Towards Video on the Web with HTML5

François Daoust[1], Philipp Hoschka[1], Charalampos Z. Patrikakis[2], Rui S. Cruz[3], Mário S. Nunes[3], David Salama Osborne[4]

[1]W3C/ERCIM, Sophia-Antipolis, France
[2]School of Electrical and Computer Engineering, NTUA, Athens, Greece
[3]IST/INESC-ID/INOV, Lisbon, Portugal
[4] Atos Research & Innovation - Atos Origin, Madrid, Spain

E-mail: [1]fd@w3.org, ph@w3.org
[2]bpatr@telecom.ntua.gr
[3]rui.cruz@inesc-id.pt, mario.nunes@inesc-id.pt
[4]david.salama@atosresearch.eu

Abstract: Motivated by the revolution in the media industry brought by recent developments in video access and sharing, this paper investigates the future of internet video. We present new web standards such as HTML5, promising unified, simple and platform independent access to video files and streams, as well as novel techniques for adaptive video coding. We also analyze how these techniques can be used both over traditional client server and novel P2P based media distribution models. This analysis is followed by a description of the remaining challenges for making video a true citizen of the Web.

Keywords: HTML5, video, codec, streaming, adaptive, P2P, scalable coding

1 INTRODUCTION

Following the boom of peer to peer systems that have taken the lion's share of Internet traffic, Internet video is quickly becoming the main source of traffic on the net. According to CISCO [1], by 2014, global online video will approach 57 percent of consumer Internet traffic (up from 40 percent in 2010). The Web is already preparing to embrace these developments, paving the way for new standards.

HTML5 [2] is the core specification that shapes the next open Web platform. Although HTML5 is not a Web standard yet, latest versions of Web browsers have started to implement key parts of HTML5. In particular; most desktop Web browsers now support the **<video>** tag, a key initiative in HTML5 to integrate video on the Web.

The <video> tag opens the possibility to manipulate video within a regular Web page. Innovative user interfaces that mix video and other content can now be created without having to resort to dedicated plug-ins.

As of today, the <video> tag does not mandate support for specific codecs and streaming technologies though. Standardization work around these technologies is needed to fully realize the potential of video on the Web as well as to allow the use of Web technologies on TV and other devices that stream video.

This paper is structured as follows: In Section 2, we give more background on the situation for video on the Web. Section 3 focuses on video delivery techniques on the Web. Section 4 looks at the future.

2 BACKGROUND

2.1 Before HTML5

In the absence of a standard way to include video in an HTML page, video on the Web has been mostly the prerogative of browser-specific implementations or third-party plug-ins such as the VLC plug-in [3], Apple QuickTime or Adobe Flash Player. These plug-ins are activated through the use of the <object> tag in HTML.

Relying on third-party plug-ins to render the video works fine in a variety of use cases. It does come with drawbacks though, because the video is rendered in a black box from the Web browser's perspective.

Thus, CSS cannot be used to style the video, or to apply transformations. SVG cannot be used to apply masks and filters on the video. In short, visual effects such as the ones that appear in Figure 1 cannot be achieved using regular Web technologies.

Corresponding author: François Daoust, W3C/ERCIM, 2004 route des Lucioles 06410 Sophia-Antipolis, France, +33 1 77 21 16 85, fd@w3.org

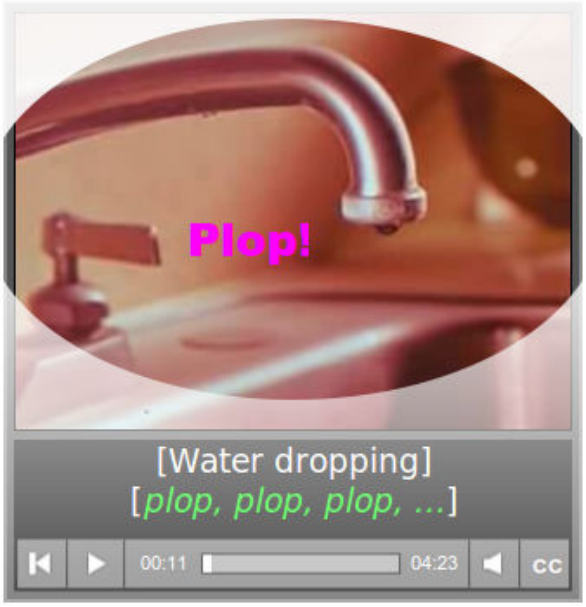

Figure 1: Example of a visual effect using standard Web technology hard to achieve when video is rendered via video plug-in

There is no standard way either to control the "black box" through JavaScript, and thus no way to change the look and feel of the video playback interface from within the rest of the page. Finally, these plug-ins are rarely available on devices that are not "regular" desktop computers, e.g. on mobile devices.

2.2 The <video> Tag to the Rescue

The <video> tag introduced by HTML5 alleviates these problems. Since video becomes a regular tag such as <p> or , it is directly integrated within the rest of the Web page. CSS may thus be used to style and transform the video box, as shown in Figure 2.

Figure 2: A <video> tag transformed through CSS in Firefox

The HTML5 specification also defines a standard API to access and control the video from JavaScript. This new API opens up the possibilities in terms of the user interface, as everything may now be done through regular HTML, SVG, CSS, and Javascript. Figure 3 shows a video player entirely developed in SVG and JavaScript by Philippe le Hégaret in W3C [4]. The text that appears at a specific time on top of the video are regular HTML paragraphs controlled through JavaScript and positioned through CSS.

Figure 3: Implementation of a video player interface in SVG

One of the advantages of sticking to Web technologies for networked media content such as video is that they were designed with accessibility in mind. Rich user interfaces can be made accessible easily, following the Web Content Accessibility Guidelines (WCAG) [5] and Accessible Rich Internet Application (ARIA) Authoring Practices [6].

2.3 Codecs and Containers

The HTML5 specification does not restrict the list of video formats and codecs that a browser may support, leaving the door open for future formats. The <video> tag features a fallback mechanism whereby alternative "sources" can be specified, each using a different format or codec. Should the Web browser not support the first format, it will try to use the second one, then the third one, and so on.

Corresponding author: François Daoust, W3C/ERCIM, 2004 route des Lucioles 06410 Sophia-Antipolis, France, +33 1 77 21 16 85, fd@w3.org

```
<video id="movie" width="640" height="480">
 <source src="video.mp4" type='video/mp4;
        codecs="avc1.42E01E, mp4a.40.2"' />
 <source src="video.webm"  type='video/webm;
        codecs="vp8, vorbis"' />
 <p>The video is available as <a href="video.mp4">H.264 in an
MP4 container</a>, or as <a href="video.webm">VP8 in a
WebM container</a>.</p>
</video>
```

Figure 4: Fallback mechanism of the <video> tag in HTML5

As of today, HTML5 does not mandate support for any specific codec and format container either. The situation among primary desktop browsers is currently – or will be in a near future – as follows:

- H.264 in an MP4 container: supported by Internet Explorer 9.0, Safari, and Google Chrome. Support for H.264 usually means support for the Baseline profile. In particular, this profile is the only one supported on mobile devices such as iPhone or Android.
- Theora in an Ogg container: supported by Firefox, Google Chrome, and Opera
- VP8 in a WebM container: supported by Firefox, Google Chrome, and Opera

As things stand, there will be no way to target all HTML5 capable browsers with one version of a video. Delivering video on the Web will today require at least two versions:

1) one version that uses the H.264 Baseline profile in an MP4 container
2) one version that uses VP8 in a WebM container, or Theora in an Ogg container.

Microsoft announced [7] that support for additional codecs could be added to Internet Explorer, provided they are installed on the operating system. Apple has not provided information on its plans. It is impossible to add support for additional codecs in Firefox that would be supported in the <video> tag. New codecs in Firefox can only be added via the <object> tag and plug-ins, which has the "black box" disadvantages already described.

The problems around codecs are caused by patents [8] and licensing fees. H.264 is a codec with clearly identified players and Intellectual Property Rights holders and a solid set of coders and decoders that take advantage of the underlying hardware. It does not come with a royalty-free license though. VP8 was released with a royalty-free license by Google early 2010, but unknown IPR holders may still surface. Theora has a similar problem.

2.4 Subtitles

Subtitles formats for HTML5 are still under discussion at the time of writing this paper. A format adapted from the SRT file format [9] to include styling information has been proposed. Other options based on the Timed Text Markup Language [10] or on SMIL text module [11] are still possible. It is clear, however, that HTML5 will include a subtitles format supported by each and every Web browser.

3 STREAMING VIDEO ON THE WEB

3.1 HTTP Progressive Delivery

The easiest way to deliver video on the Web is to use HTTP progressive delivery, in other words to serve video as any other content on the Web. Progressive delivery is supported by all Web browsers.

Using regular HTTP has three main advantages:

- A regular HTTP Web server may be used.
- Video delivery is transparent for firewalls. Other transport protocols usually require changing the settings on firewalls to let the content pass.
- Existing solutions to improve caching can be used with video as well. In particular, Content Delivery Networks (CDN) provided by companies such as Akamai may be used to distribute video content more efficiently to a potentially large audience.

Note that HTTP-based video delivery may actually *require* caching solutions and CDN support for videos with massive audience, to avoid a meltdown effect triggered by the parallel streaming of the same video to a huge number of clients (HTTP delivery is "unicast").

HTTP progressive delivery is well suited for short video clips where the user does not need to be able to jump forward and backward in the video. Jumping forward in the video requires that the client has received the video up to the targeted position. This is not practical in the case of longer video clips or movies.

Furthermore, there is no guarantee that the delivery rate will remain consistent over time. If the network throughput varies over time during the delivery of the video, playback will eventually stop.

Newer "intelligent" approaches to stream video on the Web that go beyond progressive delivery are being invented. Most of these methods still rely on HTTP though. Other protocols, e.g. DCCP or RTP/RTSP, will probably play a role in video delivery but are not envisioned as the main streaming methods for the <video> tag so far.

3.2 HTTP Streaming

HTTP Streaming is HTTP progressive delivery, coupled with a protocol used by the client to be able to request a given slice of the video. This feature lets users jump to a specific position in the video at any time. The current download is simply interrupted and another HTTP exchange between the client and the server is started.

Existing solutions are deployed in popular Web sites such as Youtube [12] or Vimeo [13]. Ongoing standardization work on Media Fragments URI [14] will create a standard syntax for constructing media fragment URIs that can be used over the HTTP protocol.

```
http://www.example.com/example.ogv?t=10,20
```

Figure 5: Example of a media fragment URI

Support for HTTP Streaming may need to be added to the HTTP Web server. In most cases, this is relatively

Corresponding author: François Daoust, W3C/ERCIM, 2004 route des Lucioles 06410 Sophia-Antipolis, France, +33 1 77 21 16 85, fd@w3.org

straightforward as the protocol relies on a couple of HTTP query parameters (see Figure 5 above).

HTTP Streaming is well suited for longer video clips. Content is still delivered progressively though, and not actually "streamed" in the traditional sense of the word (e.g. as in the "streaming media players" of the late 1990's/early 2000's). Thus, HTTP Streaming suffers from the same limitations as HTTP progressive delivery: it cannot adjust to changing network conditions during video delivery.

3.3 HTTP Adaptive Streaming

With Adaptive Streaming [15] the bitrate of the video is changed based on real-time conditions experienced by the video player. Adaptive Streaming lets the video degrade gracefully when the network cannot keep up with higher bitrates. This is particularly useful for the comfort of users when watching a movie, a long video or a live TV show, as network throughput often varies over time. It is essential to deliver video to devices whose primary focus is on video such as TV screens, or to devices that are connected to erratic and quickly changing networks such as mobile devices.

HTTP Adaptive Streaming uses HTTP as a transport protocol. The usual approach is to maintain copies of the video content encoded using different quality levels and sizes on the server. These copies are sliced into segments (chunks) of 2-10 seconds. When the client requests the video, it receives an index file (or manifest file) that describes the different segments and copies (quality/bitrate levels) available. It then fetches each segment using regular progressive download. The next segment is chosen based on the network conditions experienced during the playback of the current slice.

This technique is used by Apple in its HTTP Live Streaming protocol, described in an Informational Internet Draft [16], and implemented in Safari for MacOS X and in iOS for the iPod, iPhone and iPad. Microsoft IIS Smooth Streaming [17] and 3GPP Adaptive HTTP Streaming [18] (reused by the Open IPTV Forum) are other examples.

The format of the index file that describes the video segments varies depending on the solution. Apple used a regular M3U8 playlist textual format while Microsoft and 3GPP solutions are XML-based.

The index file may also be the container itself. The WebM container proposed by Google, derived from Matroska [19], could also be used to implement adaptive streaming solutions for instance.

A similar approach to the HTTP adaptive streaming technique can be followed using Scalable Video Coding (SVC) and Multiple Description Coding (MDC) technologies, specifically developed with adaptive streaming in mind. They can have an important role in the future of video delivery, although they are not yet supported by any of the primary Web browsers.

3.4 Peer-to-peer Video Delivery on the Web

Video may also be delivered using a peer-to-peer mechanism. As opposed to HTTP-based solutions, CDNs are not required in the case of peer-to-peer, as the idea is to take advantage of the number of users that watch videos concurrently to exchange video segments and improve the efficiency of the delivery.

On the Web, the Websocket API and protocol [20] [21] go some way towards providing a persistent two-way connection between peers, but the API cannot be used to establish connections between clients directly. For security reasons, connections have to go through HTTP servers. On top of that, the WebSocket API and protocol are currently limited to sending and receiving text frames. Binary data would need to be encoded as a regular string before they may be exchanged, which is definitely not optimal for video content. Support for binary frames should be added at some point in the future though.

Proper peer-to-peer connections on the Web were dropped from HTML5 for initial lack of interest from Web browser vendors and moved to a separate specification called HTML Device [22]. The editor of the specification issued a call for actions early July 2010 to have this work move forward.

Since traditional HTTP cannot be used to support peer-to-peer transmissions, video streaming between peers may use non-HTTP protocols. The H.264 SVC profiles could become prevalent for this sort of usage, as demonstrated by Google for its Gmail Video chat [23] or as proposed by the P2P-Next project for their NextShare platform [24]. Furthermore, MDC (Multiple Description Coding) which addresses the issue of content adaptation from the point of resilience and robustness rather than scalability, is another important candidate for supporting media streaming over P2P architectures. The inherent ability to use any of the decoded descriptions for the reconstruction of the video rather than decoding layers successively make it ideal for use under P2P architectures.

In the SARACEN project [25] the system architecture is planned as a Multi-Source HTTP client and server providing an advanced form or WebSeeding (HTTP based peer-to-peer downloading/uploading) [26], aimed to support layered video coding such as the H.264 SVC profile and the Multiple Description Coding. In the context of the project, the use of SVC and MDC in different scenarios for video streaming over P2P architectures will be evaluated, demonstrating the benefits of use of adaptive coding techniques both in situations where robustness is a key factor, but also in cases where adaptation from HD down to SD can be used to provide enhanced Quality of Experience.

4 FUTURE WORK

As we have seen in previous sections, the promotion of video as a first-class citizen on the Web both opens up new possibilities and creates challenges that need to be addressed in the future.

Corresponding author: François Daoust, W3C/ERCIM, 2004 route des Lucioles 06410 Sophia-Antipolis, France, +33 1 77 21 16 85, fd@w3.org

4.1 The Web on TV and Set-Top Boxes

The <video> tag brings video to the Web. The opposite is true as well: the <video> tag brings the Web to video. In other words, TV and set-top boxes may now take full advantage of Web technologies to create rich user interfaces based on:

- HTML5
- Access to device APIs, whose standardization is underway (Geolocation, Calendar, Media Capture, Contacts), to leverage the functionalities of the device and to react to user's preferences, as well as his physical and social environment
- The possibility to package Web applications as widgets that may be verified, signed, and installed as any other application.

More APIs, more specifically targeted at these kinds of equipment, may need to be defined. For instance, widgets on TV should be able to retrieve information about the TV channel that the user is currently watching.

4.2 Consolidation of HTTP Streaming

One of the key points that is still missing for the use of Web technologies within video equipment is a robust and standard way to stream video on the Web and adapt to changing network conditions in real time.

In the absence of agreement for common codecs and container formats on the Web, video streaming needs to be defined at a different level. This is the direction taken by most recent initiatives around HTTP Adaptive Streaming that describe the different segments of a video in an index (manifest) file separated from the content itself.

Consolidation of the existing index formats among players is needed to avoid running into a situation where streaming video on the Web requires more than one streaming mechanism server-side.

The standard format should take the form of a playlist format and should not impose or rely on the use of specific codecs for the video (and audio). To reach a global audience, this format should simply work *as-is* with a regular HTTP server, possibly completed with support for media fragments URIs.

4.3 Peer-to-peer Connections on the Web

Technical solutions can be found to enable the use of peer-to-peer connections within the Web browser sandbox. For example, security issues that arise when if peer is allowed to connect to another peer may be solved using a peer introduction mechanism within the control of the Web server.

On top of file exchange and video delivery, peer-to-peer is needed for video conferencing and real-time network games, as HTTP-based solutions are not reliable enough to handle these use cases.

Support for advanced graphic rendering through the WebGL [27] specification and background worker threads (Web Workers [28]) put the Web as a platform in a strong position for gaming in particular, provided peer-to-peer is possible.

Thus, the incentive to add peer-to-peer support in Web browsers will come from multiple fronts and a standard peer-to-peer interface is likely to emerge in a near future.

Acknowledgments

The research leading to these results has received funding from the European Union's Seventh Framework Programme (FP7/2007-2013) under grant agreement n°248687 - Open Media Web (OMWeb) and n° 248474 Socially Aware, collaboRative, scAlable Coding mEdia distributioN (SARACEN)

References

[1] Cisco Visual Networking Index (VNI) Forecast, 2009-2014
[2] HTML5, W3C Working Draft, 24 June 2010, Ian Hickson, http://www.w3.org/TR/html5/
[3] VLC Media Player plug-in for Firefox, http://addons.mozilla.org/fr/firefox/addon/14370/
[4] XHTML5 Video Player, Philippe le Hégaret, http://www.w3.org/2009/04/video-player.xhtml
[5] Web Content Accessibility Guidelines (WCAG) 2.0, W3C Recommendation, 11 December 2008, http://www.w3.org/TR/WCAG/
[6] Accessible Rich Internet Applications (WAI-ARIA) 1.0, W3C Working Draft, 15 December 2009, http://www.w3.org/TR/wai-aria/
[7] First blog post on HTML5 video in IE9 by Dean Hachamovitch, Microsoft, 3 May 2010, http://blogs.msdn.com/b/ie/archive/2010/05/03/follow-up-on-html5-video-in-ie9.aspx
[8] Second blog post on HTML5 video in IE9 by Dean Hachamovitch, Microsoft, 19 May 2010, http://windowsteamblog.com/windows/b/bloggingwindows/archive/2010/05/19/another-follow-up-on-html5-video-in-ie9.aspx
[9] Description of the SRT subtitles format, http://www.matroska.org/technical/specs/subtitles/srt.html
[10] Timed Text Markup Language (TTML) 1.0, W3C Candidate Recommendation, 23 February 2010, http://www.w3.org/TR/ttaf1-dfxp/
[11] Synchronized Multimedia Integration Language (SMIL 3.0), W3C Recommendation, 01 December 2008, http://www.w3.org/TR/SMIL/
[12] Youtube, http://youtube.com/
[13] Vimeo, http://vimeo.com/
[14] Media Fragments URI 1.0, W3C Working Draft, 24 June 2010, http://www.w3.org/TR/media-frags/
[15] R. S. Cruz, M. S. Nunes, and J. P. E. Goncalves, "A Personalized HTTP Adaptive Streaming WebTV," in Proceedings of the First International Conference on User Centric Media Workshop, UCMedia '09. Venice, Italy: ICST, Dec. 2009.
[16] HTTP Live Streaming, Informational Internet Draft, 5 June 2010, R. Pantos, http://tools.ietf.org/html/draft-pantos-http-live-streaming
[17] Microsoft IIS Smooth Streaming, http://www.iis.net/download/SmoothStreaming
[18] Presentation of 3GPP Adaptive HTTP Streaming by Ericsson Labs, https://labs.ericsson.com/apis/streaming-media/documentation/3gpp-adaptive-http-streaming-ahs
[19] Matroska media container, http://www.matroska.org/
[20] The Web Sockets API, W3C Working Draft, 22 December 2009, Ian Hickson, http://www.w3.org/TR/websockets/
[21] The WebSocket protocol, Internet Draft, 6 May 2010, Ian Hickson, http://tools.ietf.org/html/draft-hixie-thewebsocketprotocol
[22] HTML Device, Editor's Draft, 15 June 2010, Ian Hickson, http://dev.w3.org/html5/html-device/#peer-to-peer-connections
[23] Blog post on Gmail voice and video chat, Justin Uberti, 11 November 2008, http://juberti.blogspot.com/2008/11/say-hello-to-gmail-voice-and-video-chat.html
[24] P2P-Next project home page, http://p2p-next.org/
[25] R. S. Cruz, M. S. Nunes, C. Z. Patrikakis, N. C. Papaoulakis "SARACEN: A platform for adaptive, socially aware multimedia distribution over P2P networks," submitted to IEEE Globecom 2010 Workshop on Enabling the Future Service-Oriented Internet, Miami, FL, USA: Dec. 2010.

Corresponding author: François Daoust, W3C/ERCIM, 2004 route des Lucioles 06410 Sophia-Antipolis, France, +33 1 77 21 16 85, fd@w3.org

[26] SARACEN Consortium, “Socially Aware, collaboRative, scAlable Coding mEdia distributioN.” [Online]. Available: http://www. saracen-p2p.eu/

[27] WebGL Specification, Working Draft, 10 June 2010, Chris Marrin, http://cvs.khronos.org/svn/repos/registry/trunk/public/webgl/doc/spec/WebGL-spec.html

[28] Web Workers, W3C Working Drat, 22 December 2009, Ian Hickson, http://www.w3.org/TR/workers/

Corresponding author: François Daoust, W3C/ERCIM, 2004 route des Lucioles 06410 Sophia-Antipolis, France, +33 1 77 21 16 85, fd@w3.org