining and using the open source software is free, but of course providing a service is not for free. It requires operational, maintenance and possibly development costs. Potentially, WebM could result in savings related to licensing costs of H.264, if the development and deployment costs of a WebM-based video service are less than these licensing costs.

The software used in this pilot is released under GPL license. This means that all components can be used commercially by third parties without license costs. The software covers all existing open source components, tooling developed by CodeShop and the programs developed by TNO for the video quality assessment. The documentation and used test material is released under open conditions as well: the VRPO Tegenlicht content is available under a creative commons license, and the EBU test footage[18] can freely be used for for testing purposes. So, as a result this pilot can be repeated by third rather easily.

Given the identified issues in chapter 2, it is expected that the costs to solve these issues will be higher than its potential licensing fee savings.

3.2 “Open” aspects

3.2.1 *Public opinion*

The Dutch government demands public parties, like public broadcasters, to use open source software and open standards as much as possible. Within this context the program “Nederland Open in Verbinding” (*Netherlands Open in Connection*. NOiV) was started to stimulate usage of open source and open standards. According to NOiV an open standard is characterized by the following properties[19]:

---

[16] “Open Source Software Pilot gebaseerd op WebM”, by TNO 2010.
[17] Assuming the open source software is provided on a royalty free basis
[18] http://tech.ebu.ch/Jahia/site/tech/cache/offonce/hdtv/hdtv_test-sequences
[19] NOiV Definition of an Open Standaard (Dutch): https://noiv.nl/open-standaarden-en-open-source/wat-zijn-open-standaarden/

1. The standard is approved by, and is maintained by a non-profit organization. Current developments are taking place according to an open decision making process that is accessible for all interested parties.
2. The standard is published and the specification can be obtained for a nominal or no fee.
3. The intellectual property is irrevocable available on a royalty free basis.
4. There are no limitations with respect to re-usage of the standard.

At this moment WebM does not satisfy properties 1 and 3[20]. This implies that WebM is not an open standard, but because serious alternatives are not available WebM is considered to be the most open video standard.

This pilot investigated this “open” development with the restriction that it should satisfy the operational requirements of the current VoD-services. This made the pilot realistic in comparison to investigations that only focus on open standards or in case of WebM investigations that primarily focus on the video quality.

Figure 2. Parties that joined the WebM Community Cross-License Initiative[21]

3.2.2 *Patent analysis for WebM*

Patents are important with respect to risks of using WebM. One of the entailments to use closed standards is the license fee for its usage, usually due to usage of patented technology. Although WebM is positioned as a free format, it is possible that other parties, besides Google, have patents on WebM, or more precisely, have patents covering parts of the VP8 video format. This could result in a situation that WebM usage will not be free of charge in the future. Recently, two developments are important within this context:

[20] Google irrevocably provides VP8 on a royalty free basis, but as will be clear from section 3.2.2, other parties likely hold patents on VP8 technology as well.

[21] Source: http://www.engadget.com/2011/04/26/coalition-of-companies-creates-webm-community-cross-license-init/

- The MPEG-LA[22] made a call to form a patent licensing pool[23][24] for WebM and they announced that some parties want to join this pool. At this moment MPEG-LA is investigating whether these parties indeed hold relevant patents. If this is the case, then this process could result in a situation where WebM is no longer free of charge.
- Google started a WebM Community Cross-license Initiative[25], which is a kind of opposite patent pool, because the goal of the initiative is to achieve a royalty-free situation for all joining parties. These parties include Google, Cisco, Logitech and Samsung as figure 2 shows.

These developments suggest that besides Google other parties are having patents on WebM. As long as it is not clear whether WebM is or remains royalty-free, WebM cannot become an open standard, such that it can be accepted by W3C[26] or become the standard multimedia format for HTML5.

3.3 Quality of Experience

The quality of experience (QoE) is a subjective measure of the quality as perceived by an end-user. For the WebM pilot several QoE metrics where selected and evaluated, where possible:
- **Picture quality** as a function of its bit rate; chapter 4 elaborates this aspect.
- **Playback delay** is the time the service takes between clicking on an item for playback and the moment it starts to playback. While testing the playback of the VoD functionality some latency problems were identified, causing a WebM video to be buffered entirely before playback started. This problem was solved for the Opera web browser by adjusting the HTML5 code, but it was not yet solved for playback in Chrome and Firefox 4. Playback of the content in Firefox 4 was not very fluent, which decreases the quality of experience. This aspect showed that quite some work is to be done, to obtain a quality that satisfies the end-users. At this moment, it is not clear, whether these issues are due to the encoding process, the Apache play out, or to the respective WebM implementations in the HTML5 browsers.
- **Trick play:** play, stop and pause functions work properly, but forward and backward jumping within an episode required quite some buffering and thus reduces its responsiveness.
- **Skip between items;** this aspect is not evaluated.

3.4 Manageability and sustainability

The fact that Google invests in WebM and other major companies are supporting Google gives a certain guarantee for the future. It is good that a party as Google invests in an open video format, but a question that remains is what is Google's

[22] MPEG-LA organizes patent pools for multimedia formats: http://www.mpegla.com/main/default.aspx
[23] http://www.mpegla.com/main/pid/vp8/default.aspx
[24] http://www.h-online.com/open/news/item/WebM-Community-Cross-License-initiative-launched-1232367.html
[25] http://www.webm-ccl.org/
[26] World Wide Web Consortium (W3C) develops standards for the web. They are responsible for standards as HTML, XHTML, XML en CSS.

motive? For Google, the licensing costs for using H.264 in products like YouTube, GoogleTV and Android can be huge, such that WebM could possibly provide a royalty-free alternative. These developments will not be experienced as positive everywhere.

Table 2 presents two messages that give some expectations for the future of WebM. It makes clear that the problems that where identified in this pilot, for example with respect to encoding speed, are being addressed by Google as well.

Table 2. Two messages that give support positive expectations for WebM

VP8 Codec SDK "Bali" Released[27]
For Bali we focused on making the encoder faster while continuing to improve its video quality. Using our previous releases (our initial 0.9.0 launch release and "Aylesbury") as benchmarks, we've seen the following high-level encoder improvements:

- "Best" mode average encoding speed: On x86 processors, Bali runs 4.5x as fast than our initial release and 1.35x faster than Aylesbury.
- "Good" mode average encoding speed: Bali is 2.7x faster than our initial release and 1.4x faster than Aylesbury.
- On ARM platforms with Neon extensions, real-time encoding of video telephony content is 7% faster than Aylesbury on single core ARM Cortex A9, 15% on dual-core and 26% on quad core.
- On the NVidia Tegra2 platform, real time encoding is 21-36% faster than Aylesbury, depending on encoding parameters.
- "Best" mode average quality improved 6.3% over Aylesbury using the PSNR metric.
- "Best" mode average quality improved 6.1% over Aylesbury using the SSIM metric.

Introducing "Anthill," the First VP8 Hardware Encoder IP Release[28]
Last week the WebM Finland team finalized our H1 hardware RTL design. The H1 is the world's first VP8 hardware encoder. This initial release, which we're calling "Anthill," is now available through the WebM Project hardware page. Google does not require payment of any license fee or royalty in connection with use of the H1 encoder RTL.

With respect to the practical manageability of WebM, in the end it is merely alternative for H.264, so it is reasonable to expect the WebM-based video platforms to be controlled in a comparable way as the current operational web-based video platforms.

3.5 Reachability and future perspective

For broadcasters, it is important to reach as many users as possible. With respect to a video service based on WebM the dominating factor is the number of web browsers that potentially support WebM. Google Chrome, Firefox4 and Opera provide out-of-the-box WebM support. Internet Explorer provides support for WebM as of version 9 using a plug-in[29]. To stimulate the usage of WebM, Google announced to drop native video support of H.264 in Chrome, such that H.264 based

[27] http://blog.webmproject.org/2011/03/vp8-codec-sdk-bali-released.html
[28] http://blog.webmproject.org/2011/03/introducing-anthill-first-vp8-hardware.html
[29] https://tools.google.com/dlpage/webmmf

video can only be used with another plug-in such as Flash or Silverlight. Table 3 presents an overview of browsers that support WebM. This analysis shows that three major players are fighting for the viewers in the current web landscape: Google (advertisement and content), Microsoft (software) and Apple (devices and content).

Tabel 3. Browsers supporting WebM, where Flash and Silverlight are browser plug-ins

Platform	**Container**	**Format**		**Transport**
Web browser	WebM	Video	Audio	
		VP8	Ogg Vorbis	
Microsoft Internet Explorer 9 (Only Windows Vista and Windows 7)	Google cooperates with Microsoft to support WebM. A preview is available.			HTTP
Apple Safari 4	No support			-
Google Chrome 3	✓			HTTP
Mozilla Firefox 3.5	✓			HTTP
Opera 10	✓			HTTP
Android Browser	Available in Android 2.3			HTTP

With respect to reachability Google and Apple are winning market share from Microsoft as table 4 shows. The previous reachability analysis was made in December 2010 and at that moment the percentage of potential WebM viewers was about 30%. Currently approximately 40% of the Dutch browser market supports WebM. When the IE9 viewers are taken into account, then this percentage increases to nearly 50%. For mobile, Google added WebM support to Android 2.3. Given the growth of the Android platform in smart phones market share (36% in US), the potential reach of mobile users is also high[30].

Table 4. Market share of different web browsers in the Netherlands, (green = WebM support, red = no support, shaded = support by plug-in)

Web browser	NL December 2010 (Statcounter[31])	NL June 2011 (Statcounter[32])	Trends
Microsoft IEx	**63,7%**	**56,22%**	↓
Mozilla Firefox	**19,8%**	**19.36%**	≈
Google Chrome	**9,8%**	**16.05%**	↑
Apple Safari	**4,9%**	**6.91%**	↑
Opera	**1,2%**	**1.04%**	≈
Other	**0,6%**	**0.42%**	≈

[30] http://blog.nielsen.com/nielsenwire/consumer/android-leads-u-s-in-smartphone-market-share-and-data-usage

[31] Statcounter; http://gs.statcounter.com/#browser-NL-monthly-200909-201009

[32] Statcounter; http://gs.statcounter.com/#browser-NL-daily-20110601-20110606

3.6 Video platform functionality

Besides playback of live and VoD, the video services offered by a broadcaster have several additional requirements.

3.6.1 *Digital rights management*

Digital rights management (DRM) for the content protection of a broadcaster is mandatory to be able to receive broadcasting rights for sports events such as soccer matches or the Olympic games.

HTML5 does not have the possibilities to manage DRM as also SURFnet concluded in their HTML5 inventory[33]. Flash and Silverlight are able to provide these functions. For this reason it is hard to protect content in a WebM-based setup and at this moment there are no solutions provided by third parties.

3.6.2 *Audience measurement*

The Dutch public broadcaster has the legal obligation to report viewer statistics to the Commesariaat voor de Media (Commisseriat for the Media).

At this moment audience measurement is performed by specific plug-ins based on Flash and Silverlight. In case of HTML5 these results can be achieved using javascript. How this should be solved exactly is not investigated by this pilot.

3.6.3 *Adaptive bit rate streaming*

Adaptive bit rate streaming enables the easy offering of video services on internet connections with variable bandwidth conditions.

HTML5 does not specify an adaptive bit rate streaming mechanism. The community behind WebM is trying to develop an adaptive bit rate streaming mechanism for WebM, but it is unclear when it will be available, or if it will be adopted by all HTML5/WebM enabled web browsers.

3.6.4 Concluding remarks

At this moment these functions are not (yet) supported with WebM (in combination with HTML5 based playback), and this pilot did not investigate possibilities to implement these functions since it showed that the service in itself is hard to realize.

These functionalities will become more and more important as soon as WebM is providing a serious alternative for H.264. Most of these functions are not specific functions of WebM, but mainly browser of application functions that should support also WebM. As stated these functions should still be developed. So, these observations support the conclusion that WebM is not mature enough for web-based video services that satisfy the operational requirements of a broadcaster.

3.7 Conclusions

This chapter provided the evaluation of the VoD-delivery platform based on relevant aspects related to a business case for WebM. The most important conclusions are:

---

[33] http://www.surfnetkennisnetproject.nl/attachments/session=cloud_mmbase+2312355/HTML5_rapport_feb2011.pdf

- It is probable that third parties have patents on WebM. Google and MPEG-LA try to make this transparent. If this aspect will not be solved, then it could be a show stopper to rollout for WebM, at least as open standard.
- The fact that Google invests in the development of WebM gives confidence in the future of WebM, such that it can become a serious alternative for H.264.
- The potential reach of WebM has grown from 30% in December 2010 to nearly 50% in June 2011. If this growth continues, then reach is not a problem any longer. To be realistic, it is a utopia to achieve 100% reach ability given the large number of different devices and platforms, but a reach similar to or bigger then the reach obtained with H.264 based video services is not unlikely
- WebM does not provide a serious alternative for H.264, because important functionalities as DRM, audience measurement and adaptive streaming are not available. Given the design of HTML5, it is a question whether DRM and audience measurement can be implemented with a comparable security and reliability level as proprietary solutions currently offer. Second, because the quality of experience depends on the web browser and this pilot showed that it requires more investigations to provide a good experience for all browsers.

4 Qualitative WebM evaluation with a panel test

Determining video quality is not a trivial procedure. On one hand, it is preferable to obtain an absolute measure for an encoder, such that different encoders can be compared easily. On the other hand, there is a large number of factors that determine the quality of an encoder, such that it is nearly impossible to define this objective quality measure. To obtain insight in the quality of WebM is chosen to develop a subjective panel test. It compares the quality of WebM video (encoded with VP8 encoding library libvpx) with the quality of H.264 video (with H.264 encoding library libx264) using FFmpeg as encoding tool. X264 is used as a reference, because this open source implementation of the H.264 encoder is broadly used. It is considered one of the best H.264 encoding implementations and since the release in 2004 has been optimized by the x264 community.

a. Crowd run

b. Ducks take off

c. Park joy

d. Talking woman

Figure 3. The four fragments used for the subjective panel test

4.1 Approach

The experiment is focused on the video quality that a UZG user currently experiences. For this reason, fragments are compared with a resolution of 640 x 360 pixels, the resolution used for UZG video.

4.1.1 Test deployment

Four fragments were selected for the evaluation, where each fragment consists of one shot with a typical feature. These fragments are encoded using different profiles for the respective encoders with different bit rate settings. The obtained fragments are compared with each other by a test panel that consists of six participants. Each participant was forced to select the fragment with “best” quality. This test is called a binary paired-comparison test, because fragments are compared pair-wise, and it is binary because the participants were forced to choose.

Figure 3 presents a representative frame for each fragment. These fragments are assumed to be representative for the Dutch television and UZG. Three fragments are from the EBU[34]: “Crowd run”, “Ducks take off” and “Park joy”. These are used all over the world for quality assessments. The fourth fragment originates from the “Tegenlicht” episode “California Dreaming”[35] and obtained the name “Talking woman”. Table 5 presents the motivation and settings of these fragments.

Table 5. Technical details and motivation for the selected fragments

ID	Motivation (duration) / Sequence ID	Type	Source resolution
EBU_CROWD 1_CrowdRun_720p50_CgrLevels_SINC_FILTER_SVTdec05_	Much detail, colors and continue much movements (10s)	Image sequence	1280x720 50 fps
EBU_PARK 2_ParkJoy_720p50_CgrLevels_SINC_FILTER_SVTdec05_	Typical color usage and focus foreground / background (10s)	Image sequence	1280x720 50 fps
EBU_DUCKS 3_DucksTakeOff_720p50_CgrLevels_SINC_FILTER_SVTdec05	Quite shot, with sudden movements (10s)	Image sequence	1280x720 50 fps
VPRO_WOMAN TEGENLICHT_CA-VPRO00002VUJ_115000_3178600.mxf	Static camera and subtitling. (11s)	Video file MPEG-2 in MXF container	1920x1080 25 fps

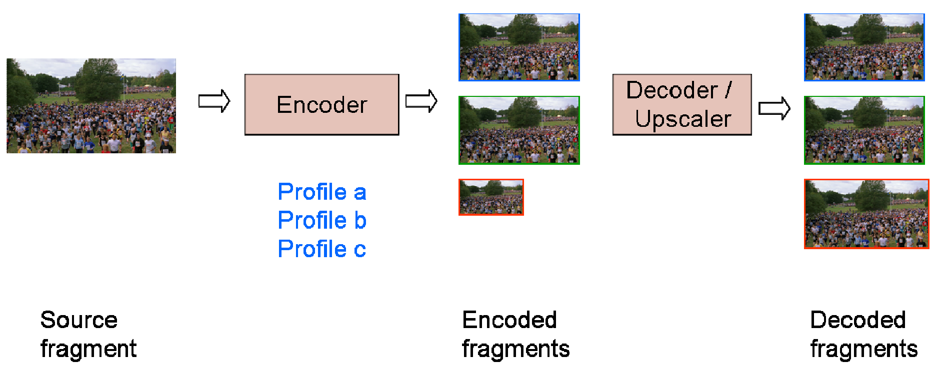


Figure 4. The source is encoded according to a selected profile; the results are converted to an uncompressed format and scaled to the right resolution.

4.1.2 *Encoding profiles*

Before the four fragments were encoded with the chosen profiles, each source was transcoded to a uniform format without quality loss. The pilot focused on SDI signals and therefore the sources were transcoded to SD: MPEG-2 Video, a video

[34] EBU: http://tech.ebu.ch/Jahia/site/tech/cache/offonce/hdtv/hdtv_test-sequences
[35] Tegenlicht: http://tegenlicht.vpro.nl/afleveringen/2010-2011/california-dreaming.html

bit rate of 50Mbps, a resolution of 720 x 406 and a frame rate of 25fps in a MXF container.

Table 6 presents the 16 investigated encoding profiles. The default encoding settings of FFmpeg were used: libx264 (fast, normal) and lipvpx (360p). All source files were encoded with each profile. Thereafter, it was decoded to a lossless format, such that they could be merged in one video for comparison. Figure 4 presents this process for three profiles graphically.

Table 6. Overview of the 16 investigated FFmpeg profiles. “downscaling” implies that the content is downscaled to a lower resolution than UZG, while “default” means direct encoding to resolution 640 x 360.

ID	Codec	Resolution	Muxed Bit rate (kbps)	Video Bit rate (kbps)	Encoder preset	Profile
1	libvpx	112x64	64	48	360p	libx264_downscaling_64k
2	libvpx	320x180	200	140	360p	libvpx_downscaling_200k
3	libvpx	640x360	200	140	360p	libvpx_default_200k
4	libvpx	640x360	350	302	360p	libvpx_default_350k
5	libvpx	640x360	500	460	360p	libvpx_default_500k
6	libvpx	640x360	750	694	360p	libvpx_default_750k
7	libvpx	640x360	1000	908	360p	libvpx_default_1000k
8	libx264	112x64	64	48	fast	libx264_downscaling_64k
9	libx264	640x360	200	140	fast	libx264_default_200k
10	libx264	320x180	200	140	fast	libx264_downscaling_200k
11	libx264	640x360	350	302	normal	libx264_defaultnormal_350k
12	libx264	640x360	350	302	fast	libx264_default_350k
13	libx264	640x360	500	460	fast	libx264_default_500k
14	libx264	640x360	750	694	fast	libx264_default_750k
15	libx264	640x360	750	694	normal	libx264_defaultnormal_750k
16	libx264	640x360	1000	908	fast	libx264_default_1000k

The profiles are chosen based on the following arguments:

- **Codecs** are libvpx for WebM and libx264 for H.264. These libraries are compiled in FFmpeg. Table 7 presents the version numbers of the libraries that were used.
- **Resolution** is adjusted to 640 x 360. This is the maximum resolution that is provided by UZG. Besides this resolution, 320 x 180 and 112 x 64 were used for lower bit rates.
- **Bit rates** are bases on the bit rate settings used by UZG and the broadcasters tests with adaptive bit rate streaming. Only the video bit rate is taken into account, the audio bit rates are ignored. Audio was not part of the evaluation.
- **Encoding speed** is important, because live content should be available on UZG fast. For this reason only single pass encoding is investigated and is dual pass ignored. For libx264 the presets “fast” and “normal” are evaluated, while for libvpx is preset p360 used.

Table 7. Used versions of libvpx, libx264 and FFmpeg.

Tool / lib	Version
libvpx (WebM)	v0.9.5-286-g945dad2
libx264	x264-snapshot-20110127-2245
FFmpeg	revision git-e22910b

To keep the test manageable a few criteria where not taken into account:

- Device profiles; many video standards provided specific profiles that determine which options of an encoder are used by taking into account. These profiles also determine the requirements of the decoder in the device used to display the content plus the features that the device supports. Given that a large number of these profiles exist, this test assumes the content is watched in a web browser on a laptop or PC.
- Frame rates; this test used a constant frame rate of 25 frames per second. In case of low bit rates often lower frame rates are used. This test ignored that aspect.

4.1.3 *Binary Paired-comparison test*

The panel test was performed by six users. The panel consisted of trained video quality assessors as well as untrained assessors. In a pairwise comparison of the same fragment encoded with a different profile setting they had to select the best encoded fragment. Figure 5 presents the interface that was used during this test. In total, each participant assessed ((16x15)/2=) 120 comparisons per fragment and in total 480 assessments. This implies that 480 fragments are identified as best encoded. Every time that a fragment is assessed as “best”, then the encoding profile obtained a point. The relative number of points that a profile obtained is used as a quality measure.

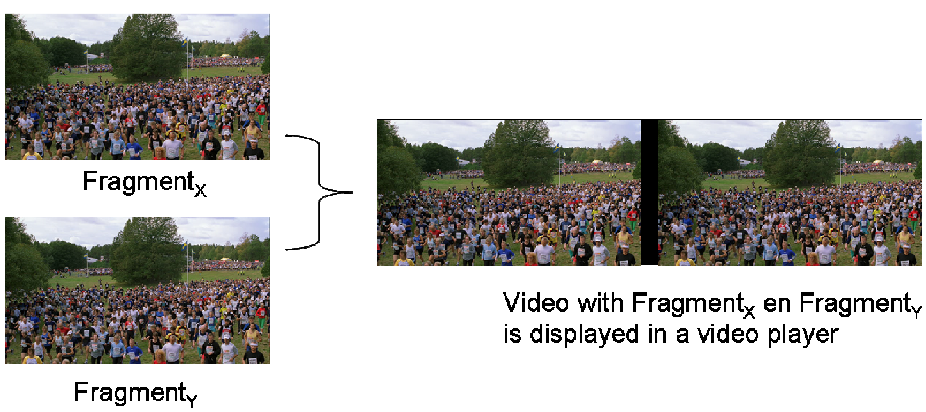


Figure 5: The generated video that was used for a pairwise comparison between two encoding profiles.

The test was performed on identical laptops with identical hardware and contrast adjustments. On each laptop VLC media player was used, where the participants had a distance of about 40 centimeter to the screen. This distance corresponds to five times the height of the fragment (~8cm). Appendix B presents the configuration of the test system. The total number of 120 comparisons per fragment are randomized before the test started (including random selection which fragment is displayed on the left and which one on the right). To avoid boredom, each fragment was shown in sets of 20 comparisons, thereafter the next fragment followed. Each fragment had duration of about 10 seconds. In total, 6 rounds were needed to show all the comparisons. It was up to the participants to take a pause when they wanted.

The large number of comparisons that should be assessed is the most important drawback of this test, because one participant was able to assess only 16 profiles in two hours. On the other hand, this test provides sufficient insight in the quality differences between different profiles.

Table 8. Ranking of all compared profiles and sorted on the relative number of comparisons that this profile was identified as best profile. Note: the encoding setting "libvpx_default_200k" led to an actual bitrate that was significantly (40%-120%) higher then the target bitrate.

FFmpeg profile	Percentage identified as "best" profile
libx264_default_1000k	89,7%
libx264_default_750k	83,3%
libvpx_default_1000k	82,7%
libx264_defaultnormal_750k	80,5%
libvpx_default_750k	77,2%
libx264_default_500k	67,1%
libvpx_default_500k	60,6%
libx264_defaultnormal_350k	53,6%
libx264_default_350k	52,2%
libvpx_default_350k	45,8%
libvpx_default_200k*	33,6%
libx264_downscaling_200k	25,8%
libvpx_downscaling_200k	20,6%
libx264_default_200k	17,5%
libx264_downscaling_64k	6,1%
libvpx_downscaling_64k	4,2%

4.2 Results

This section discusses the obtained results. First, the overall results are presented, which provide the most important conclusions. Thereafter, an analysis follows of the different fragments and the scores of the participants. Finally, the resulting bit rates of the encoded videos and the encoding speeds are assessed.

4.2.1 *Overall picture*

Table 5 presents the ranking of profiles based on the relative number of assessments that identified this profile as the “best” in the pairwise tests. First, this ranking shows that the scores linearly decreases with the selected bit rate. Second, it shows that libx264 scored better then libvpx in every case for the same bit rate. The differences between the libx264 and libvpx for a specific bitrate are limited, what lead to the conclusion that libx264 provides the best quality, but libvpx is slightly worse. With respect to picture quality WebM could provide an alternative for H.264, by assuming that both implementations provide the best possible performance.

4.2.2 *Outcomes per fragment and per participant*

Comparable rankings were found for each individual fragment. Only in the case of the VPRO_WOMAN fragment a deviation was found: the libx264 setting in normal mode at 350kps was significantly better than libx264 in fast mode at 350kbps. This observation can be explained as follows. This fragment is rather static. So, more encoding time, as used in normal mode, resulted in better encoding of picture details.

Because the panel test is a subjective test, participants can assess quality differently. One participant could have more problems with blocking effects, while another participant notices flickering. This effect could lead to structural differences in the obtained results. Table 9 presents an overview with the percentage of identical assessed fragments. All percentages are comparable to each other, and no outliers appear, so the results of all participants are relevant for this test.

Table 9. Confrontation table with percentages that different participants agreed with each other on the different assessments

	A	B	C	D	E	F
A	100%	88%	87%	86%	88%	86%
B	88%	100%	88%	89%	87%	87%
C	87%	88%	100%	87%	88%	86%
D	86%	89%	87%	100%	83%	83%
E	88%	87%	88%	83%	100%	87%
F	86%	87%	86%	83%	87%	100%

4.2.3 *Bit rates and encoding speed*

The encoders are configured to encode a fragment to a target bit rate as table 6 showed. In practice, the actual bit rate is not equal to the target bit rate. This difference was analyzed and in most cases the difference was about 2-3%. These relevant observations were obtained by investigating the bit rate:

- The libvpx encoder has difficulties to comply to the target bit rate. For all EBU fragments, the encoder with profile libvpx_default_200k uses about 40-120% more bits. This implies that the rating of this profile setting would be less good, if it would have been restricted to the target bit rate. This observation explains why this was the only profile, where libvpx scores better than libx264.
- The fragment VPRO-CA shows for all encoders a lower bit rate than its target bit rate. This can be explained by the fact it is a relatively static fragment and therefore has a lot of redundancy, which is not so hard to encode.

Appendix 3 presents the graphics that illustrate the differences between target and real bit rate.

Besides the picture quality and the resulting bit rates, the encoding speed is the third aspect that influences the video quality. In order to assess this aspect, the encoding speed of the encoders is determined by capturing the encoding time of the following two fragments[36]:
– 20 minutes of a “Tegenlicht” episode,
– 30 seconds fragment based on the three EBU clips.

The resulting encoding time is divided by the number of frames in the clips to determine the encoding speed in number of frames per second. This was then devided by the frame rate (25 fps) to determine the real-time encoding factor.

Libvpx and libx264 show rather great differences in encoding performance as shown in figure 6. Where libx264 can reach 4-5 times real-time encoding (100-125 fps) libvpx, cannot provide real-time encoding, only for very low bit rates and low resolutions. On average 12-16 fps could be encoded, which is ~0.5 times real-time.

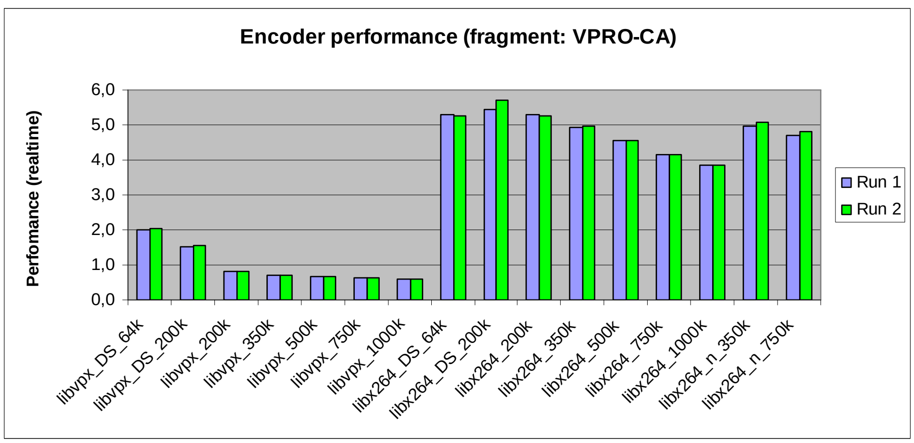
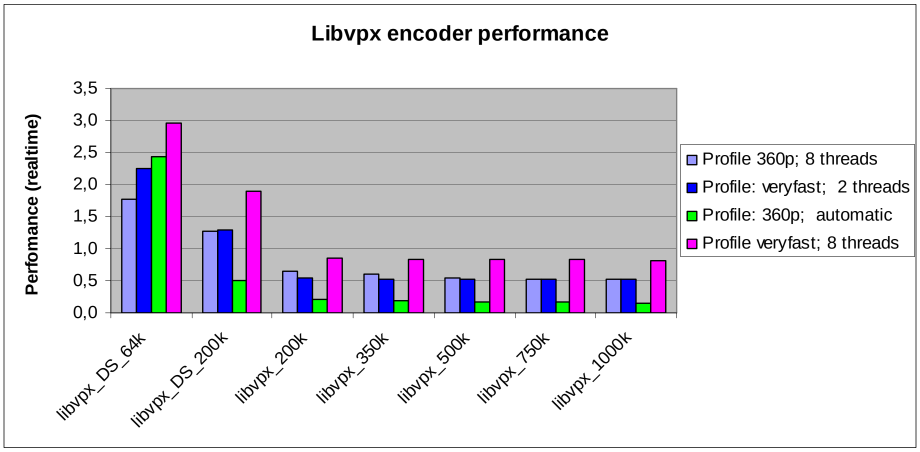


Figure 6. Encoding speed for the different profiles; DS = downscaled; n = normal preset.

To determine whether the encoder preset was the predominant factor in limiting the encoding speed, an additional encoding run was performed with the lipvpx *veryfast* preset. This preset however also did not achieve real-time encoding (see Figuur 7). For this reason, it is impossible to use WebM for live-streaming.

---

[36] Fragments with a longer duration where chosen, to account for the encoder initialization time, which might be relatively big for fragments with a duration of 10 seconds.



Figuur 7. Libvpx encoder performance, showing performance for 360p and veryfast preset, with different thread settings.

Further investigations to the cause of the low performance shows that libvpx does not use all possible CPU cores, while libx264 uses all available resources. This shows that the libvpx implementation is by far not as mature as libx264. For this reason, improvements are expected in the near future on more efficient encoding, better integration with FFmpeg and multi-threading support.

4.3 Conclusions

The obtained results lead to the following conclusions:

- With respect to picture quality WebM video format VP8 provides an alternative for H.264. The binary pairwise comparison showed that the picture quality of WebM is slightly worse than H.264 for the tested encoder implementations.
- With respect to encoding speed WebM currently provides no real alternative for H.264, because VP8 encoding is more than five times slower than x264 encoding, one of the best H.264 implementations). Real-time encoding could not be achieved with WebM, therefore providing live video services is currently not possible.
- The libvpx implementation of VP8 has problems to manage the target bit rates than libx264. For some fragments the target bit rate is exceeded more than 40%. From an operational point of view, it is important to have confidence in the settings.

These conclusions show that WebM is not as mature as H.264. Although Google invests in improvements, at this moment WebM does not provide an alternative for H.264 for the broadcaster’s web based video services.

Finally, we would like to thank the participants of the panel test: Kees Noort en Bram Tullemans (NPO), Anton Havekes, Joost de Wit, Martin Prins en Mark van Staalduinen (TNO). We also would like to thank Kamal Ahmed (TNO) for his support to setup this panel test.

5 Conclusions

This pilot resulted in relevant insights with respect to the practical usage of WebM. By taking into account the operational requirements of a broadcaster made the pilot more complex, but the obtained results are therefore more useful.

The most important conclusions with respect to the operational usage of WebM are:

- At this moment, it is impossible without extra development investments to provide WebM-based video services using open source components that fit the operational requirements of a broadcaster, because of the following issues:
  - With respect to encoding speed and maintaining target bit rates, the current WebM encoder library performs clearly worse than its x264 counterpart.
  - The encoding speed is too low for real-time WebM encoding, so live streaming is not possible.
  - FFmpeg does not support the required WebM formatting to enable live-streaming,
  - FFmpeg is not able to down mix the multi-channel audio to stereo, which is needed for the conversion of MXF files to WebM files. The broadcaster uses programs with multi-channel audio, which are stored in the MXF file format.
  - The quality of experience of WebM play out varies within the different browsers that support WebM. Therefore the QoE requirements of the broadcaster regarding startup delay and supported functionality are not fulfilled.
  - Important video platform functionalities as DRM, audience measurement and adaptive bit rate streaming are not available for playback of WebM in HTML5-enabled web browsers. Given the design of HTML5, it is a question whether DRM and audience measurement can be implemented with a comparable security and reliability level as proprietary solutions currently offer.
- The fact that Google invests in WebM gives confidence with respect to the required solutions for all the identified issues. So, WebM can be an alternative for H.264 in the near future. These facts support this conclusion:
  - The subjective panel test showed that the picture quality of WebM is close to the picture quality of H.264.
  - The reachability of WebM based on web browser market share has grown from 30% in December 2010 to nearly 50% in June 2011 in the Netherlands.
- It is probable that other parties besides Google have patents on WebM. These patents could obstruct the free and open usage of WebM. Google and MPEG LA are trying to make this situation transparent.

Appendix A: Encoder settings

For the video quality evaluation, the following FFmpeg settings are used:
-r 25 –vb [videobitrate] –an –threads 8 –s [resolution] –vpre [preset]

Parameter	**Meaning**
-r 25	25 frames per second
-vb [videobitrate]	The chosen video-bitrate
-an	No audio
-threads 8	Use 8 threads (the ecoding machine has 8 CPU cores)
-s [resolution]	The output resolution
-vpre [preset]	The video encoder preset (see table below). The presets are predefined encoder profiles, which are provided with FFmpeg.
-fmt [container]	The container format: *mp4* for H.264 *webm* for VP8

Meaning of presets:

Preset (encoder library)	**Encoder Flags**

360p
(libvpx)
	vcodec=libvpx
	g=120
	rc_lookahead=16
	quality=good
	speed=0
	profile=0
	qmax=63
	qmin=0
	vb=768k
	
	#ignored unless using -pass 2
	maxrate=1.5M
	minrate=40k

veryfast
(libvpx)
	vcodec=libvpx
	g=120
	rc_lookahead=16
	quality=good
	speed=8
	profile=0
	qmax=63
	qmin=0
	vb=768k
	
	#ignored unless using -pass 2
	maxrate=1.5M
	minrate=40k

fast (libx264)	coder=1 flags=+loop cmp=+chroma partitions=+parti8x8+parti4x4+partp8x8+partb8x8 me_method=hex subq=6 me_range=16 g=250 keyint_min=25 sc_threshold=40 i_qfactor=0.71 b_strategy=1 qcomp=0.6 qmin=0 qmax=69 qdiff=4 bf=3 refs=2 directpred=1 trellis=1 flags2=+bpyramid+mixed_refs+wpred+dct8x8+fast pskip wpredp=2 rc_lookahead=30
normal	coder=1 flags=+loop cmp=+chroma partitions=+parti8x8+parti4x4+partp8x8+partb8x8 me_method=hex subq=6 me_range=16 g=250 keyint_min=25 sc_threshold=40 i_qfactor=0.71 b_strategy=1 qcomp=0.6 qmin=10 qmax=51 qdiff=4 bf=3 refs=2 directpred=3 trellis=0 flags2=+wpred+dct8x8+fastpskip wpredp=2

Appendix B: System settings

Testsytem

All participants of the panel test performed the test on identical laptops with the following specifactoins:

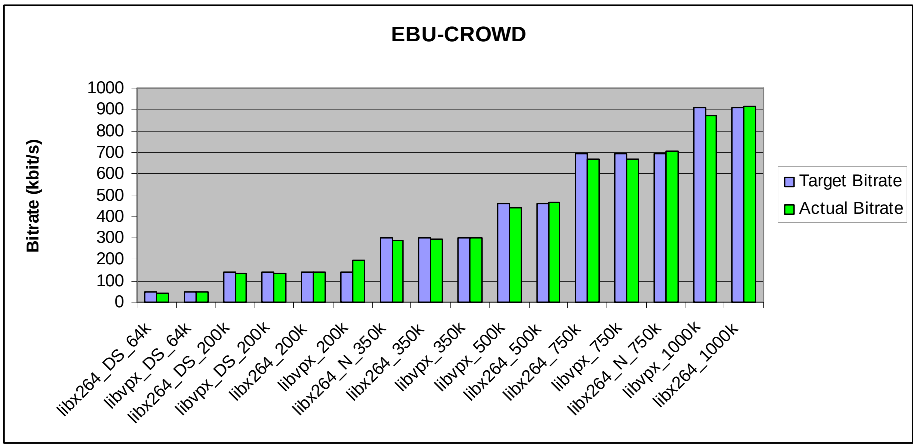
- Dell Latitude E6400
- OS: Windows XP Professional Version 2002 Service Pack 3
- Proccessor: Intel(R) Core(TM) 2 Duo CPU P8400 @ 2.26Ghz.
- Memory: 3,45GB RAM
- Video card: Intel 4 Series Express Chipset. Driver: 6.14.10.4964
- Maximum contrast.
- Media player: VLC Media Player 1.1.9 The Luggage. Windows 32 bit

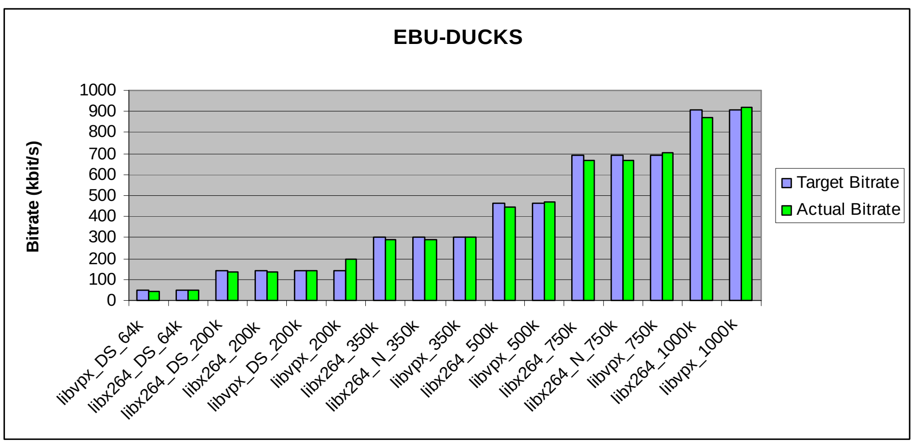
Encoding system

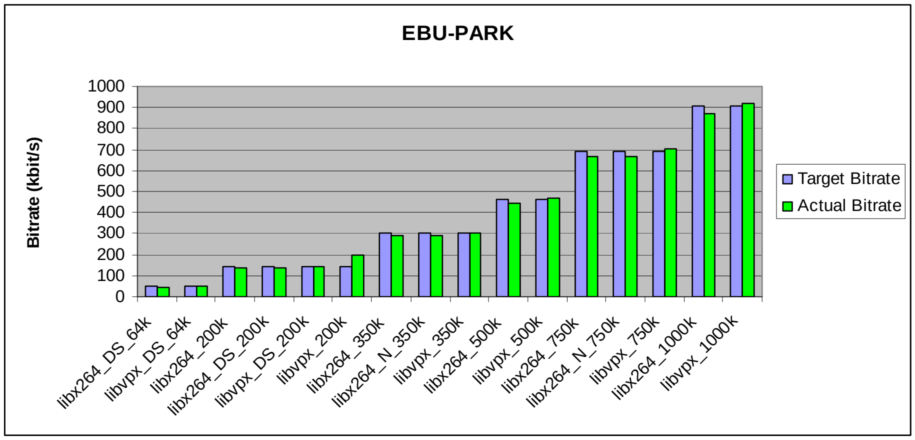
- Operating System: Ubuntu 10.10 nl
- Processor: AMD Opteron Processor 6128 8x2000Mhz
- Memory: 8GB ram

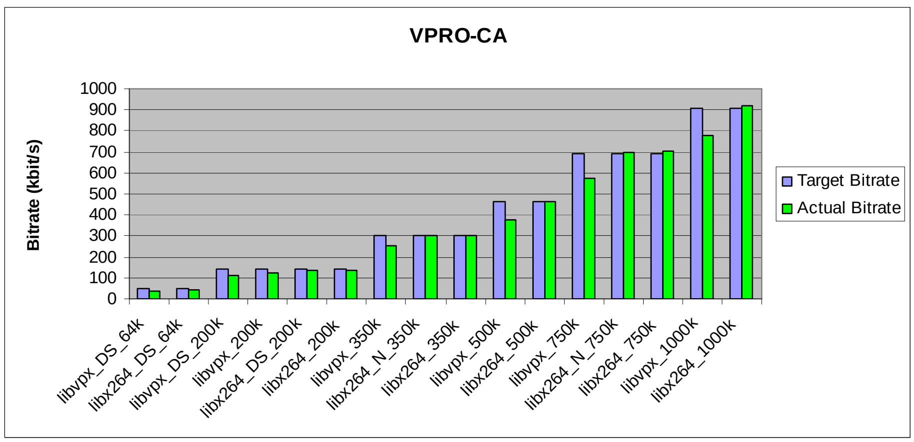
Appendix C: Maintenance target bit rates

These graphics present the differences between target and actual bit rate for the different encoding profiles. The difference between target and actual is maximal for libvpx_default_350k. The profiles are ordered with respect to the actual bitrates, so the ordering differs per fragment.






EBU-PARK
Bitrate (kbit/s)
1000
900
800
700
600
500
400
300
200
100
0
libx264_DS_64k
libvpx_DS_64k
libx264_200k
libx264_DS_200k
libvpx_DS_200k
libvpx_200k
libx264_350k
libx264_N_350k
libvpx_350k
libx264_500k
libvpx_500k
libx264_750k
libx264_N_750k
libvpx_750k
libx264_1000k
libvpx_1000k
Target Bitrate
Actual Bitrate


VPRO-CA
Bitrate (kbit/s)
1000
900
800
700
600
500
400
300
200
100
0
libvpx_DS_64k
libx264_DS_64k
libvpx_DS_200k
libvpx_200k
libx264_DS_200k
libx264_200k
libvpx_350k
libx264_N_350k
libx264_350k
libvpx_500k
libx264_500k
libvpx_750k
libx264_N_750k
libx264_750k
libvpx_1000k
libx264_1000k
Target Bitrate
Actual Bitrate