Accessibility for the HTML5 <video> element

Silvia Pfeiffer
Annodex Association
Sydney
Australia
+61 2 8012 0937
spfeiffer@acm.org

Conrad Parker
Kyoto University
Kyoto
Japan
+81 075 753 5385
conrad@dl.kuis.kyoto-u.ac.jpl

ABSTRACT
In this paper, we describe existing implementations for putting subtitles and captions alongside the HTML5 <video> tag inside Web pages and a proposal for standardizing such approaches, which will make them interoperable and easier to be processed by automated tools. Since video and audio are fundamental data types that any Web user will want to make use of nowadays – if young or old – if impaired or not – a standard means of providing accessibility to such fundamental data types is of utmost importance as part of the standardization process of the <video> and <audio> tags. The proposal is an outcome of a Mozilla grant and of extensive discussions within the Xiph accessibility group. Ideas discussed on the WHATWG mailing list have also been taken into account. Standardization work for this is ongoing.

Categories and Subject Descriptors
K.4.2 Social Issues: Assistive technology for persons with disabilities, H.5.1 Multimedia Information Systems, H.2.4 Systems / Multimedia databases

General Terms
Design, Human Factors, Standardization, Experimentation

Keywords
Video accessibility, Web video, captions, subtitles

1. INTRODUCTION
The new HTML <video> element (http://www.what-wg.org/specs/web-apps/current-work/#video) is seeing increasing implementation in Web browsers: Firefox, Opera and Safari have support for it. However, the mechanism for including accessibility data – even just for simple subtitles and captions, not to speak of audio annotations and sign language – does not yet have a agreed solution. We will outline use-cases both for including accessibility data within a video file format, and for distributing accessibility data as a separate file.

Most freely available subtitles for videos are distributed as text files independent of the binary video file. For example, the large online subbing community, composed mostly of non-English speaking subtitle authors and distributors, tends to use the simple SRT (SubRip) file format for sharing subtitles for video content. This text is then loaded into a video player in parallel to the binary video file and the video player synchronizes their presentation. The implementation of a Web application to manage textual representations of videos – such as subtitles, captions, transcripts, karaoke text, etc – is simplified by keeping such data in a database in an editable format rather than having it encapsulated inside a binary video file.

At the same time, accessibility data also needs to be addressed as part of binary video files for several reasons. Firstly, audio annotations and sign language are audio and video data respectively. They have to be made part of a multi-track video file to be provided in sync in Web video players. Secondly, when exchanging video files, users much prefer being able to handle a single video file that contains all its information inside itself over having to share a set of files that are e.g. a video and subtitles in different languages. Lastly, there are existing file formats such as mp4, 3gpp and QuickTime that already support such encapsulated accessibility data, so it is important to define how to deal with this from inside Web browser. Indeed, the current consensus for HTML5 is to store video accessibility data within the video file format, and to encourage implementation within the video decoding framework associated with that file format. For example, if you have a QuickTime video, your captions should be inside that video and your QuickTime framework (on Windows or Mac OS X) should be able to decode and display that content.

The video encapsulation format of interest to us is Ogg, since patent unencumbered and royalty-free usable audio and video codecs are available through the Ogg framework of tools and we see a necessity of having such codecs available for an open platform like the World Wide Web.

2. EXAMPLE SUBTITLE SUPPORT
In this section, we present three different existing example implementations of how to provide subtitles and captions for a <video>. These are also shown in the submitted demo. We further propose a more comprehensive specification for associating time-aligned text with <video> or <audio> - a specification that has the potential for standardization.

The first example of using subtitles with HTML5 <video> tags was implemented by Jan Gerber at http://v2v.cc/~j/jquery.SRT/. You will need Firefox 3.1 in order to view it. Jan is using JavaScript to load and parse an SRT file and map it into HTML and thus onto the screen. The syntax for associating the SRT file with the <video> looks like this:

```
<script type="text/JavaScript" src="jquery.js"></script>
<script type="text/JavaScript" src="jquery.SRT.js"></script>
```


W4A2009, April 20-21, 2009, Madrid, Spain. Co-Located with the 18th International World Wide Web Conference.

```
<video src="http://example.com/video.ogv" id="video" />
<div class="SRT" data-video="video"
      data-SRT="http://example.com/video.SRT" />
```

The SRT file is specified through the data-srt attribute of the tag into which the text is ultimately loaded. The SRT file is associated with the video through the data-video attribute. Attributes that start with "data-" are custom data attributes that HTML5 provides to store custom data private to the page or application, for which there are no more appropriate attributes or elements. This solution works perfectly fine, but requires custom JavaScript routines and custom "data-" attributes to do the conversion of individual subtitles from SRT to HTML and their timed display.

Through our work for Mozilla and the discussions within the Xiph accessibility group, a different specification for associating time-aligned textual data with <video> emerged. The first version of this proposal introduced a <text> sub-element to the <video> element with attributes category (for the type of time-aligned text this is), lang (for the language the time-aligned text is in), type (for the MIME type of the time-aligned text file), and src (for the URL of the file).

Michael Dale from Metavid.org implemented an example of this syntax in the bottom of the two videos at http://metavid.org/w/extensions/MetavidWiki/skins/mv_embed/example_usage/sample_timed_text.php . If you click on the "CC" button on the player and click on "Select Text Layers", you will see the different subtitles in English and Spanish. If you click onto a text element, the video will play from that offset. The syntax looks like this:

```
<video src="sample_fish.Ogg" poster="sample_fish.jpg">
  <text category="SUB" lang="en" type="text/x-SRT"
        default="true" title="english SRT subtitles"
        src="sample_fish_text_en.SRT">
  </text>
  <text category="SUB" lang="es" type="text/x-SRT"
     title="spanish SRT subtitles" src="sample_fish_
es.SRT">
  </text>
</video>
```

Inspired by Jan's implementation and the proposed syntax, Philippe Le Hegaret from the W3C Timed Text working group extended the current DFXP test suite to demonstrate use of the proposed syntax with DFXP and Ogg video inside the browser. The demo is at http://www.w3.org/2008/12/dfxp-testsuite/web-framework/START.html. To see the result, you will need Firefox 3.1. If you select the "HTML5 DFXP player prototype" as test player, you can click on the tests on the left and it will load the DFXP content. Philippe actually adapted Jan's JavaScript file for this. His syntax looks like this:

```
<video src="example.ogv" id="video" controls>
  <text lang='en' type="application/ttaf+xml"
        src="testsuite/Content/Br001.xml">
  </text>
</video>
```

Common to all these implementations is that they map the time-aligned text to a HTML tag. For DFXP the styling attributes are mapped to CSS. Thus, the data is made part of the webpage and displayed through traditional means. This will be the right way for any time-aligned text. Now all we need is a common representation of time-aligned text in HTML5.

3. TIME-ALIGNED TEXT

There are multiple problems with the above presented <text> element: firstly, it conflicts in name with the SVG <text> element; secondly, it maps an external file into a HTML file potentially creating security issues.

A proposal to take forward is to create an element with a secure browsing context similar to the iframe element that will take the external text file with styling and JavaScript information and display it into a transparent overlay that is secured away from the main HTML page. Such an element is proposed as <itext> at http://wiki.xiph.org/index.php/Timed_Divs_HTML#Direct_linking_on_a_HTML5_page. An example is:

```
<video src="http://example.com/video.ogv" controls>
  <itext category="CC" lang="en" src="caption.SRT"/>
  <itext category="SUB" lang="de" src="german.SRT"/>
  <itext category="SUB" lang="jp" src="japanese.SRT"/>
</video>
```

A reference implementation of this approach is currently under discussion with Mozilla. Also, the Chrome team recently provided a similar proposal to the WHATWG, so we may see progress in the support of time-aligned text for video in Web browsers soon. Both, Mozilla and Google are focusing their efforts on SRT.

4. OGG ACCESSIBILITY

For delivering synchronized audio descriptions or sign language video, Ogg is capable of creating multi-track audio and video files. Audio descriptions should preferably be encoded in Speex, while sign language would be in Theora. In addition, an extension to the Ogg Skeleton headers will be necessary to indicate which tracks carry which type of data. Once this is done, Ogg is fully capable of meaningfully delivering audio annotations and sign language.

Additionally, Ogg requires a recommended means of encapsulating time-aligned text to provide in-line subtitles and captions. Multiple solutions have been specified, including a direct embedding of SRT into Ogg. The currently preferred solution that has an implementation and patches for video players is Ogg Kate (http://wiki.xiph.org/index.php/OggKate), an overlay codec. Ogg Kate can already use SRT as an input and output file. Ogg Kate will also be shown in the submitted demo.

5. NEXT STEPS

The SRT format is the simplest of all time-aligned text formats. Support for SRT in synchronization with <video> is well on its way. However, there are many more needs for time-aligned text in synchronization with <video>. For example, good captions provide formatting, such as different colors for different speakers, or left- and right-aligned text, or placement of text at the top part of the screen when it would obstruct in-video text.

Further, the possibilities of time-aligned text go beyond mere subtitles and caption – text categories such as karaoke text, ticker text, music lyrics, speech bubbles, DVD style chapter markers, transcripts, or clickable annotations can and should be supported in similar ways to subtitles and captions.

To further understand the requirements of such other categories of time-aligned text, the next activity in video accessibility includes the creation of a collection of example files for these categories. Both, the Ogg-embedded and the <itext> approaches will then need to be extended to allow support for these categories of text.