SWAD−Europe Deliverable 9.1: Visualisation and Accessibility – initial analysis

Project name:
> Semantic Web Advanced Development for Europe (SWAD-Europe)

Project Number:
> IST-2001-34732

Workpackage name:
> 9 Visualisation and Accessibility

Workpackage description:
> ☞http://www.w3.org/2001/sw/Europe/plan/workpackages/live/esw-wp-9.html

Deliverable title:
> 9.1 Visualisation and Accessibility

URI:
> ☞http://www.w3.org/2001/sw/Europe/reports/w3c_note_sw_accessibility

Authors:
> Charles McCathieNevile, W3C

Abstract:
> This report surveys areas in which the Semantic Web may be able to support increased accessibility of the Web for people with disabilities. It is a position paper which will list ideas that will be developed to the point where they can be offered to the Web Accessibility Initiative.

Status:
> Completed report: 2003-02-01.
>
> This document has not been formally reviewed by the Web Accessibilty Initiative (WAI). It does not reflect a consensus, and techniques outlined in this document may introduce new accessibility problems. It is conceived as a "position paper". A further report to be produced as part of ☞Work package 9: Visualisation and Accessibility will document the response of working groups within WAI to the proposals in this report. When that new report is published (expected in January 2004) readers should refer to that instead, and this report will be updated to reflect the availability of the new document.
>
> Comments on this document are welcome and should be sent to the ☞public-esw@w3.org list, archived at ☞http://lists.w3.org/Archives/Public/public-esw/. General discussion of accessibility issues should take place on the W3C WAI Interest Group list ☞w3c-wai-ig@w3.org, archived at ☞http://lists.w3.org/Archives/Public/w3c-wai-ig/. Further information on the ☞Web Accessibility Initiative and further resources on Web accessibility for people with disabilities are available at http://www.w3.org/WAI

Contents

1 Introduction

This report is part of ☞SWAD-Europe ☞Work package 9: Visualisation and Accessibility.
> For those in a hurry: go straight to the ☞theme summary section

Scope - This report outlines some preliminary suggestions for ways in which the Semantic Web may further the goals of the Web Accessibility Initiative (WAI) in making the Web accessible to all people, regardless of disability. It is primarily focused on four of WAI's 5 areas of work:

Guidelines
> WAI produces four Guidelines specifications:

> Web Content Accessibility Guidelines [WCAG1, WCAG2]
>> This specification outlines requirements for content to ensure that it is as accessible as possible to all

people.

Authoring Tool Accessibility Guidelines [ATAG1, ATAG2]

> This specification outlines requirements for tools used to produce Web Content, to ensure that they make it possible for all people to produce content for the Web, and to ensure that tools support the creation of content that is accessible (meets the requirements of WCAG), even if the author does not have any prior knowledge of accessibility requirements. ATAG 1.0 was published as a W3C Recommendation on 2 February 2000. Work is underway on ATAG 2.0, which is expected to clarify some minor issues and align with WCAG 2.0.

User Agent Accessibility Guidelines [UAAG]

> This specification outlines requirements for user agents such as Web browsers, Multimedia players and plug-ins, to ensure that they can be used to provide the best possible access to Web content for people regardless of disability. UAAG 1.0 was published as a W3C Recommendation on 17 December 2002.

XML Accessibility Guidelines [XAG]

> This draft specification outlines requirements for new markup languages to ensure that it is possible to create and use content written in those languages regardless of disability. This is a W3C Working draft.

Tools

> The Evaluation and Repair Tools working group coordinates development of tools designed to improve accessibility. There are two major focuses - evaluation of the accessibility of Web Content, and improving its accessibility both as an author and as a user.

Format Review

> The Protocols and Formats working group of WAI works to ensure that W3C specifications support accessibility to the greatest extent possible.

Education and Outreach

> The Education and Outreach working group produces documents, educational materials, and develops and coordinates presentations designed to help people understand why and how to improve the accessibility of their Web Content or Web-related tools.

The fifth area of WAI work is providing some coordination for Research and Development in areas related to, or beneficial to accessibility. This work package has provided some input to that work, and this report may be used as further input, but it is beyond the general scope of this report to provide inut specifically to that group.

Structure - This report is structured as a list of Semantic Web technologies that might provide useful techniques for the work outlined above. The suggestions are listed according to the area for which they relevant, following the order above. In the case of specifications which have a current Recommendation and a version 2.0 working draft (WCAG and ATAG) suggestions are given for the current Recommendation.

Since many techniques outlined are relevant to several areas, they are named each time, with a cross reference to a single explanation of the technique. Where appropriate, external documents explaining the techinque may be referred to or linked to as the primary explanation.

2 Background

The Web can be understood as an information warehouse, with various types of browsers and user agents used to access it. It is also the host for a number of services - the same user agents provide the "front desk" for interacting with the services available. As the Web has grown in popularity many services are only available, or primarily available, online. For people with disabilities this has offered the opportunity to use services which they could not use in the "physical" world, but it has also produced new problems of access. These are discussed more fully in documents produced by the Web Accessibility Initiative (WAI) of W3C and in other areas.

Many of the problems introduced appear to be opportunities for useful application of the Semantic Web. The following proposals have been put together by people working in the SWAD-E project, and have not been endorsed as appropriate or useful solutions. Further collaboration between WAI and the SWAD-E project will identify which of the techniques suggested below merit further development and implementation.

3. Guidelines – Web Content Accessibility Guidelines

The Web Content Accessibility Guidelines 1.0 was published as a W3C Recommendation on 5 May 1999. It consists of 65 checkpoints divided into three levels of priority and 14 general guidelines. These are the fundamental requirements for user agents to be able to present content in different ways to meet the needs of different users, and they are the requirements for the content that authoring tools need to guide authors to produce.

Work is now underway on WCG 2.0 [☞WCAG2] which will clarify the way the Guidelines apply to a wide range of Web Content. It is expected that only a few requirements will change, as a result of changes in the state of the art since the publication of WCAG 1.0. There is a mapping published by the working group between WCAG 1.0 and WCAG 2.0 drafts. The following is based on WCAG 1.0. New requirements introduced by WCAG 2.0 (for example for newer technologies) will be discussed as part of collaboration with the Working Group. The focus of this work will be to identify techniques that justify and require further development in deliverable 9.2 of the SWAD-Europe project.

WCAG 1.0 - ☞**Checkpoint 1.1 Provide a text equivalent for every non-text element**

Annotea associated alternatives
> Annotea is an RDF-based protocol that allows for third-party annotations of any Web Resource.

Jpeg embedded alternatives - RDFPic, EXIF, XMP
> RDFPic allows authors to embed data about the picture within the picture itself. This can include a description of the picture, and functions for which the picture is suitable.

SVG Overlay - image annotation, Patrick Roth's work
> The SVG format allows for multiple levels of description. It also allows for a bitmapped image (jpeg or png formats are required by the specification, and gif images are commonly understood by tools) to be included in the image. This provides the possibility of overlaying an image with small invisible regions which can be navigated, and which provide descriptions of parts of the image.

SVG Metadata - network example
> The SVG format allows information about an image or part(s) of an image to be directly embedded within the metadata element.

☞**Checkpoint 1.2** **Provide redundant text links for each active region of a server-side image map.**

☞**Checkpoint 1.3** **Until user agents can automatically read aloud the text equivalent of a visual track, provide an auditory description of the important information of the visual track of a multimedia presentation.**

☞**Checkpoint 1.4** **For any time-based multimedia presentation (e.g., a movie or animation), synchronize equivalent alternatives (e.g., captions or auditory descriptions of the visual track) with the presentation.**

SMIL/SVG/Annotea
> Because the Annotea system uses URI references including Xpointers, and W3C multimedia formats such as SMIL and timed-text use XML as their basis, it is possible to make annotations which refer to particular timestamps. In addition, if the annotea context property allows for RDF content very powerful mechanisms can be made available for referring to a part of a time-based presentation. One approach to this is to create a vocabulary to deal with timing that has a clear mapping to SMIL, similar to Jim Ley's vocabulary for regions of an image which maps to SVG [☞JImAnno]].

Visual versions of text or audio systems
> Languages such as timed text (under development) and VoiceXML specify mechanisms for audio or text-based interface design. However it is possible, by annotating the components of a VoiceXML application or a timed text track, to provide alternatives in the form of graphic symbols or sign-language captions which could be used by user agents to represent the application using these formats.

☞**Checkpoint 1.5** **Until user agents render text equivalents for client-side image map links, provide redundant text links for each active region of a client-side image map**

☞**Checkpoint 2.1** **Ensure that all information conveyed with color is also available without color, for example from context or markup.**

☞**Checkpoint 2.2** **Ensure that foreground and background color combinations provide sufficient contrast when viewed by someone having color deficits or when viewed on a black and white screen. [Priority 2 for images, Priority 3 for text].**

☞**Checkpoint 3.1** **When an appropriate markup language exists, use markup rather than images to convey information.**

Annotating images / closed binary formats with pointers to equivalent pieces in more open formats can allow for CC/PP solutions to select provide the best option a user can handle. See also 6.3, 8.1, etc

☞**Checkpoint 3.2** **Create documents that validate to published formal grammars.**

☞**Checkpoint 3.3** **Use style sheets to control layout and presentation.**

☞**Checkpoint 3.4** **Use relative rather than absolute units in markup language attribute values and style sheet property values**

☞**Checkpoint 3.5** **Use header elements to convey document structure and use them according to specification.**

See also XAG checkpoint 3.2 - documenting navigable structures, WCAG checkpoint 9.4 provide "tab" order

☞**Checkpoint 3.6** **Mark up lists and list items properly.**

☞**Checkpoint 3.7** **Mark up quotations. Do not use quotation markup for formatting effects such as indentation**

☞**Checkpoint 4.1 Clearly identify changes in the natural language of a document's text and any text equivalents (e.g., captions)**

Annotating "alt-text"
>One of the most common methods of putting text alternatives on the web is via HTML's alt attribute. However HTML does not provide a mechanism for identifying the language of particular attributes, let alone parts of an attriute value. This can be achieved through the use of annotation systems such as annotea or information included within a document.

☞**Checkpoint 4.2 Specify the expansion of each abbreviation or acronym in a document where it first occurs**

ILS approach
>The use of glossaries to provide possible expansions, combined with the ability to compare the possible entries against the domain of a document to "guess" the most probable, can help. It is likely to be more useful in an authoring tool, where the author can ensure that a particular explansion is the corect one.

☞**Checkpoint 4.3 Identify the primary natural language of a document.**

☞**Checkpoint 5.1 For data tables, identify row and column headers.**

☞**Checkpoint 5.2 For data tables that have two or more logical levels of row or column headers, use markup to associate data cells and header cells.**

☞**Checkpoint 5.3 Do not use tables for layout unless the table makes sense when linearized. Otherwise, if the table does not make sense, provide an alternative equivalent (which may be a linearized version)**

☞**Checkpoint 5.4 If a table is used for layout, do not use any structural markup for the purpose of visual formatting.**

EARL
>Annotating a table as a visual layout table can help evaluation tools. It can also be used to identify things which could be replaced in a service that convers layout to a more appropriate format - like tablin [TABLIN] but poducing CSS or SVG.

☞**Checkpoint 5.5 Provide summaries for tables.**

☞**Checkpoint 5.6 Provide abbreviations for header labels.**

☞**Checkpoint 6.1 Organize documents so they may be read without style sheets. For example, when an HTML document is rendered without associated style sheets, it must still be possible to read the document.**

☞**Checkpoint 6.2 Ensure that equivalents for dynamic content are updated when the dynamic content changes.**

☞**Checkpoint 6.3 Ensure that pages are usable when scripts, applets, or other programmatic objects are turned off or not supported. If this is not possible, provide equivalent information on an alternative accessible page.**

Semantically described services
>The use of Semantically rich descriptions for services on the Web can include machine-processable information about alternative versions of a service that may be appropriate to a particular delivery context. This can allow searching for more accessible versions of a script or applet being used, or directing useers directly to an alternative.

☞**Checkpoint 6.4 For scripts and applets, ensure that event handlers are input device-independent**

☞**Checkpoint 6.5 Ensure that dynamic content is accessible or provide an alternative presentation or page.**

Annotation of controls
>For interactive elements, knowing what a control does is important, but too much detail in a page can make it harder to use. Some languages, such as Xforms, provide for contect-sensitive help, but this can also be added through Annotea or similar systems where it is not already available.

☞**Checkpoint 7.1 Until user agents allow users to control flickering, avoid causing the screen to flicker.**

☞**Checkpoint 7.2 Until user agents allow users to control blinking, avoid causing content to blink (i.e., change presentation at a regular rate, such as turning on and off)**

☞**Checkpoint 7.3** **Until user agents allow users to freeze moving content, avoid movement in pages**

☞**Checkpoint 7.4** **Until user agents provide the ability to stop the refresh, do not create periodically auto-refreshing pages.**

☞**Checkpoint 7.5** **Until user agents provide the ability to stop auto-redirect, do not use markup to redirect pages automatically. Instead, configure the server to perform redirects.**

☞**Checkpoint 8.1** **Make programmatic elements such as scripts and applets directly accessible or compatible with assistive technologies.**

☞**Checkpoint 9.1** **Provide client-side image maps instead of server-side image maps except where the regions cannot be defined with an available geometric shape.**

☞**Checkpoint 9.2** **Ensure that any element that has its own interface can be operated in a device-independent manner.**

☞**Checkpoint 9.3** **For scripts, specify logical event handlers rather than device-dependent event handlers.**

☞**Checkpoint 9.4** **Create a logical tab order through links, form controls, and objects.**

☞**Checkpoint 9.5** **Provide keyboard shortcuts to important links (including those in client-side image maps), form controls, and groups of form controls.**

Site Mapping
> Automatically generated site maps can be used to identify resources that are linked to from many parts of the site as good candidates for a shortcut.

☞**Checkpoint 10.1** **Until user agents allow users to turn off spawned windows, do not cause pop-ups or other windows to appear and do not change the current window without informing the user.**

☞**Checkpoint 10.2** **Until user agents support explicit associations between labels and form controls, for all form controls with implicitly associated labels, ensure that the label is properly positioned**

☞**Checkpoint 10.3** **Until user agents (including assistive technologies) render side-by-side text correctly, provide a linear text alternative (on the current page or some other) for *all* tables that lay out text in parallel, word-wrapped columns.**

☞**Checkpoint 10.4** **Until user agents handle empty controls correctly, include default, place-holding characters in edit boxes and text areas.**

☞**Checkpoint 10.5** **Until user agents (including assistive technologies) render adjacent links distinctly, include non-link, printable characters (surrounded by spaces) between adjacent links.**

☞**Checkpoint 11.1** **Use W3C technologies when they are available and appropriate for a task and use the latest versions when supported**

TR Automation
> The availability of an RDF version of the W3C's specifications catalogue allows searching for newer versions of specifications than the one proposed for use, and in some cases automatic updating services are possible.

Schema Annotation
> The use of annotation to decsribe the purpose of various modules (see also ☞XAG checkpoint 2.9) allows searching for appropriate modules for a particular task.

☞**Checkpoint 11.2** **Avoid deprecated features of W3C technologies.**

Schema Annotation
> Annotating elements of schemas which are new, obsolete or deprecated, or render sections of earlier specifications obsolete would allow for some automatic testing of this checkpoint.

☞**Checkpoint 11.3** **Provide information so that users may receive documents according to their preferences (e.g., language, content type, etc.)**

CC/PP
> This is a framework designed to allow servers to communicate with a user's client and determine their needs and preferences, then attempt to provide the most appropriate version of content according to those needs or preferences

EARL
> This allows descriptions of how particular pieces of content meet requirements. It can be used in a system

which dynamically generates content in order to select a version of some information which meets some specified requirements - for example in conjunction with CC/PP.

☞**Checkpoint 11.4 If, after best efforts, you cannot create an accessible page, provide a link to an alternative page that uses W3C technologies, is accessible, has equivalent information (or functionality), and is updated as often as the inaccessible (original) page.**

☞**Checkpoint 12.1 Title each frame to facilitate frame identification and navigation.**

☞**Checkpoint 12.2 Describe the purpose of frames and how frames relate to each other if it is not obvious by frame titles alone.**

Site Mapping
> The navigation paths and combinations of frames - which links can lead to which collections of framesets, and therefore what are the "obvious" pathways could be developed by extending automated site-mapping systems to be aware of framesets.

☞**Checkpoint 12.3 Divide large blocks of information into more manageable groups where natural and appropriate.**

☞**Checkpoint 12.4 Associate labels explicitly with their controls.**

Extending HTML
> In HTML, form controls can be explicitly associated with a label, but this is a one to one mapping. There are mechanisms in certain cases for associating a label with a group of controls (e.g. fieldset). But it is not possible, for example, to associate an HTML link explicitly with a particular part of its immediate context, nor to associate a group of controls with two or more labels in the general case. (It is commonly achieved by using table markup to associate headers as labels, but this is not always appropriate).

☞**Checkpoint 13.1 Clearly identify the target of each link.**

RDF about
> RDF information describing a particular URI reference can be used to provide information about a link target which is not directly included in a page. For example having multi-lingual titles or descriptions of the resource linked to, including visual representations such as sign language. These can be used in systems such as IRC with the annotea protocol, or presented as (possibly visual) tool-tips by advanced user agents.

☞**Checkpoint 13.2 Provide metadata to add semantic information to pages and sites.**

In particular CC/PP and EARL type approaches.

☞**Checkpoint 13.3 Provide information about the general layout of a site (e.g., a site map or table of contents)**

Site mapping

☞**Checkpoint 13.4 Use navigation mechanisms in a consistent manner.**

Site mapping

☞**Checkpoint 13.5 Provide navigation bars to highlight and give access to the navigation mechanism.**

Site mapping

☞**Checkpoint 13.6 Group related links, identify the group (for user agents), and, until user agents do so, provide a way to bypass the group.**

☞**Checkpoint 13.7 If search functions are provided, enable different types of searches for different skill levels and preferences.**

Semantically enhanced searches
> Plain text searching in a page is a function normally provided by a browser. There are commercially available and free systems for providing plain-text indexing of a site. These systems can be augmented with Semantic Web-based searching to allow searches for specific types of information, etc.

Site Mapping
> Tools can crawl sites and determine the graph defined by links between various parts of the site. When collected as RDF this information can then be used with RDF visualisation tools such as GraphViz or RDF Author to provide a visual map of the site.

☞**Checkpoint 13.8 Place distinguishing information at the beginning of headings, paragraphs, lists, etc.**

☞**Checkpoint 13.9 Provide information about document collections (i.e., documents comprising multiple pages.).**

Cataloguing metadata
> The use of cataloguing metadata can allow the automatic generation of the information required to meet this checkpoint.

☞**Checkpoint 13.10 Provide a means to skip over multi-line ASCII art**

☞**Checkpoint 14.1 Use the clearest and simplest language appropriate for a site's content.**

See ILS method

☞**Checkpoint 14.2 Supplement text with graphic or auditory presentations where they will facilitate comprehension of the page.**

See 1.1 for annotations, and ILS method

☞**Checkpoint 14.3 Create a style of presentation that is consistent across pages.**

Site mapping
> Using RDF to collect information about the pages in a particular collection can provide a list of pages to check for a tool that compares the presentation information (style sheets used, basic structure of pages, etc).

4. Guidelines – Authoring Tool Accessibility Guidelines (ATAG)

ATAG 1 - ☞**Checkpoint 1.1 Ensure that the author can produce accessible content in the markup language(s) supported by the tool.**

Image annotation
> In developing real-time interactive image editing tools, image annotation can provide the ability to build images using previously described components, or to add descriptions as new parts of images are added.

☞**Checkpoint 1.2 Ensure that the tool preserves all accessibility information during authoring, transformations, and conversions.**

Annotea
> Some accessibility information cannot readily be encoded in some formats. One technique for maintaining information about a particular part of a document is to store it as an annotea annotation

Images
> In most image formats there is a method for storing comments, which can be used to store various types of accessibility information as RDF (see also ideas for WCAG checkpoint 1.1). Being able to extract such information can be helpful

Add information back
> Where accessibility information has been stored as an external annotation, it may be possible to include that information directly in a resource.

☞**Checkpoint 1.3 Ensure that when the tool automatically generates markup it conforms to the *W3C*'s Web Content Accessibility Guidelines 1.0 [☞WCAG1].**

☞**Checkpoint 1.4 Ensure that templates provided by the tool conform to the Web Content Accessibility Guidelines 1.0 ☞[WCAG1].**

EARL
> Tools can use EARL reports to ensure meeting this checkpoint while making online content available as a template.

☞**Checkpoint 2.1 Use the latest versions of *W3C* Recommendations when they are available and appropriate for a task.**

TR Automation
> The W3C's technical reports list is now available as RDF, including information about the history of documents. If a given format is selected this can be used to find the latest available version.

☞**Checkpoint 2.2 Ensure that the tool automatically generates valid markup.**

☞**Checkpoint 2.3 If markup produced by the tool does not conform to W3C specifications, inform the author.**

☞**Checkpoint 3.1 Prompt the author to provide equivalent alternative information (e.g., captions, auditory descriptions, and collated text transcripts for video).**

See also ☞ATAG checkpoint 3.5, WCAG checkpoint 1.1

Prompting help
> Where there are annotations about alternative versions of a media object these can be used to propose a value to the author.

Glossary help
> RDF-based dictionary services can be used to offer expansions for acronyms. Extensions of this functionality can provide likely contexts, or rank alternatives by likelihood according to contextual information.

☞**Checkpoint 3.2 Help the author create structured content and separate information from its presentation.**

☞**Checkpoint 3.3 Ensure that prepackaged content conforms to the Web Content Accessibility Guidelines 1.0 ☞[WCAG1].**

☞**Checkpoint 3.4 Do not automatically generate equivalent alternatives. Do not reuse previously authored alternatives without author confirmation, except when the function is known with certainty.**

Image annotation
> LIFT for Dreamweaver (and other tools) include a functionality for classifying images. This can be used to determine that certain images are spacers, and a null alternative is appropriate.

☞**Checkpoint 3.5 Provide functionality for managing, editing, and reusing alternative equivalents for multimedia objects.**

Image Annotation
> See for example the Amaya SVG library as a way of implementing this in an authoring tool. The LIFT for Dreamweaver tool also collects images according to their type.

☞**Checkpoint 4.1 Check for and inform the author of accessibility problems.**

EARL
> The use of EARL to record accessibility problems found can be integrated with information from the author, to provide a more accurate report. An example of this is implemented in the open-source tool WAI-Nu, developed in part with support from the SWAD-Europe project.

☞**Checkpoint 4.2 Assist authors in correcting accessibility problems.**

Extending EARL.
> Extending EARL tools to identify the particular way in which a test is failed would allow tools to track information on repairs that can be applied for that class of failure. This could be used to integrate different tools or tool modules into a workflow to produce a result benefitting from the syngergies available.

☞**Checkpoint 4.3 Allow the author to preserve markup not recognized by the tool.**

Round-tripping
> A tool may need to change unrecognised markup as it makes changes to a document - for example to maintain code validity. Storing information about the changes made can be used to allow the author to reconstruct the original code, or for a different tool that understands the markup to do so. A simple vocabulary constructed around the diff utility would seem a good candidate strategy.

☞**Checkpoint 4.4 Provide the author with a summary of the document's accessibility status.**

EARL
> This can be used to provide reporting of current status in an authoring tool. (This technique is demonstrated in tools such as AccVerify or LIFT which integrate with authoring tools).

☞**Checkpoint 4.5 Allow the author to transform presentation markup that is misused to convey structure into structural markup, and to transform presentation markup used for style into style sheets.**

☞**Checkpoint 5.1 Ensure that functionality related to accessible authoring practices is naturally integrated into the overall look and feel of the tool.**

Reading XAG

The XAG requirements for documented techniques for accessibility may give rise to RDF or RDF-compatible information that can be presented to the author when using a particular feature of a tool or language. See also ☞ATAG Checkpoint 6.2

☞**Checkpoint 5.2 Ensure that accessible authoring practices supporting Web Content Accessibility Guidelines 1.0 ☞[WCAG1] Priority 1 checkpoints are among the most obvious and easily initiated by the author.**

☞**Checkpoint 6.1 Document all features that promote the production of accessible content.**

☞**Checkpoint 6.2 Ensure that creating accessible content is a naturally integrated part of the documentation, including examples.**

☞**Checkpoint 6.3 In a dedicated section, document all features of the tool that promote the production of accessible content.**

Document Metadata

If the help documentation includes appropriate metadata, features that promote the production of accessible content can be collected together in a single view as required. This is particularly relevant to Content Management Systems which are self-describing.

☞**Checkpoint 7.1 Use all applicable operating system and accessibility standards and conventions.**

☞**Checkpoint 7.2 Allow the author to change the presentation within editing views without affecting the document markup.**

☞**Checkpoint 7.3 Allow the author to edit all properties of each element and object in an accessible fashion.**

☞**Checkpoint 7.4 Ensure that the editing view allows navigation via the structure of the document in an accessible fashion.**

☞**Checkpoint 7.5 Enable editing of the structure of the document in an accessible fashion.**

Implementing XAG

for formats which are lacking in important structure, RDF can be used top describe the underlying structure of the document. This approach is essentially that used in "external markup" systems.

☞**Checkpoint 7.6 Allow the author to search within editing views.**

5. Guidelines – User Agent Accessibility Guidelines (UAAG)

The User Agent Accessibility Guidelines describe requirements for "user agents" - browsers, media players and plugins, and user interfaces in general. This includes "web applications" - intereactive web content, which functions in some way as a user agent. The User Agent Accessibility Guidelines 1.0 became a W3C recommendation on 21 December 2002.

The fact that some user agents, such as the Mozilla browser and the Amaya editor/browser already incorporate an RDF parser (and in Amaya's case an Annotea user interface as a default, with one also available for Mozilla and other user agents) suggests that many of the following techniques could be relatively eeasy to implement as strategies to reach conformance with the requirements of these guidelines.

Guideline 1. Support input and output device-independence - ☞Checkpoint 1.1 Full keyboard access.

Describing alternative services

For content with its own interface components, such as a java applet, it is possible to use RDF or RDF-compatible languages to describe the service, and thus to search for an alternative service which has the same functions but which is accessible.

☞**Checkpoint 1.2 Activate event handlers.**

Conformance profile labels: Events

1. Allow the user to activate, through keyboard input alone, all input device event handlers that are explicitly associated with the element designated by the content focus.

2. In order to satisfy provision one of this checkpoint, the user must be able to activate as a group all event handlers of the same input device event type. For example, if there are 10 handlers associated with the onmousedown event type, the user must be able to activate the entire group of 10 through keyboard input alone, and must not be required to activate each handler separately.

☞**Checkpoint 1.3 Provide text messages.**

Image Annotation
> In Web applications annotations can be used to describe objects which are part of an interface. The development of specifications such as XUL and the CSS 3 Basic User Intereface model points to this being a trend in development.

Semantically described services
> Can be used, with the ability to describe alternatives available according to different renderings.

Guideline 2. Ensure user access to all content - ☞**Checkpoint 2.1 Render content according to specification.**

1. Render content according to format specification (e.g., for a markup language or style sheet language).

☞**Checkpoint 2.2 Provide text view.**

1. For content authored in text formats, provide a view of the text source.

☞**Checkpoint 2.3 Render conditional content.**

Conformance detail: For all content
> 1. Allow configuration to provide access to each piece of unrendered conditional content "C".
> 2. When a specification does not explain how to provide access to this content, do so as follows:

If C is a summary, title, alternative, description, or expansion of another piece of content D, provide access through at least one of the following mechanisms:
> (1a) render C in place of D;
> (2a) render C in addition to D;
> (3a) provide access to C by allowing the user to query D. In this case, the user agent must also alert the user, on

a per-element basis, to the existence of C (so that the user knows to query D); and
> (4a) allow the user to follow a link to C from the context of D.
> Otherwise, provide access to C through at least one of the following mechanisms:
> (1b) render a placeholder for C, and allow the user to view the original author-supplied content associated with

each placeholder;
> (2b) provide access to C by query (e.g., allow the user to query an element for its attributes). In this case, the

user agent must also alert the user, on a per-element basis, to the existence of C; and
> (3b) allow the user to follow a link in context to C.

☞**Checkpoint 2.4 Allow time-independent interaction.**

1. For rendered content where user input is only possible within a finite time interval controlled by the user agent, allow configuration to provide a view where user interaction is time-independent.

☞**Checkpoint 2.5 Make captions, transcripts, audio descriptions available.**

Conformance detail: For all content
> 1. Allow configuration or control to render text transcripts, collated text transcripts, captions, and audio descriptions in content at the same time as the associated audio tracks and visual tracks.

☞**Checkpoint 2.6 Respect synchronization cues.**

Conformance profile labels: Video, Audio
> 1. Respect synchronization cues (e.g., in markup) during rendering.

☞**Checkpoint 2.7 Repair missing content.**

Image Annotation
> Image annotation techniques above may mean that some data about images (such as a description, or alternative content used for the same image in a different document) is available to the user agent

☞**Checkpoint 2.8 No repair text.**

Conformance detail: For all content
> 1. Allow at least two configurations for when the user agent recognizes that conditional content required by the format specification is present but empty content:
> generate no repair text.
> generate repair as described in checkpoint 2.7.

☞**Checkpoint 2.9 Render conditional content automatically.**

Conformance detail: For all content

1. Allow configuration to render all conditional content automatically.

2. As part of satisfying provision one of this checkpoint, provide access according to specification, or where unspecified, by applying one of the techniques 1a, 2a, or 1b defined in provision two of checkpoint 2.3.

☞**Checkpoint 2.10 Don't render text in unsupported writing systems.**

1. For graphical user agents, allow configuration not to render text in unsupported scripts (i.e., writing systems) when that text would otherwise be rendered.

2. When configured per provision one of this checkpoint, indicate to the user in context that author-supplied content has not been rendered due to lack of support for a writing system.

Guideline 3. Allow configuration not to render some content that may reduce accessibility - ☞Checkpoint 3.1 Toggle background images.

Conformance profile labels: Image

1. Allow configuration not to render background image content.

☞**Checkpoint 3.2 Toggle audio, video, animated images.**

Conformance profile labels: Animation, Video, Audio

1. Allow configuration not to render audio, video, or animated image content, except on explicit user request.

☞**Checkpoint 3.3 Toggle animated or blinking text.**

Conformance profile labels: VisualText

1. Allow configuration to render animated or blinking text content as motionless, unblinking text. Blinking text is text whose visual rendering alternates between visible and invisible, at any rate of change.

☞**Checkpoint 3.4 Toggle scripts.**

1. Allow configuration not to execute any executable content (e.g., scripts and applets).

☞**Checkpoint 3.5 Toggle automatic content retrieval.**

1. Allow configuration so that the user agent only retrieves content on explicit user request.

☞**Checkpoint 3.6 Toggle images**

1. Allow configuration not to render image content.

Guideline 4. Ensure user control of rendering - ☞Checkpoint 4.1 Configure text scale.

Conformance profile labels: VisualText

1. Allow global configuration of the scale of visually rendered text content. Preserve distinctions in the size of rendered text as the user increases or decreases the scale.

2. As part of satisfying provision one of this checkpoint, provide a configuration option to override rendered text sizes specified by the author or user agent defaults.

3. As part of satisfying provision one of this checkpoint, offer a range of text sizes to the user that includes at least:

the range offered by the conventional utility available in the operating environment that allows users to choose the text size (e.g., the font size), or

if no such utility is available, the range of text sizes supported by the conventional APIs of the operating environment for drawing text.

☞**Checkpoint 4.2 Configure font family.**

Conformance profile labels: VisualText

1. Allow global configuration of the font family of all visually rendered text content.

2. As part of satisfying provision one of this checkpoint, provide a configuration option to override font families specified by the author or by user agent defaults.

3. As part of satisfying provision one of this checkpoint, offer a range of font families to the user that includes at least:

the range offered by the conventional utility available in the operating environment that allows users to choose the font family, or

if no such utility is available, the range of font families supported by the conventional APIs of the operating environment for drawing text.

☞**Checkpoint 4.3 Configure text colors.**

Conformance profile labels: VisualText
    1. Allow global configuration of the foreground and background color of all visually rendered text content.
    2. As part of satisfying provision one of this checkpoint, provide a configuration option to override foreground and background colors specified by the author or user agent defaults.
    3. As part of satisfying provision one of this checkpoint, offer a range of colors to the user that includes at least:
    the range offered by the conventional utility available in the operating environment that allows users to choose colors, or
    if no such utility is available, the range of colors supported by the conventional APIs of the operating environment for specifying colors.

☞Checkpoint 4.4 Slow multimedia.

Conformance profile labels: Animation, Audio
    1. Allow the user to slow the presentation rate of rendered audio and animation content (including video and animated images).
    2. As part of satisfying provision one of this checkpoint, for a visual track, provide at least one setting between 40% and 60% of the original speed.
    3. As part of satisfying provision one of this checkpoint, for a prerecorded audio track including audio-only presentations, provide at least one setting between 75% and 80% of the original speed.
    4. When the user agent allows the user to slow the visual track of a synchronized multimedia presentation to between 100% and 80% of its original speed, synchronize the visual and audio tracks (per checkpoint 2.6). Below 80%, the user agent is not required to render the audio track.

☞Checkpoint 4.5 Start, stop, pause, and navigate multimedia.

Media Annotation
    Annotation of multimedia such as SMIL can be used to provide important navigation marks for a media file, as per ☞XAG checkpoint 2.4.

☞Checkpoint 4.6 Do not obscure captions.

1. For graphical viewports, allow configuration so that captions synchronized with a visual track in content are not obscured by it.

☞Checkpoint 4.7 Global volume control.

Conformance detail: For both content and user agent
    1. Allow global configuration of the volume of all rendered audio, with an option to override audio volumes specified by the author or user agent defaults.
    2. As part of satisfying provision one of this checkpoint, allow the user to choose zero volume (i.e., silent).

☞Checkpoint 4.8 Independent volume control.

Conformance profile labels: Audio
    1. Allow independent control of the volumes of rendered audio content synchronized to play simultaneously.

☞Checkpoint 4.9 Configure synthesized speech rate.

Conformance profile labels: Speech
    1. Allow configuration of the synthesized speech rate, according to the full range offered by the speech synthesizer.

☞Checkpoint 4.10 Configure synthesized speech volume.

Conformance profile labels: Speech
    1. Allow control of the synthesized speech volume, independent of other sources of audio.

☞Checkpoint 4.11 Configure synthesized speech characteristics.

Conformance profile labels: Speech
    1. Allow configuration of synthesized speech characteristics according to the full range of values offered by the speech synthesizer.

☞Checkpoint 4.12 Specific synthesized speech characteristics.

Conformance profile labels: Speech
    1. Allow configuration of synthesized speech pitch. Pitch refers to the average frequency of the speaking voice.
    2. Allow configuration of synthesized speech pitch range. Pitch range specifies a variation in average frequency.
    3. Allow configuration of synthesized speech stress. Stress refers to the height of "local peaks" in the intonation

contour of the voice.
    4. Allow configuration of synthesized speech richness. Richness refers to the richness or brightness of the voice.

☞**Checkpoint 4.13 Configure synthesized speech features.**

Conformance profile labels: Speech
    1. Provide support for user-defined extensions to the synthesized speech dictionary.
    2. Provide support for spell-out: where text is spelled one character at a time, or according to language-dependent pronunciation rules.
    3. Allow at least two configurations for speaking numerals: one where numerals are spoken as individual digits, and one where full numbers are spoken.
    4. Allow at least two configurations for speaking punctuation: one where punctuation is spoken literally, and one where punctuation is rendered as natural pauses.

☞**Checkpoint 4.14 Choose style sheets.**

1. Allow the user to choose from and apply alternative author style sheets (such as linked style sheets).
    2. Allow the user to choose from and apply at least one user style sheet.
    3. Allow the user to turn off (i.e., ignore) author and user style sheets.

Guideline 5. Ensure user control of user interface behavior - ☞**Checkpoint 5.1 No automatic content focus change.**

1. Allow configuration so that if a viewport opens without explicit user request, neither its content focus nor its user interface focus automatically becomes the current focus.

☞**Checkpoint 5.2 Keep viewport on top.**

1. For graphical user interfaces, allow configuration so that the viewport with the current focus remains "on top" of all other viewports with which it overlaps.

☞**Checkpoint 5.3 Manual viewport open only.**

1. Allow configuration so that viewports only open on explicit user request.
    2. When configured per provision one of this checkpoint, instead of opening a viewport automatically, alert the user and allow the user to open it with an explicit request (e.g., by confirming a prompt or following a link generated by the user agent).
    3. Allow the user to close viewports.

☞**Checkpoint 5.4 Selection and focus in viewport.**

Conformance profile labels: Selection
    1. Ensure that when a viewport's selection or content focus changes, it is at least partially in the viewport after the change.

☞**Checkpoint 5.5 Confirm form submission.**

1. Allow configuration to prompt the user to confirm (or cancel) any form submission.

Guideline 6. Implement interoperable application programming interfaces - ☞**Checkpoint 6.1 Programmatic access to HTML/XML infoset.**

1. Provide programmatic read access to XML content by making available all of the information items defined by the W3C XML Infoset.
    2. Provide programmatic read access to HTML content by making available all of the following information items defined by the W3C XML Infoset:
    Document Information item: children, document element, base URI, charset
    Element Information items: element-type name, children, attributes, parent
    Attribute Information items: attribute-type name, normalized value, specified, attribute type, references, owner element
    Character Information items: character code, parent element
    Comment Information items: content, parent
    3. If the user can modify the state or value of a piece of HTML or XML content through the user interface (e.g., by checking a box or editing a text area), allow programmatic read access to the current state or value, and allow the same degree of write access programmatically as is available through the user interface.

☞**Checkpoint 6.2 DOM access to HTML/XML content.**

1. Provide access to the content required in checkpoint 6.1 by conforming to the following modules of the W3C Document Object Model (DOM) Level 2 Core Specification and exporting bindings for the interfaces they define:

for HTML: the Core module
for XML: the Core and XML modules
2. As part of satisfying provision one of this checkpoint:
In the Java and ECMAScript operating environments, export the normative bindings specified in the DOM Level 2 Core Specification, or
In other operating environments, the exported bindings (e.g., C++) must be publicly documented.

☞Checkpoint 6.3 Programmatic access to non-HTML/XML content.

1. For content other than HTML and XML, provide structured programmatic read access to content.
2. If the user can modify the state or value of a piece of non-HTML/XML content through the user interface (e.g., by checking a box or editing a text area), allow programmatic read access to the current state or value, and allow the same degree of write access programmatically as is available through the user interface.
3. As part of satisfying provision one of this checkpoint, implement at least one API according to this API cascade:
The API is defined by a W3C Recommendation, or the API is publicly documented and designed to enable interoperability with assistive technologies.
If no such API is available, or if available APIs do not enable the user agent to satisfy the requirements,
implement at least one publicly documented API to satisfy the requirements, and
follow operating environment conventions for the use of input and output APIs.

☞Checkpoint 6.4 Programmatic access to information about rendered content.

1. For graphical user agents, make available bounding dimensions and coordinates of rendered graphical objects. Coordinates must be relative to the point of origin in the graphical environment (e.g., with respect to the desktop), not the viewport.
2. For graphical user agents, provide access to the following information about each piece of rendered text: font family, font size, and foreground and background colors.
3. As part of satisfying provisions one and two of this checkpoint, implement at least one API according to the API cascade described in provision two of checkpoint 6.3.

☞Checkpoint 6.5 Programmatic operation of user agent user interface.

Conformance detail: For user agent features
1. Provide programmatic read access to user agent user interface controls, selection, content focus, and user interface focus.
2. If the user can modify the state or value of a user agent user interface control (e.g., by checking a box or editing a text area), allow programmatic read access to the current state or value, and allow the same degree of write access programmatically as is available through the user interface.
3. As part of satisfying provisions one and two of this checkpoint, implement at least one API according to the API cascade described in provision two of checkpoint 6.3.

☞Checkpoint 6.6 Programmatic notification of changes.

Conformance detail: For both content and user agent
1. Provide programmatic notification of changes to content, states and values of content, user agent user interface controls, selection, content focus, and user interface focus.
2. As part of satisfying provision one of this checkpoint, implement at least one API according to the API cascade of provision two of checkpoint 6.3.

☞Checkpoint 6.7 Conventional keyboard APIs.

1. Implement APIs for the keyboard as follows:
Follow operating environment conventions.
If no conventions exist, implement publicly documented APIs.

☞Checkpoint 6.8 API character encodings.

Conformance detail: For both content and user agent
1. For an API implemented to satisfy requirements of this document, support the character encodings required for that API.

☞Checkpoint 6.9 DOM access to CSS style sheets.

1. For user agents that implement Cascading Style Sheets (CSS), provide programmatic access to style sheets by conforming to the CSS module of the W3C Document Object Model (DOM) Level 2 Style Specification and exporting bindings for the interfaces it defines.
2. As part of satisfying provision one of this checkpoint:
In the Java and ECMAScript operating environments, export the normative bindings specified in the CSS

module of the DOM Level 2 Style Specification, or

In other operating environments, the exported bindings (e.g., C++) must be publicly documented.

☞**Checkpoint 6.10 Timely exchanges through APIs.**

Conformance detail: For both content and user agent

1. For APIs implemented to satisfy the requirements of this document, ensure that programmatic exchanges proceed in a timely manner.

Guideline 7. Observe operating environment conventions - ☞Checkpoint 7.1 Respect focus and selection conventions.

Conformance profile labels: Selection

1. Follow operating environment conventions that benefit accessibility when implementing the selection, content focus, and user interface focus.

☞**Checkpoint 7.2 Respect input configuration conventions.**

Conformance detail: For user agent features

Conformance profile labels: Selection

1. Ensure that default input configurations of the user agent do not interfere with operating environment accessibility conventions (e.g., for keyboard accessibility).

☞**Checkpoint 7.3 Respect operating environment conventions.**

Conformance detail: For user agent features

1. Follow operating environment conventions that benefit accessibility. In particular, follow conventions that benefit accessibility for user interface design, keyboard configuration, product installation, and documentation.

☞**Checkpoint 7.4 Provide input configuration indications.**

Conformance detail: For user agent features

1. Follow operating environment conventions to indicate the input configuration.

Guideline 8. Implement specifications that benefit accessibility - ☞Checkpoint 8.1 Implement accessibility features.

Schema Annotation

Where schemas carry annotation identifying accessibility features in the specification this can be more readily tested.

☞**Checkpoint 8.2 Conform to specifications**

Conformance detail: For all content

1. Use and conform to either

W3C Recommendations when they are available and appropriate for a task, or

non-W3C specifications that enable the creation of content that conforms at level A or better to the Web Content Accessibility Guidelines 1.0 [☞WCAG1].

Guideline 9. Provide navigation mechanisms - ☞Checkpoint 9.1 Provide content focus.

User Profiles

If user profiles are kept in RDF then it is possible to include information about the focus point in content using RDF based on the context property of the Annotea specification.

☞**Checkpoint 9.2 Provide user interface focus.**

1. Provide a user interface focus.

☞**Checkpoint 9.3 Move content focus.**

1. Allow the user to move the content focus to any enabled element in the viewport.

2. Allow configuration so that the content focus of a viewport only changes on explicit user request.

3. If the author has not specified a navigation order, allow at least forward sequential navigation, in document order, to each element in the set established by provision one of this checkpoint.

☞**Checkpoint 9.4 Restore viewport state history.**

User Profile

Maintaining state information in a user profile allows for portability of this state across sessions or even as

the user changes browser (for example changing environments from a mobile low-bandwidth browser to a more powerful browser maintained at home with a higher capacity to adapt content at the client side).

☞Checkpoint 9.5 No events on focus change

Conformance profile labels: Events
1. Allow configuration so that moving the content focus to or from an enabled element does not automatically activate any explicitly associated event handlers of any event type.

☞Checkpoint 9.6 Show event handlers

Conformance profile labels: Events
1. For the element with content focus, make available the list of input device event types for which there are event handlers explicitly associated with the element.

☞Checkpoint 9.7 Move content focus in reverse

1. Extend the functionality required in provision three of checkpoint 9.3 by allowing the same sequential navigation in reverse document order.
2. As part of satisfying provision one of this checkpoint, the user agent must not include disabled elements in the navigation order.

☞Checkpoint 9.8 Provide text search

Conformance detail: For all rendered content
1. Allow the user to search within rendered text content for a sequence of characters from the document character set.
2. Allow the user to start a forward search (in document order) from any selected or focused location in content.
3. When there is a match, do both of the following:
move the viewport so that the matched text content is at least partially within it, and
allow the user to search for the next instance of the text from the location of the match.
4. Alert the user when there is no match or after the last match in content (i.e., prior to starting the search over from the beginning of content).
5. Provide a case-insensitive search option for text in scripts (i.e., writing systems) where case is significant.

☞Checkpoint 9.9 Allow structured navigation

Site Mapping
Using an RDF vocabulary to identify important navigation points within a document as well as between them (as discussed for ☞XAG checkpoint 2.4) allows a user agent to identify the important navigation structures of arbitrary XML

☞Checkpoint 9.10 Configure important elements

Schema Annotation
Providing a mechanism for a user to identify additional annotations, or to select from those available, can provide the opportunity to declare obsolete (for a particular user) parts of a schema annotation identifying important elements for navigation, as per ☞XAG checkpoint 2.4, or make such declarations for a module described according to ☞XAG checkpoint 2.9.

Guideline 10. Orient the user - ☞Checkpoint 10.1 Associate table cells and headers.

1. For graphical user agents that render tables, for each table cell, allow the user to view associated header information.

☞Checkpoint 10.2 Highlight selection, content focus, enabled elements, visited links.

Conformance profile labels: Selection
1. Allow global configuration to highlight the following four classes of information in each viewport: the selection, content focus, enabled elements, and recently visited links.
2. For graphical user interfaces, as part of satisfying provision one of this checkpoint, allow at least one configuration where the highlight mechanisms for the four classes of information:
differ from each other, and
do not rely on rendered text foreground and background colors alone.
3. For graphical user interfaces, as part of satisfying provision one of this checkpoint, if a highlight mechanism involves text size, font family, rendered text foreground and background colors, or text decorations, offer at least the following range of values:
for text size, the range required by provision three of checkpoint 4.1.
for font family, the range required by provision three of checkpoint 4.2.

for text foreground and background colors and decorations, the range offered by the conventional utility available in the operating environment for users to choose rendered text colors or decorations (e.g., the standard font and color dialog box resources supported by the operating system). If no such utility is available, the range supported by the conventional APIs of the operating environment for specifying text colors or drawing text.

4. Highlight enabled elements according to the granularity specified in the format. For example, an HTML user agent rendering a PNG image as part of a client-side image map is only required to highlight the image as a whole, not each enabled region. An SVG user agent rendering an SVG image with embedded graphical links is required to highlight each (enabled) link that may be rendered independently according to the SVG specification.

☞**Checkpoint 10.3 Single highlight configuration.**

User Profile
> Development of user profiles for various requirements can allow for common combinations of requirements to be selected with a single setting.

☞**Checkpoint 10.4 Provide outline view.**

Schema Annotation
> The use of schema annotations to identify elements which form part of an outline view, as per ☞XAG Checkpoint 2.4, can be used to provide an outline view of a document written conformant to such a schema.

☞**Checkpoint 10.5 Provide link information**

Cataloguing metadata
> The availability of Dublin Core or similar cataloguing metadata for many resources on the Web can be used to provide this information. User agents can also generate such information for re-use.

☞**Checkpoint 10.6 Highlight current viewport.**

1. Highlight the viewport with the current focus (including any frame that takes current focus).

2. For graphical viewports, as part of satisfying provision one of this checkpoint, provide at least one highlight mechanism that does not rely on rendered text foreground and background colors alone (e.g., use a thick outline).

3. If the techniques used to satisfy provision one of this checkpoint involve rendered text size, font family, rendered text foreground and background colors, or text decorations, allow global configuration and offer same ranges of values required by provision three of checkpoint 10.2.

☞**Checkpoint 10.7 Indicate viewport position**

1. Indicate the viewport's position relative to rendered content (e.g., the proportion of an audio or video clip that has been played, or the proportion of a Web page that has been viewed).

Guideline 11. Allow configuration and customization - ☞**Checkpoint 11.1 Current user input configuration.**

Conformance detail: For user agent features
1. Provide information to the user about current user preferences for input configurations.

☞**Checkpoint 11.2 Current author input configuration.**

Conformance detail: For all content
1. Provide a centralized view of the current author-specified input configuration.

☞**Checkpoint 11.3 Allow override of bindings.**

Conformance detail: For user agent features
1. Allow the user to override any binding that is part of the user agent default input configuration.

☞**Checkpoint 11.4 Single-key access.**

Conformance detail: For user agent features
1. Allow the user to override any binding in the user agent default keyboard configuration with a binding to either a key plus modifier keys or to a single key.

2. For each functionality in the set required by checkpoint 11.5, allow the user to configure a single-key binding. A single-key binding is one where a single key press performs the task, with zero modifier keys.

☞**Checkpoint 11.5 Default input configuration.**

Conformance detail: For user agent features
1. Ensure that the user agent default input configuration includes bindings for the following functionalities required by other checkpoints in this document:

move content focus to the next enabled element in document order, and move content focus to the previous enabled element in document order (checkpoints 9.3 and 9.7);

activate the link designated by the content focus (checkpoints 1.1 and 9.1);

search for text, search again for same text (checkpoint 9.8);

increase the scale of rendered text, and decrease the scale of rendered text (checkpoint 4.1);

increase global volume, and decrease global volume (checkpoint 4.7); and

stop, pause, resume, and navigate efficiently selected audio and animations, including video and animated images (checkpoint 4.5).

2. If the user agent supports the following functionalities, the default input configuration must also include bindings for them:

next history state (forward), and previous history state (back);

enter a URI for a new resource;

add a URI to favorites (i.e., bookmarked resources);

view favorites;

reload a resource;

interrupt a request to load or reload a resource;

for graphical viewports: navigate forward and backward through rendered content by approximately the height of the viewport; and

for user agents that render content in lines of (at least) text: move the point of regard to the next and previous line.

☞**Checkpoint 11.6 User profiles.**

EARL / CC/PP
> The use of RDF as a profiling mechanism allows for profiles to be described in the CC/PP vocabulary and made avialable as conforming to certain requirements, either expressed as EARL conformance to CC/PP statements, or for IMS user profiles. Other profiles built on these standardised bases can then be shared readily and interoperably.

☞**Checkpoint 11.7 Tool bar configuration.**

Conformance detail: For user agent features

1. For graphical user agent user interfaces with tool bars, allow the user to configure the position of user agent user interface controls on those tool bars.

2. Offer a predefined set of controls that may be added to or removed from tool bars.

3. Allow the user to restore the default tool bar configuration.

Guideline 12. Provide accessible user agent documentation and help - ☞**Checkpoint 12.1 Provide accessible documentation.**

Conformance detail: For user agent features

1. Ensure that at least one version of the user agent documentation conforms to at least level Double-A of the Web Content Accessibility Guidelines 1.0 [☞WCAG10].

☞**Checkpoint 12.2 Provide documentation of accessibility features.**

Semantically described Tools
> providing machine-processable documentation of accessibility features in the tool can help in generating the required information for other checkpoint 12.5.

Schema Annotation
> Schemas whose accessibility features are identified permit a user agent to more clearly identify how it implements accessibility features, combining this information with information about its own accessibility to conform to this checkpoint.

☞**Checkpoint 12.3 Provide documentation of default bindings.**

Conformance detail: For user agent features

1. Provide documentation of the default user agent input configuration (e.g., the default keyboard bindings).

☞**Checkpoint 12.4 Provide documentation of changes between versions**

Describing Tools
> Providing machine-processable descriptions of features available in a tool can automate some of this process.

☞**Checkpoint 12.5 Provide dedicated accessibility section.**

Describing Tools
> Providing machine-processable descriptions of features available in a tool can automate some of this process.

6. XAG

Guideline 1: Ensure that authors can associate multiple media objects as alternatives - ☞Checkpoint 1.1 Provide a mechanism to explicitly associate alternatives for content or content fragments.

☞Checkpoint 1.2 Define flexible associations, where a given kind of relationship can link to or from objects of varying types without constraint.

Guideline 2. Create semantically-rich languages - ☞Checkpoint 2.1 Ensure all semantics are captured in markup in a repurposable form.

☞Checkpoint 2.2 Separate presentation properties using stylesheet technology/styling mechanisms.

☞Checkpoint 2.3 Use the standard XML linking and pointing mechanisms (XLink and XPointer).

☞Checkpoint 2.4 Define element types that allow classification and grouping (header, section, list, etc).

☞Checkpoint 2.5 Provide for a full containment model with chunks of reasonable size.

☞Checkpoint 2.6 Define element types that identify important text content.

☞Checkpoint 2.7 Provide a mechanism for identifying summary / abstract / title.

☞Checkpoint 2.8 Don't overload element and attribute names.

☞Checkpoint 2.9 Reuse existing accessible modules, as originally specified / intended.

Schema Annotation
Annotations for schema modules can describe their purpose, including EARL documentation of their conformance to XAG, which would enable searching for an appropriate module and judging the relative accessibility. (Likewise, documentation of their implementations can be useful).

☞Checkpoint 2.10 Allow association of metadata with distinct elements and groups of elements.

RDF data
Providing a metadata element that allows for the direct inclusion of RDF, or providing an explicit linkage to machine-readable data combined with the ability to address distinct elements by URI, achieves this.

☞Checkpoint 2.11 Specific checkpoint for Final-form applications.

Schema Annotation
Annotating schemas to point out that they are intended only as a final rendering format, and not as an authoring format, and pointing to authoring formats that can be used to generate content for the format may both be useful techniques for this checkpoint.

Guideline 3. Design an accessible user interface - ☞Checkpoint 3.1 Provide default style sheets for multiple output modalities.

☞Checkpoint 3.2 Define navigable structures that allow discrete, sequential, structured, and search navigation functionalities.

☞Checkpoint 3.3 Use CSS or XSLT to describe a basic outline view

☞Checkpoint 3.4 Use a device-independent interaction and events model / module.

☞Checkpoint 3.5 Allow for user control of interaction timing - rate of change, external events triggering document changes, etc.

Guideline 4 Document and export semantics - ☞Checkpoint 4.1 Provide explicit human readable definitions for markup semantics.

Schema Annotation
The use of an element within the schema language that is defined as a natural language description can be processed by referring to RDF descriptions of the schema language itself
Annotea
Annotea annotations can be used to provide human-readable information about a part of an XML document, including one which describes the elements and attributes which can be used in an XML language.

☞**Checkpoint 4.2 Ensure that at least one version of the XML application's documentation conforms to at least level Double-A of the Web Content Accessibility Guidelines 1.0 [WCAG1].**

☞**Checkpoint 4.3 Provide a machine-understandable means/mechanism to get from a document instance to the schema.**

Versioning
> Where a document format does not differentiate versions, different requirements may apply in different versions of a document's definition. External descriptions can be use to identify which version of a format a particular document instance uses.

☞**Checkpoint 4.4 Use a schema language that can support explicit human-readable documentation or annotation of semantics.**

Schema annotation
> Languages such as XML Schema allow explicit documentation of each element.

Annotea
> It is possible to use Annotea to annotate any XML langauge, including one which describes a schema for a markup language, as well as any element name which can be expressed as a URI with RDF.

☞**Checkpoint 4.5 Provide semantic relationships to other schema where appropriate and possible.**

RDF Vocabularies and OWL
> providing an RDF version of two schemata allows relations between them to be declared in terms of properties defined by the RDF Vocabulary specification (for subClasses, subTypes, and so on), OWL (for further definition), or in specific RDF vocabularies developed from the purpose.

☞**Checkpoint 4.6 Document all features of the XML application that benefit accessibility.**

Schema annotation
> An RDF vocabulary could be used to define types of accessibility property. This can be used to automate the process of authoring tools presenting accessibility functionalities, as required by ATAG checkpoint 5.2

☞**Checkpoint 4.7 Include accessibility requirements in conformance requirements.**

☞**Checkpoint 4.8 Document techniques for WCAG, ATAG, and UAAG with respect to the XML application.**

Schema Annotation
> In many cases existing techniques will work for new langauges. Annotating the schema to point to existing techniques which are applicable may considerably reduce the work required to conform to this checkpoint.

☞**Checkpoint 4.9 Do not assume that element or attribute names provide any information about element semantics.**

☞**Checkpoint 4.10 Document navigable structures. Describe how discrete, sequential, structured, and search navigation mechanisms should work**

Site Mapping
> A vocabulary used for site mapping could be re-used to describe navigation within documents as a general scheme.

7 Themes

Several general areas of work seem to offer promise in a numer of areas. Additionally, there are some areas of work which don't cover a particular checkpoint or guideline, but where the Semantic Web seems to offer an overall improvement in the accessibility of a type of content:

CC/PP
> CC/PP - the Composite Conformance / Preferences Profile is a W3C specification describing, in RDF, a framework for negoatiating different version of a resource according to the needs or preferences of a user. Such needs or preferences may be determined by disability, or equally by capabilities of the platform the user happens to have available at a given moment.

Describing Tools
> See ☞Semantically described Services and Tools

EARL
> EARL - The Evaluation And Reporting Language is a W3C working draft for an RDF language that can describe the results of conformance evaluations. It was developed by the W3C's Evaluation and Repair Tools

Working Group, a part of the WAI Technical Activity.

Mapping and geographical information

Current models for presenting geographical information on the Web tend to revolve around images, with the notable exception of text-based driving directions. The use of semantic Web technology for manipulating geographic information could offer more accessible presentations of such information on scales from the global to the location of things in a room. Work in areas such as location-specific information on mobile telephones, 'webtiles', and wayfinding services, show some of the interactions between information that is or could be in the Semantic Web and people's everday lives in the physical world.

Multimedia annotations

One of the large areas of accessibility concern is multimedia, because people with disabilities may not be able to use all the media concerned, or may not be able to work simultaneously with the multiple media. There are many techniques which can be used to provide supplementary information about media objects, that can be used in authoring support, or in user agents.

Schema Annotation

Annotating schema definitions directly can be used by authoring tools in many ways to support accessible use of an XML vocabulary. It can also provide clues for evaluation or presentation in user agents, with the use of machine-processable annotations.

Semantically described services and tools

Providing machine-processable descriptions of tool functionalities (including Web services) can allow users to find or incorporate into the framework of their work environment an appropriate alternative to a particular functionality without having to change their toolset completely. For users who require assistive technologies it can also simplify the process of integration into their work style, or even managing compatibility.

Text annotation

The techniques available for multimedia annotation can also be applied to addressable text on the Web, noting simpler, more interactive, or otherwise different presentations of it.

User profiles

There are a number of possible approaches to defining a user's preferences and requirements which can be used to determine whether a (possibly) more suitable version of content can be offered than might otherwise be the case. These techniaques can also be applied to configuration requirements, a major part of the User Agent Accessibility Guidelines, and which are also required by the Authoring Tool Accessbility Guidelines.

8 Implementations

Some of the appraoches described above have existing implentation as either a product or a proof-of-concept. Others are in development at the time of writing. Although any survey of implementations is unlikely to be exhaustive, the following are some that may be of interest. None of these implmentations are endorsed as being a usful, accurate, or fit implementation of any particular approach, and this list is known to be arbitrary and primarily focused on European-developed Open Source technology. However information about semantic web tools for accessibility can be sent to public-esw or wai-er-ig with a request that they be incorporated into this list during the life of this document.

The MUTAT evaluation tool [☞MUTAT] was adapted to use Annotea as a storage and retrieval system, working with the experimental EARL server provided by W3C. The Axform tool suite [☞AXFORM] has been developed as a replacement, using Xforms technology to replace server scripting for greater portability.

Accessibility testing tools including Access Valet, AccVerify™, Axform, WAINu, produce reports in EARL of the accessibility of Web Content. A list of many accessibility testing tools [☞ERTools] is maintained by WAI.

RDFPic [☞RDFPic] demonstrates image annotations included in an image file. (Further tools for manipulating this information have been produced by Norm Walsh [☞JPEGRDF]).

Jim Ley's image annotation tools [☞JImAnno] demonstrate the use of annotations stored seperately from the image file.

See ☞CC/PP implementation information maintained by W3C's Device Independence Activity.

9 Future work

The next stage in this workpackage is to work with WAI to determine which of the suggestions in this document are considered valuable techniques, providing further details or implementation examples as necessary, and incorporating them into the relevant techniques Notes where appropriate.

Some of this work has now been done, including presenting some RDF techniques to the Web Content Accessiblilty Guidelines working group meeting in Venice, July 2003, and presenting (by email) some tools developed for EARL as part of this project. Further work with WAI, the Dublin Core Metadata Initiative's Accessibility Interest Group, and the EuroAccessibility Consortium in particular, is anticipated.

References

[ANNOCOL]

The ☞Annotea protocol is documented at http://www.w3.org/2001/Annotea/User/Protocol

[ANNOTEA-R]

The ☞Schema for threaded annotations, developed to allow for threaded responses, is available at http://www.w3.org/2001/03/thread

[ANNOTOOLS]

A small ☞library of tools developed for use with the Annotea protocol, and as example code for people wanting to develop their own tools. These tools are described at http://www.w3.org/2001/sw/Europe/200209/annodemo/readme.html

[ATAG1]

"☞Authoring Tool Accessibility Guidelines 1.0" ed. Jacobs, McCathieNevile, Richards, Treviranus is a W3C Recommendation published on 2 February 2000. It is available at http://www.w3.org/TR/ATAG10

[ATAG2]

"☞Authoring Tool Accessibility Guidelines 2.0" ed. McCathieNevile, May, Richards, Treviranus is a W3C Working Draft. It is available at http://www.w3.org/TR/ATAG20

[ANNOT-IMPLE]

A list of ☞known Annotea Implementations is maintained by W3C's Annotea project at http://www.w3.org/2001/Annotea/#Comp

[AXFORM]

Some ☞Xforms-based tools for creating EARL evaluations, designed as a replacement for the MUTAT tool. Available at http://www.w3.org/2001/sw/Europe/200305/axforms/

[EARL]

☞EARL (The Evaluation and Reporting Language) is a specification in development by the W3C's Evaluation and Repair Tools group. It is an RDF vocabulary for expressing conformance to arbitrary requirements. The Latest published draft is available at http://www.w3.org/TR/EARL10

[ERTools]

WAI maintains a list of ☞tools for accessibility evaluation and repair, at http://www.w3.org/WAI/ER/existingtools

[JImAnno]

Some tools for annotating images through SVG. Available from produced by Jim Ley.

[MUTAT]

The Open Source ☞MUTAT tool is designed to provide an interview-style interface for producing conformance reports in the EARL [☞EARL] format. It has an extension which allows the reports to be posted as annotations to an annotea server. An online version of the Tool is available at http://www.w3.org/QA/Tools/MUTAT/

[JPEGRDF]

☞Norm Walsh's tools for working with RDF in JPEG files. Available from http://nwalsh.com/java/jpegrdf/

[RDFPic]

A ☞tool for adding RDF to a JPEG image file developed at W3C. Available from http://jigsaw.w3.org/rdfpic

[UAAG]

The "☞User Agent Accessibility Guidelines 1.0", ed. Gunderson, Hansen, Jacobs, is a W3C Recommendation published on 17 December 2002, available at http://www.w3.org/TR/UAAG10

[WCAG1]

The "☞Web Content Accessibility Guidelines 1.0", ed. Chisholm, Jacobs, Vanderheiden is a W3C Recommendation published on 5 May 1999, available at http://www.w3.org/TR/WCAG10

[WCAG2]

The "☞Web Content Accessibility Guidelines 2.0", ed. Caldwell, Chisholm, White, Vanderheiden is a W3C Working Draft. The version used in this report was published on 22 August 2002, and the latest version is available at http://www.w3.org/TR/WCAG20

[WWW-ANNOTATION]

The www-annotation@w3.org mailing list is a public discussion forum for annotation systems, including Annotea. ☞Its archives (including instructions for subscribing) are available at http://lists.w3.org/Archives/Public/www-annotation/

[XAG]

"☞XML Accessibility Guidelines" ed. Dardailler, McCathieNevile, Palmer is a W3C Working Draft. The The version used in this report was published on 3 October 2002, and the latest version is available at http://www.w3.org/TR/xag

[LICENSE]

The ☞W3C Software Copyright license is a BSD-style license allowing the free use of software in open-source or proprietary products with appropriate acknowledgement. The full license is available at http://www.w3.org/Consortium/Legal/copyright-software