TOWARDS BEHAVIOUR-DRIVEN TRANSCODING OF WEB CONTENT THROUGH AN ANALYSIS OF USER COPING STRATEGIES

A thesis submitted to the University of Manchester
for the degree of Doctor of Philosophy
in the Faculty of Engineering and Physical Sciences

2009

By
Darren Richard Lunn
School of Computer Science

Contents

List of Tables

List of Figures

Abstract

People with visual impairments are hindered when accessing information on the World Wide Web (Web) as content is not designed with their needs in mind. Visually impaired users can access the Web through screen readers that use the underlying structure of a page to create a sequential, audio rendering of the content. However, most designers are mainly concerned with how content is presented, rather than its structure and meaning. Consequently, implicit information available through the visual rendering of the content is lost to screen readers and therefore users. To address this problem, tools that transcode Web content into a format more suitable for screen readers have been developed. While these tools have assisted users in accessing Web content, limitations have been identified. Firstly, the approaches taken have either been scalable but inaccurate, or accurate but unscalable. Secondly, the transformations have tended to focus on adapting content to meet the needs of the device rather than the user.

This thesis presents work that addresses both these limitations. SADIe, a content transcoder, was developed that is both accurate and scalable. This is achieved by annotating the Cascading Style Sheet of a Website. The annotations provide accurate transcoding as they identify key elements of the page when applying the transformations. As most Websites typically have one set of style sheets that all pages refer to, the annotations propagate to every page providing scalability. Technical evaluations of SADIe established that it was capable of consistently transcoding a diverse range of Websites. Unlike previous tools, the transformations used were based upon an understanding of behavioural strategies users employ when accessing Web content. A study of eleven users identified forty-eight strategies categorised into six abstract patterns. Transformations based on four of these patterns were incorporated into SADIe. Qualitative and quantitative user studies of the behaviour-driven transcoding demonstrated that the approach can assist users in accessing Web content beyond that of previous solutions.

Declaration

No portion of the work referred to in this thesis has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.

Copyright

Acknowledgements

More than most I would like to thank my supervisors Simon Harper and Sean Bechhofer for all their support, advice and encouragement. Throughout the entire duration of this research they have engaged me in interesting and enthusiastic discussion that has often been a source of great inspiration. I am also grateful to Eleni Michailidou for her invaluable advice on devising and analysing user studies and Yeliz Yesilada for her feedback and constructive comments that have contributed towards the work presented in this thesis.

I would like to express gratitude to the Human Centred Web Lab and the wider Information Management Group for creating an enjoyable and productive working environment. The regular debates over coffee were often stimulating and a joy to be involved with.

In addition, I appreciate the assistance of Jim Hough, David Devoy and the staff and students of Henshaws' Society for Blind People who allowed me into their classroom and took time from their teaching schedules to help me gather research data. Also Jenny Craven, who kindly provided a significant amount of raw data with which to validate my own results.

Finally, my entire family, particularly my parents, who have been supportive throughout this work for which I am indebted.

Chapter 1

Introduction

Originally the World Wide Web (Web) existed as a text-based environment that connected documents through simple hyperlinks [Berners-Lee 1999]. The language used to compose pages — the HyperText Markup Language (HTML) — was accessible to assistive technologies as it focussed upon the structure of the content of the document. The original specification of HTML did not focus on presentation and instead defined structural elements such as headings, links, paragraphs and lists [Berners-Lee 1992]. Since then, the Web has evolved into a complex, graphical, multimedia environment that plays an important role in society [Ivory & Megraw 2005]. The Web has become a vast resource of information and ideas and is becoming an essential element of commercial activity, communication and social interaction.

People with disabilities, in particular visual impairments, can make use of tools to access the Web. These tools, known as screen readers, access the underlying structure of the HTML to create a sequential, audio rendering of the document [James 1998]. However, visually impaired users can be hindered in their access to information on the Web when the content is not designed with their needs in mind [DRC 2004]. Most Web designers are mainly concerned with how content is presented on screen, rather than its structure and meaning. Consequently, implicit information that is available through the visual rendering of the page is lost to the screen reader, and therefore the user.

As an example, consider a navigation menu common on many Websites. Figure 1.1 on page 15 shows a page from the Human Centred Web Lab Website[1].

[1]URL: `http://hcw.cs.manchester.ac.uk/`. Last accessed: 12th August 2008

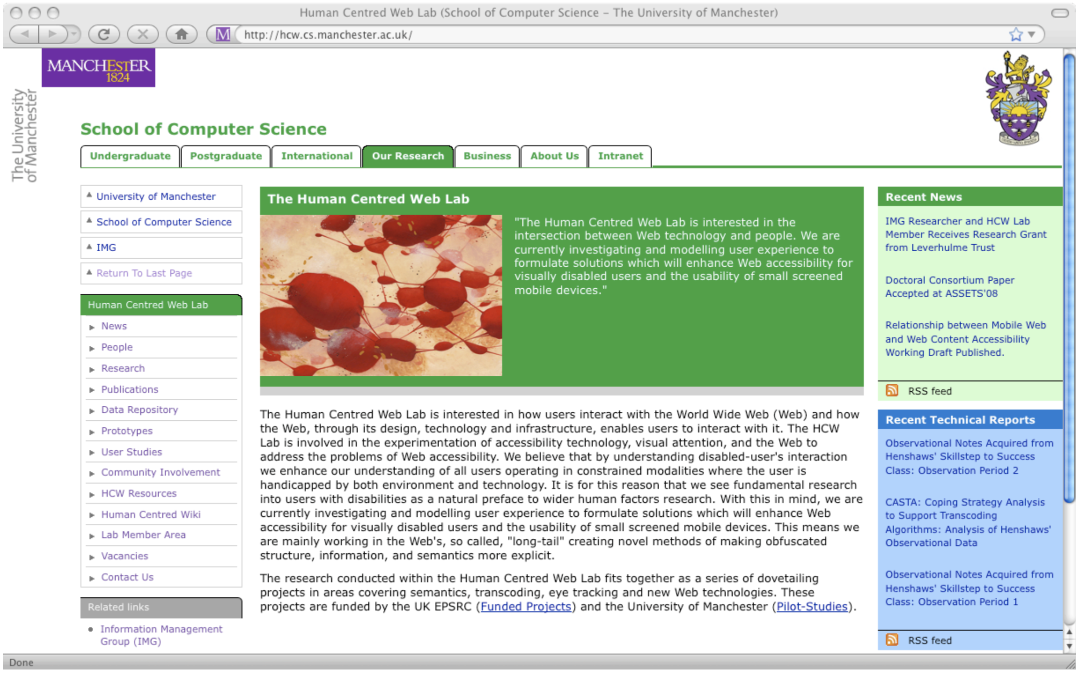


Figure 1.1: The Human Centred Web Lab Website. The navigation menu, located on the left-hand-side of the page, is identifiable due to presentation and layout.

The navigation menu is located towards the left-hand-side of the screen and, contained within its own distinct chunk, separated from the rest of the content. A sighted user is aware which element on screen is the navigation menu due to the way it is rendered. There is nothing explicitly stating that the element on the Web page is a menu. The knowledge that it is a menu is implicit from the visual presentation, which is only available to those who can see it, as opposed to those people who use a screen reader to read aloud the page content.

To overcome the difficulties associated with visual-centric Web pages, tools exist that transform Web pages into a format more suited to the sequential audio rendering of screen readers [Takagi & Asakawa 2000]. These tools, known as transcoders, have improved access to the Web by adapting the content through structure-driven adaptations. These adaptations are based on an understanding of the structure of the content and how this structure can be mapped between different devices.

Transcoders seek to provide adaptations to Web content that are both scalable and accurate. In this context, scalable can be defined as the ability to adapt a large number of Web pages with a minimal amount of human intervention. Accuracy can be defined as the ability to consistently apply transformations to a set of Web pages in a systematic and predictable manner. Previous approaches to transcoding have broadly fallen into two categories. The first, known as heuristic transcoders, use techniques that are highly scalable so that a large number of Web pages can be adapted easily. However, in order to accommodate the large number of Websites, accuracy suffers leading to transformations that do not always produce the desired results [Brown & Robinson 2001; Gupta *et al.* 2005]. Alternatively, semantic transcoders have been proposed that have high levels of transcoding accuracy. These use techniques that typically require Web pages to be annotated before any transformation can take place. Due to the workload placed on the human when adding annotations to Web pages, these systems have lacked the scalability to be applied to a large number of pages [Asakawa & Takagi 2000; Huang & Sundaresan 2000].

Structural-Semantics for Accessibility and Device Independence (SADIe) is an approach to transcoding that combines the benefits of heuristic and semantic transcoding first proposed by Harper & Bechhofer [2004] but never developed or evaluated. The principle idea behind the approach is that the rendering of a Web page element is closely associated with its role. Rendering information can be defined within the Cascading Style Sheet (CSS) and associated with the HTML via tag attributes such as `class` or `id`. A Website typically has a homogenous cascade of styles containing all the style definitions. Therefore, rather than annotate every page, the SADIe approach is to annotate the CSS classes themselves. This reduces the annotation overhead as, rather than annotate every page within the Website, annotation occurs in one location at the CSS level. As the pages of the Website are associated with site-wide style definitions of the CSS elements, they not only inherit the rendering information but also the annotations that have been associated with the CSS. The CSS annotations allow semantic transcoding to take place as the role of each element is explicitly stated, improving the accuracy of the adaptations. In addition, scalability is achieved as defining the roles of CSS elements allows every page within the Website to be transcoded due to the CSS containing site-wide style definitions that all pages use.

It should be noted that the transcoding techniques discussed previously are typically aimed at Web pages that contain static content where information can be accessed in a linear fashion. However there is a trend towards interactive Websites, known as Web 2.0, where users actively contribute to the content [Millard & Ross 2006]. Web 2.0 Websites make use of Flash, JavaScript, and Asynchronous JavaScript and XML (AJAX) technologies to create dynamically updating regions within the page that break the linear access assumption [Thiessen & Chen 2007]. While we acknowledge that the Web is becoming more interactive, and therefore cannot be transcoded using techniques such as SADIe, we assert that a majority of the Web still consists of static-based documents, and the work presented in this thesis targets these pages.

1.1 Research Questions and Contribution

As with previous transcoders, the original SADIe proposal involved a series of transcoding solutions that were based on established structure-driven transcoding techniques. To further enhance Web accessibility, progression from structure-driven transcoding towards behaviour-driven levels of content adaptation is required. This can be achieved through an understanding of user behavioural strategies and a specialised form of behavioural strategy known as coping strategies.

Behavioural strategies are techniques users employ to achieve their goals as they browse the Web. Figure 1.2 on page 18 shows the search results of the Google Search Engine[2]. To view all the results, novice users will use the arrow buttons, indicated as ①, to slowly move the page up and down line by line. More advanced users will click the mouse button on the blue scroll bar, indicated as ②, and rapidly drag the page up and down to scan the results. Both behavioural strategies are encoded within the Web browser and allow users to achieve the same goal of seeing all the search results, albeit suited to two different user groups.

Coping strategies are special behavioural strategies that are developed when behavioural strategies are no longer sufficient or available to access content effectively. At this point, users develop alternative techniques to overcome the difficulties of accessing content within the page. As described previously, sighted users can make use of the scroll bars to access the search results. For visually impaired users, there is no equivalent mechanism encoded into the page. This results in most users having to listen to all the content of the page, including the banner and advertisements, before they reach the main content; which in this

[2]URL: `http://www.google.co.uk/`. Last accessed: 12$^{\text{th}}$ August 2008

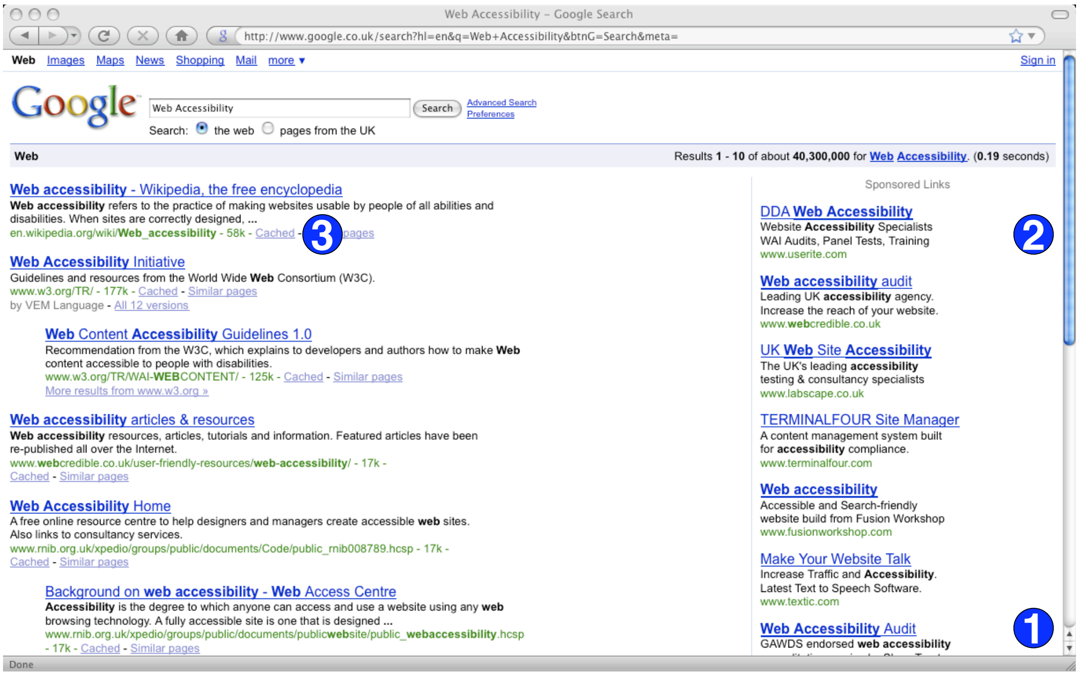


Figure 1.2: The Google search results Web page. Visually impaired users can find the word "cached", indicated as ③, to quickly reach the main content.

example is the search results. However, it has been observed that some visually impaired users have developed a technique whereby finding the word "cached", indicated as ③ in Figure 1.2, quickly moves the focus of the screen reader to the first search result [Yesilada *et al.* 2007b]. This allows users to avoid the banner and advertisements and access information contained within the page in a more satisfactory manner.

While this is a well known example, there has been little research conducted into coping strategies from a Web accessibility or computer science viewpoint. The motivation to investigate coping strategies and understand their effectiveness at enhancing Web accessibility gave rise to a series of research questions. The investigation into these questions is discussed in the remainder of this thesis and represents the contribution made by this work.

1. **Can SADIe Consistently Apply Transcoding to a Diverse Range of Websites Systematically?**
   SADIe is an approach that has been proposed but not thoroughly developed or evaluated. Implementing a prototype of SADIe as a transcoding proxy will allow us to investigate the feasibility of the SADIe approach from both

a technical and user perspective. If SADIe is demonstrated to be capable of consistently and predictably applying transcoding to a diverse range of Websites in a systematic fashion, it can be used as a platform upon which to experiment and investigate behaviour-driven transcoding. The advantages of this would be that a consistent comparison of results could be made. Implementing and evaluating the prototype would provide a base level of SADIe's capabilities. Any subsequent user studies conducted after changes to the transcoding engine had been made would then be attributable to those modifications as opposed to other factors.

2. **Can Coping Strategies Be Identified as Visually Impaired Users Access Web Content?**
   There is a point at which a Web page is difficult to access with behavioural strategies. At this point users rely upon coping strategies in order to effectively access the content. Understanding how users cope with Web content will allow us to investigate coping strategies and look into methods of modelling those strategies in a consistent manner.

3. **Can Identified Coping Strategies Be Used to Define Transcoding Solutions?**
   Should we show that it is possible to identify and classify coping strategies, our overall objective is to use the identified strategies as a means to enhance transcoding applications. The test-bed for incorporating any algorithms that are derived from identified coping strategies will be the SADIe platform. Should SADIe prove not to be a viable transcoding solution, then alternative transcoding platforms, such as HomePage Reader [Asakawa & Itoh 1998], could be used for our experimentation although we would lose some control over the transcoding functionality and how it is applied. Answering this question will support our approach from a technical perspective and demonstrate the feasibility of identifying transcoding solutions from user behaviour.

4. **Do Deeper Levels of Transcoding Improve Access to Web Content?**
   Evaluating user performance with the original version of a Web page, the structure-driven transcoded version and the behaviour-driven transcoded version will allow comparisons to be made as to which approach is most

effective. Our overall objective is to demonstrate that the use of behaviour-driven transcoding is a viable method of enhancing Web accessibility beyond that provided by structure-driven transcoding.

1.2 Thesis Structure

The investigations conducted to answer the questions outlined in Section 1.1 are described in the remainder of this thesis, which is structured in the following manner:

Chapter 2: Background and Related Work.
A review of the literature shows that while accessibility guidelines exist, they are not sufficient to provide Web pages that are accessible. Research into transcoding, whereby Web content is adapted into a format that is suitable for assistive technologies, has been conducted to further improve Web accessibility. While a number of approaches have been taken to transcode Websites, broadly speaking the proposed solutions are either highly scalable to the detriment of accuracy, or highly accurate but lack scalability. Furthermore, the adaptations that are provided are structure-driven and based upon anecdotal evidence. While coping and coping strategies have been observed during these studies, there has been little investigation into understanding the phenomena and of trying to use coping strategies as a means to enhance transcoding functionality and Web accessibility.

Chapter 3: The SADIe Experimental Prototype.
Supporting Research Question 1, an experimental prototype of the SADIe transcoder was created. During development of the prototype two approaches to annotating the role of the elements contained within the Web page were investigated. The first involved a two-part ontology that contained elements found within the CSS in the lower ontology and abstract role concepts from an upper ontology. During usage, weaknesses within the annotation model became apparent therefore a second approach was investigated. This approach involved extending the Cascading Style Sheet properties. This resulted in increased flexibility of the annotation approach and reduced the overhead of annotating Websites. A publication that discusses the rational and the contributions made to the paper are discussed

in item 6 in Section 1.3. Technical evaluations of the method using established, structure-driven transcoding techniques demonstrated that the approach was a transcoding solution which provided accurate and predictable transcoding that was scalable to a large number of Web pages. Publications that resulted from the prototype evaluations, and the contributions made to each paper, are discussed in items 1, 2, 3, 5 and 8 in Section 1.3. This provided a solid basis for using SADIe as the transcoding platform upon which to investigate both structure-driven and behaviour-driven transcoding from a user perspective.

Chapter 4: Behaviour-Driven Transcoding.

Supporting Research Question 2, studies identified the coping strategies that users employed when access to Web content became too taxing, resulting in a framework consisting of seven dimensions. This framework was validated using two additional data sources and demonstrated to be flexible and accurate enough to be applicable to multiple data sources. In total the three data sources identified forty-eight coping strategies that users employed.

Supporting Research Question 3, an analysis of the coping strategies identified from the coping strategies framework revealed six abstract patterns of coping. This allowed for the development of behaviour-driven transcoding functionality. The transcoding that was identified transformed static Web content into interactive content by allowing users to navigate between primary elements of the page through a consistent set of key presses. This demonstrated that identified coping strategies can be used to define transcoding solutions. These solutions were then incorporated into the SADIe experimental transcoding proxy to form the basis of the behaviour-driven transcoding user evaluation.

Chapter 5: Transcoding Evaluation.

Supporting Research Question 4, user studies comparing time-to-task completion for a number of Web pages were conducted using the original versions of Web pages, structure-driven transcoded versions and behaviour-driven transcoded versions. Qualitative and quantitative results of the evaluation demonstrated that SADIe's structure-driven transcoding improved access to content for visually impaired users when compared to completion times for the the original pages. Publications that resulted from the user evaluations, and the contributions made to each paper, are discussed in items 4 and 7 in Section 1.3. Further qualitative and quantitative results also revealed that behaviour-driven transcoding improved access to content for visually impaired users when compared to structure-driven transcoding. These results demonstrated that SADIe's behaviour-driven transcoding improved access to content for visually impaired users beyond the structure-driven transcoding.

Chapter 6: Summary and Conclusions.
This research has shown that SADIe can consistently and predictably apply transcoding to a diverse range of Websites in a systematic fashion and is therefore a suitable platform for improving access to the Web for visually impaired users. Further investigations have identified a number of behaviour-driven transcoding solutions that have also been shown to enhance access to the Web for visually impaired users. We have therefore provided evidence to support the four research questions and demonstrated that behaviour-driven transcoding, achieved through an understanding of user coping strategies, can further enhance Web accessibility.

1.3 Publications

The results of the research conducted to answer the questions described in Section 1.1, and outlined in Section 1.2, have led to the following peer reviewed publications, shown in chronological order.

1. **Taming the Inaccessible Web** *Simon Harper, Sean Bechhofer and Darren Lunn*. In SIGDOC '06: Proceedings of the 24$^{\text{th}}$ ACM International Conference on the Design of Communication, Pages 64–69, October 2006

   This paper discussed how SADIe can assist in finding semantic information within documents that can aid visually impaired users without incurring significant design overhead. The contribution of Lunn was to implement a working prototype to demonstrate that the ontologies describing the CSS roles could be used to apply transcoding to the page.

2. **Transcoding Based On CSS** *Simon Harper, Sean Bechhofer and Darren Lunn*. In ASSETS '06: Proceedings of the 8^{th} International ACM Conference on Computers and Accessibility, Pages 259–260, October 2006

   This paper demonstrated how the transcoding of the SADIe approach could be of benefit to visually impaired users. The contribution of Lunn was to implement a working prototype to demonstrate the transcoding available to visually impaired users.

3. **SADIe: Semantic Annotation for Accessibility** *Sean Bechhofer, Simon Harper and Darren Lunn*. In ISWC '06: Proceedings of the 5^{th} International Semantic Web Conference, Pages 101–115, November 2006

   This paper provided a technical evaluation of SADIe to demonstrate the viability of the method. The contribution of Lunn was to perform the evaluation on a selection of forty Websites and perform the statistical analysis.

4. **SADIe: Exposing Implicit Information to Improve Accessibility** *Darren Lunn*. In ASSETS '07: Proceedings of the 9^{th} International ACM Conference on Computers and Accessibility, Pages 263–264, October 2007

   This paper presented the preliminary results of a user evaluation with a single participant. The results provided evidence that the structure-driven transcoding functionality provided by SADIe was of benefit to visually impaired users. This paper was submitted to the ACM Student Research Competition at ASSETS 2007 finishing second and receiving $500 prize money.

5. **The SADIe Transcoding Platform** *Darren Lunn, Sean Bechhofer and Simon Harper*. In W4A '08: Proceedings of the 2008 International Cross-Disciplinary Conference on Web Accessibility, Pages 128–129, May 2008

   This paper described the SADIe system architecture of the transcoding proxy and how this worked in conjunction with the ontologies containing the CSS annotations. The contribution of Lunn was to construct the transcoding engine and the architecture that pieced the components together. This paper was submitted to the Web Accessibility Challenge at W4A 2008 winning the Judges Award.

6. **Experiences of Exposing Semantics to Drive Transcoding** *Darren Lunn, Sean Bechhofer and Simon Harper.* In ONTORACT '08: Proceedings of the 1st International IEEE Workshop on Ontologies in Interactive Systems, Pages 9–17, September 2008

   During the experimentation with the transcoding, it became apparent that there were issues with the ontology approach that had been taken. This paper describes those problems and presents an alternative approach using an extension to CSS properties. The contribution of Lunn was to investigate the alternative approach and show that not only could the original transcoding be achieved but that it provided additional flexibility in the transcoding provided.

7. **A User Evaluation of the SADIe Transcoder** *Darren Lunn, Sean Bechhofer and Simon Harper.* In ASSETS '08: Proceedings of the 10th International ACM Conference on Computers and Accessibility, Pages 137–144, October 2008

   This paper presented the results of a user evaluation with a number of users. The results demonstrated that SADIe was a suitable platform for providing transcoding that was of benefit to visually impaired users. The contribution of Lunn was to design and run the evaluation and perform statistical analysis on the results.

8. **Combining SADIe and AxsJAX to Improve the Accessibility of Web Content** *Darren Lunn, Simon Harper and Sean Bechhofer.* In W4A '09: Proceedings of the 2009 International Cross-Disciplinary Conference on Web Accessibility, Pages 75–78, April 2009

   This paper described how the SADIe approach was flexible enough to not only manipulate the structure of the page but also use the AxsJAX framework to insert ARIA statements into the content. The contribution of Lunn was to adapt the transcoding engine to be able to make use of the AxsJAX framework while still retaining the underlying CSS annotations.

In addition to the peer reviewed publications described above, the results of the research conducted for this thesis have led to the publication of thirteen technical reports that describe the methods and analysis used throughout the project. A full listing of these reports is provided in Appendix K.

1.4 Remainder Of The Thesis

The remainder of this thesis describes the processes used to reach the findings outlined above, in addition to an analysis of the conclusions reached for each chapter. The work presented is solely the work of the author, with the following exceptions. The principle of using Cascading Style Sheets as an annotation method for the SADIe transcoding platform, discussed in Section 3.2, was developed by Harper & Bechhofer [2004]. Lunn used the approach to develop and implement the SADIe architecture and perform the technical evaluations.

The data used to validate the coping strategy definition, discussed in Section 4.3, was obtained by Craven & Brophy [2003]. Lunn used the data to act as independent third-party data that was unbiased towards the identification of coping strategies to test the validity of the framework being developed.

Chapter 2

Background and Related Work

In this chapter we review the literature surrounding Web content, Web accessibility and how users cope when Web content is not accessible. As we shall see, the Web began as a device independent medium for accessing documents. There was only a minimal notion of presentation, with a large focus on structure, and as such the first Web documents could be conceived as being accessible. As the Web evolved, multimedia elements were introduced that assumed users accessed the Web on large screens, hindering users who used devices that did not conform to this. However, now there is a trend back towards the separation of structure and presentation. This, in conjunction with Semantic Web technologies that annotate and describe Web resources, provides opportunities for creating a device independent, accessible Web.

A review of Web accessibility guidelines will show that, while they have helped to raise the awareness and profile of Web accessibility, they are not sufficient to provide Web pages that are accessible to all. This has led to research into transcoding, whereby Web content is adapted into a format that is suitable for Web enabled devices, in particular screen readers. While these approaches can aid access to Web content, broadly speaking the proposed solutions are either highly scalable to the detriment of accuracy, or highly accurate but lack scalability. Furthermore, the adaptations that are provided are based upon anecdotal evidence that designers have used to devise transformations that manipulate the page at a structural level aimed at ensuring that the page suits the needs of the access device and not the user.

Finally we will look at the phenomena of coping strategies. Whilst well understood within psychology, there has been little investigation into understanding coping strategies within Web accessibility. While some studies have tried to identify coping strategies, they have been limited in their approach and therefore further research is required into how such strategies can be used as a way to enhance transcoding functionality and Web accessibility.

2.1 Web Documents

The fundamental building block of a Web pages are the structured markup languages used to compose the documents. These languages provide a description of the data that is held within a document. As well as structure, Web documents can also be presented in a variety of ways, both for human consumption and for the various types of devices used to access Web content.

2.1.1 Structure

A markup language adds metadata to a text document. This metadata provides structure to the document or adds presentation to the text. The original language of the Web was the HyperText Markup Language (HTML), developed as a way of authoring and linking electronic documents [Musciano & Kennedy 2000]. The initial purpose of the Web was for scientists at the European Organisation for Nuclear Research (CERN) to share information with each other using the Internet as a method of transport. The original design principle was that HTML should convey the structure of the information within the document, not its presentation [Berners-Lee 1999], and as such, the first HTML specification contained no formatting tags [Berners-Lee 1992]. The markup available for HTML documents included tags for elements such as titles (`<TITLE>`), paragraphs (`<P>`), links (`<A>`) and menus (`<MENU>`) and was designed to create a structured document that could be displayed on any number of devices that were used to access it. It could be conceived therefore that accessibility is the soul of the Web and from the beginning, it was designed to be accessible [Harper *et al.* 2004].

As Web developers wanted more control over the presentation of their Web content, browsers, such as Mosaic, included their own specific presentation extensions to the HTML language [Lie 2005]. These extensions included tables and modification of the text, such as strike-through, bold and blink. As will be discussed in Section 2.3, a major cause of Web inaccessibility is the visual nature of Web pages. The tags that groups such as Mosaic added to the HTML specification facilitated the move from accessible structured documents to inaccessible presentation-orientated documents. Information contained within the page became implicit, available through the on-screen rendering, rather than explicitly defined using HTML markup. Furthermore, the device independence that the original HTML specification promoted was lost. It was not immediately obvious how tags such as blinking text (indicated by `<blink>`) would be rendered on a text-only access device, let alone a screen reader.

While HTML provided simple structure to pages, it had a fixed set of tags that limited the structure that could be applied to data within a document. When extracting data from a HTML document, a Web application can find the address data by parsing the document and looking for an `<address>` tag. Due to the limited tag set of HTML, it is not possible to do this for addressee or telephone number as these tags are not part of the specification. Therefore, to discover this data, Web applications must search the document and try to determine which data is the name and which data is the telephone number, rather than parse the structure.

The Extensible Markup Language (XML) describes rules for structuring information using embedded markup [Bray *et al.* 2006]. This gives the user control over how information is represented by allowing the creation of a vocabulary that more accurately describes the data elements and attributes contained within the information [Holman 2000]. XML tags are defined specifically for describing the content of the document and provide a richer structure to the document than the HTML tags do. Contact details consisting of a name, address and a telephone number can be explicitly stated, making it easier for machines to discover specific parts of the document.

The vocabulary used within an the XML document can be defined using a number of methods. One option is to use a Document Type Definition (DTD). This is a grammar language that was originally used to define Standard Generalized Markup Language (SGML) documents, of which XML is a subset [Goldfarb 1990]. Alternatively one may use XML Schema [Thompson *et al.* 2004], a World Wide Web Consortium (W3C) recommendation language that defines the structure of an XML document, but unlike DTD supports datatypes. Regardless of the method used, the purpose of defining an XML grammar is to define what is, and what is not, an acceptable use of the tags used within the XML document [Abiteboul *et al.* 1999]. As such, it becomes easier for different individuals and organisations to share and process XML documents. For example, an organisation may send its address to a customer in XML. Unlike HTML, the tags will be defined specifically for the address and so the customer, upon receiving the XML file, will be able to accurately extract data such as the name and telephone number and process it as required.

XML provides the ability to define vocabulary for a specific document structure. To exploit the markup structure of XML documents, several languages have been developed. These include XQuery, which allows XML documents to be queried in order to find specific data [Boag *et al.* 2007]; XLink, which allows links between XML documents to be created [DeRose *et al.* 2001]; and XSLT which is a transformation language to convert XML into alternative data formats [Kay 2007]. The disadvantage of XML is that it is more difficult to write XML documents than HTML documents, especially for novice Web authors who simply want to publish personal data on the Web. In addition, a large number of legacy Web browsers exist that do not support XML, limiting the potential number of users who can access XML Web documents [Musciano & Kennedy 2000]. Therefore, the W3C have developed the Extensible HyperText Markup Language (XHTML).

XHTML bridges the gap between HTML and XML by formalising HTML 4.01 so that it validates as an XML document [Pemberton *et al.* 2002]. This provides the user with the benefit of having a markup language that can use all the tools and languages designed for use with XML. In addition, XHTML uses many of the tags that were defined in the HTML 4.01 specification, allowing documents written in XHTML to still be accessible by legacy browsers [Raggett *et al.* 1999]. However, there are differences between HTML and XHTML. The first is that XHTML documents must be well-formed. Due to browser specific extensions, HTML parsers ignored tags that they did not understand, assuming them to be new tags developed for a specific browser. They also tended to be lenient with the quality of the markup. Web designers could mix tags so that `<p><b>some text</p></b>` was considered as valid as `<p><b>some text</b></p>`. In XHTML, only the latter would be classed as valid markup and only tags defined within the specification would be accepted [Pemberton *et al.* 2002].

A second major difference is that XHTML extends the attempts initiated in HTML 4.01 to revert back to the original goal of being a language that defines the structure of a Web document. To ease the transition from HTML to XHTML, the W3C provides different versions of the XHTML specification. XHTML Transitional supports the presentation elements found in HTML 4.01. However XHTML Strict treats presentation tags such as `<font>`, `<b>` and `<strike>` as deprecated. Instead presentation must be provided through Cascading Style Sheets.

While XHTML is the current stable recommendation for authoring Web pages, the W3C are currently in the process of developing XHTML's successor, known as HTML 5 [Hickson & Hyatt 2009]. HTML 5 is currently at the Working Draft stage and liable to change, however there are aspects currently proposed that may have a detrimental impact to accessibility. For example, part of the proposal of HTML 5 allows authors to be lenient with the quality of the markup. Unlike XHTML, both `<p><b>some text</p></b>` and `<p><b>some text</b></p>` would be valid markup as the proposal has rules that allow the user agent to generate implied end tags. Leniency of the quality of markup could hinder users of assistive technology as their tools would need to interpret what authors intended if the markup tags were not closed correctly. This could lead to potential errors in the interpretation of the document markup and incorrect information being conveyed to the user.

There are also a number of programatic features included in the proposal that are necessary for tags to operate correctly. For example there is an element called `<canvas>` that can be used to display graphics. However, unlike the standard `<img>` tag, `<canvas>` acts as a container and does not call an external graphic file to be displayed on the page. Instead, the element must be associated with JavaScript to dynamically render the image on the screen. This provides an advantage over `<img>` as images can not only be static, but also interactive. However, it is not immediately clear how `<canvas>` and its associated JavaScript could be made accessible to assistive technologies.

While HTML 5 does raise issues with regards to the access of Web content for people with disabilities, it should be noted that the presentation and layout of the elements contained within the page follows the same principle of XHTML and must be provided through Cascading Style Sheets.

2.1.2 Presentation and Transformation

One of the benefits of structured documents is that the content can be used in many contexts and presented in ways that suit the access device. A variety of different style sheets can be attached to the logical structure to serve different needs [Lie 2005]. One of the main problems with the later versions of HTML was that it combined the presentation and structure of a document with its content. This merger of style and structure made it difficult for machines to know which part of a document described the style and which part of a document described the content. Mixing presentation syntax with the Web document structure no longer made the Web device independent. For Web documents to be meaningful, a desktop PC with a large colour screen display was required to access them [Yin & Lee 2004].

Cascading Style Sheets (CSS) provide a way of separating the style and presentation of a HTML document from its content. A CSS file describes how the Web browser should display the data found within a Web document. It defines such things as font type; the size and colour of text; the positioning and size of images, margins, and borders; and a large number of other properties [Cranford-Teague 2004]. A major benefit of CSS is that it is relatively easy to redesign a Website. By changing the style syntax in a single location, the modified layout propagates through every page on the Website. In addition to this, CSS was also designed with multiple access devices in mind. Part of the CSS specification allows different style sheets to be defined for the same document so that it may be accessed by different media. Supported media types included aural media, which define how documents should be rendered on an audio device, print media for paper publications and projector media for presentations [Lie 2005].

Styling is the rendering of information in a form more suitable for consumption by a target audience [Holman 2000]. We have seen how HTML and CSS can be combined to produce a version of the HTML document that is more suitable for human users of the Web. CSS, however, is only a formatting language that attaches styles to elements of a HTML document. It assumes that no processing of the HTML data will be performed by a client device, and so lacks the facilities to manipulate data into a different yet more suitable form. With the advent of XML, the expectation is that client devices will receive the original data, and will therefore need to adapt and transform the data itself into a format that is more suitable [Lie & Bos 1998], as opposed to only applying a presentational style to the data.

The Document Object Model (DOM) is an interface that allows programs and scripts to dynamically access and update the content, structure and style of HTML and XML documents. The DOM represents a document as a tree structure, with each node of the tree representing an item within the document, complete with attributes, elements and processing instructions [Kay 2004]. It is possible for application developers to use the DOM to manipulate and process documents by using a standard set of Application Programming Interfaces (APIs).

Extensible Style Language Transformations (XSLT) is a W3C recommendation for performing DOM-like transformations [Kay 2007]. The XSLT language has two elements. The first is the Extensible Style Language (XSL), which is a language used for applying presentation and formatting to XML documents [Adler *et al.* 2001]. XSL works in a similar fashion to CSS and there is little difference between HTML combined with CSS and XML combined with XSL [Holman 2000].

The second element is the transformation of XML to alternative formats. It is possible to convert XML into legacy HTML code, complete with formatting tags, so that the data is viewable in legacy Web browsers. While XSLT is widely used to transform XML documents into HTML, it is also possible to convert XML documents into PDF files, plain text files or even VoiceXML files. By using XSLT, a designer can define the structure of their data once, yet have the ability to make their data available on multiple client devices using multiple data formats.

The design of XSLT was based on the recognition that a majority of the programs written to manipulate XML documents via the DOM API had similar functionality. Most made use of code that recursively traversed the tree representation of the XML document to find the node that contained the required data. As such, XSLT was created as a high level declarative language that used rules to generate output when particular patterns occurred within the input [Kay 2004].

2.1.3 Relationship To Our Research

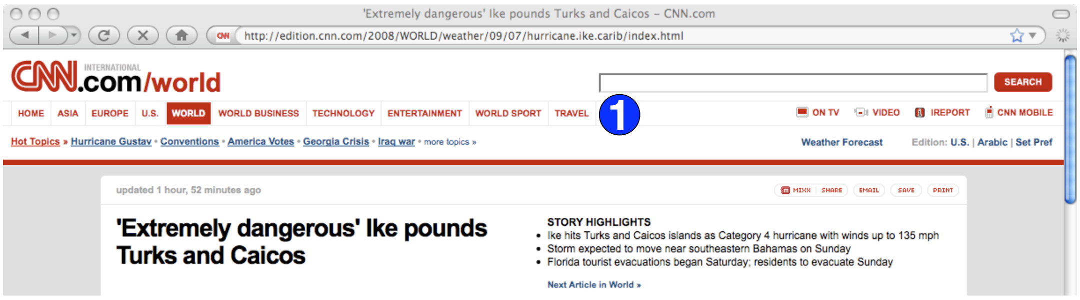
The trend towards a presentation orientated Web has hindered visually impaired users, in particularly the combining of layout and structural tags in a single document. Table elements, for example, are designed to present tabular data sets in a clear manner. When used as a layout mechanism, however, they can cause the content of the page to be read aloud in an inappropriate order that

```
<ul class="cnnNavigation">
   <li><a href="/">Home</a></li>
   <li><a href="/ASIA/">Asia</a></li>
   <li><a href="/EUROPE/">Europe</a></li>
   <li><a href="/US/">U.S.</a></li>
     ⋮
   <li><a href="/TRAVEL/">Travel</a></li>
</ul>
```

(a) XHTML that defines the CNN Menu. Note how it is an unordered list with no explicit syntax to indicate that it is the menu.

```
.cnnNavigation
   {
    margin:  0;
    padding:  0;
    list-style:  none;
      ⋮
    white-space:  nowrap;
   }
```

(b) CSS that renders the CNN Menu. This adds implicit meaning to the unordered list to indicate that it is the menu.



(c) Combining (a) and (b) results in the CNN Website menu, indicated as 1.

Figure 2.1: The combination of XHTML document structure and CSS rendering information can add implicit semantics to a Web page. In this example, users can identify a list of links as the menu of the page due to the way it is presented.

can confuse the user [Asakawa 2005]. The separation of style from presentation, with the advent of XHTML and CSS, has aided Web accessibility. Assistive technologies are presented with structural elements that can be interpreted in an appropriate manner [Asakawa & Itoh 1998; GW Micro 2007; Freedom Scientific 2007]. However, the content contained within the Web document can contain additional meaning not available in the tag set available within XHTML. An example of this is highlighted in Figure 2.1 on page 33. The area indicated as 1 in Figure 2.1c can be considered to be the navigation menu of the CNN Website[1]. Sighted users are aware that this is the menu of the Website due to the way it is rendered on the screen. The colour, layout and presentation distinguish the group of links from other elements contained within the page. In the underlying XHTML however, the menu is not syntactically marked-up as a `<menu>` element as the current version of XHTML does not support this tag [Pemberton *et al.* 2002].

[1] URL: `http://www.cnn.com/2008/WORLD/weather/09/07/hurricane.ike.carib/`. Last accessed: 7th September 2008

Instead, it is defined as an unordered list of links, as shown in Figure 2.1a. It is only when the XHTML is combined with the appropriate rendering information, shown in Figure 2.1b, that the user can easily determine the role that the element plays within the Website. The CSS adds implicit information about the element's purpose in addition to the structural meaning explicitly defined using XHTML markup.

It is this relationship between CSS elements and XHTML elements that we will exploit to further improve Web accessibility. By exposing the implicit information and making the semantic meaning of elements explicit, tools such as screen readers will be able to inform a visually impaired user which element of the page they are interacting with. Users will be aware if the current element is the menu, the footer of the page, or the main content, and be able to make more informed decisions as to what the purpose of the element is and if they wish to continue interacting with it.

2.2 Exposing Semantics Of Web Content

Web pages contain information that has meaning to a human reader but is not available to machines. By adding metadata to Web resources that expose this information, tools will have a deeper understating of the content of the document [Nagao 2000]. This will allow for more accurate search results as search engines are aware of the meaning of the content, or improved access to content for people who use assistive technologies to interact with the Web.

2.2.1 Annotation

An Annotation is an assertion attached to a portion of a document [Abe & Hori 2003]. Annotations can describe the meaning of elements within a page so that it is possible for tools to manipulate the page to meet the end user's needs. Groups of links may be annotated as a navigation menu, images may be annotated as advertisements or paragraphs annotated as the main content of the page. By analysing the annotations, a tool can adapt the page to better suit the variety of devices used to access the Web.

The Resource Description Framework (RDF) is a data model and syntax that represents metadata and provides a simple yet flexible framework for annotating and describing properties of Web resources [Abiteboul *et al.* 1999; Kahan &

Koivunen 2001]. The data model consists of triples, also known as statements, of the format `(Resource, Property, Value)` where a `resource` is any object that has a Unique Resource Identifier (URI); a `property` expresses a relationship between the resource and a value; and a `value` may be an atomic value, such as a string, or other resources [Manola & Miller 2004].

In order to provide metadata with a Web resource, methods of embedding RDF within XHTML documents have been investigated [Palmer 2002]. One advantage of this is that by describing a resource using RDF, more information can be provided to search engines. Therefore searching will be more focused than is currently possible because of the vast amount of unstructured information that is on the Web today [Miller 1998]. One method of embedding RDF within XHTML was the Tag Project [Berners-Lee 2002]. However, XHTML documents containing RDF would not validate as XHTML and so did not find favour among the community [Kew 2001]. The project concluded that the RDF specification specifies how to understand the semantics (in terms of RDF triples) in an RDF document that contains only RDF. However, it does not explain how and when semantics can be extracted from documents in other namespaces that contain embedded RDF. Furthermore, the XHTML specification explains how to process XHTML namespace content, but gives no indication about how to process embedded RDF information [Berners-Lee 2002].

As an alternative to embedding RDF statements within the XHTML document, several proposals including Annotea [Kahan & Koivunen 2001], Annotation Based Transcoder [Hori *et al.* 2000] and the Framework for Annotation Tools [Hori *et al.* 2003], have suggested the use of external RDF statements. Such approaches typically consist of a database of RDF triples that provide the annotations for the Web documents. These are then associated with the content of the Web page through the use of XPath, a W3C recommendation language for addressing parts of an XML document [Clark & DeRose 1999]. The benefits of external, remote metadata is that a document can be annotated without document ownership. Metadata associated with a Web resource can also be shared by multiple users. The difficulty that arises with the use of external RDF statements is that there will always be two parts of a document, which can cause problems of synchronicity [Berners-Lee 2002].

While RDF annotations can describe the semantics of Web content, a number of problems can arise when associating RDF with Web documents. Embedding

RDF annotations to current versions of XHTML can cause the document to become invalid whereas external RDF annotations suffer from synchronicity issues. As an alternative, Microformats have been proposed as a method of adding semantics to Web pages. Microformats circumvent the problem of XHTML validity by using facilities that already exist within the XHTML specification, such as the `class` and `title` attribute of elements [Adida 2008]. For example, in XHTML one can write a paragraph as `<p>The University of Manchester...</p>`. To tools consuming this data, the `<p>` element represents a paragraph of text with no indication as to any meaning of the content contained within the tag. Microformats can be added to this, such as `<p><span class = "org">The University of Manchester</span>...</p>`, to annotate `The University of Manchester` as an organisation. By using existing tags and attributes found in XHTML in this manner, different applications and services can reuse the information held within the document while still retaining document validity [Flores *et al.* 2006].

To allow tools to interpret the Microformat annotations correctly, several specifications exist that define the values that the XHTML attributes should make use of. Specifications include hCard [Çelik & Suda 2008] to enable applications to retrieve information about events directly from Web pages and hResume [King 2008] to represent fields found within online résumés. While such Microformats can encode explicit information to aid machine readability, they are not independent of each other and having multiple Microformats in a single Web document can create conflicts [Khare & Çelik 2006]. Both hCalendar and hResume, for example, contain a property `summary`. However the meaning of `summary` for the two specifications is different, with hCalendar referring to a summary of an event and hResume referring to a summary of work experience.

Similar to Microformats, Resource Description Framework Attributes (RDFa) allows the embedding of metadata into XHTML documents [Adida *et al.* 2008]. Unlike RDF, the use of RDFa still maintains validity in XHTML documents by using attributes associated with the tags. These attributes include `about`, `property` and `typeof`. Returning to the previous example, RDFa could be added to XHTML as `<p><span property = "foaf:Organization">The University of Manchester</span>...</p>`. Note how this is similar to Microformats. However, whereas Microformats define both the syntax and terms used in each specification, RDFa only describes the syntax [Adida 2008]. The terms used are independent and associated with a namespace, in this example the Friend of a

Friend (foaf) namepsace. This reduces the possibilities of conflicts because the namespace can be used to differentiate the meaning of the terms used when similar naming schemes are used, for example hCalendar summary and hResume summary.

2.2.2 Ontologies

Annotations can add metadata to a Web document and are a way of expressing the structure and meaning of a Web page. For machines to be able to process and use annotations effectively, annotations should be used consistently. A menu on a Web page could, for example, be annotated as `menu`, `sidebar`, `linked list` or `navigation menu`. A tool would need to be aware of all these different ways of describing the same element in the document. Therefore, to help tools take into account multiple ways of describing the same data, a set of defined terms and vocabulary is required. This can be provided by an ontology.

An ontology is an abstract model of the world that defines the properties of important concepts and the relationships that exist between them [Baader *et al.* 2004]. This can be useful for annotation because an ontology can define a common vocabulary, such as only allowing the concept of `menu`, that can be used when annotating menu elements in Web documents [Bechhofer *et al.* 2001]. When there are multiple concepts, an ontology can show how the concepts are related. An ontology for Web document terms could have the concept of `menu` as well as concepts for `sidebar` and `navigation menu`. To show that these three concepts all refer to similar elements within Web documents, a relationship can be created to show that `menu is-a-kind-of navigation menu` and `navigation menu is-a-kind-of sidebar`. Using these relationships, tools can infer the knowledge that `menu`, `sidebar`, and `navigation menu` all refer to similar elements within the Web document and that an operation designed to be performed on `menu` can also be performed on `sidebar`, and `navigation menu`.

Ontologies allow a structure to be created that allows the meaning of the resources to be explicitly defined and therefore more accessible to machines, rather than implicitly defined and accessible only to the human reader [Harper & Bechhofer 2005]. With metadata accessible to machines, automated tools are able to use the resources on the Web to discover information and knowledge in a more accurate and precise way than is currently possible. To provide a standard way of representing knowledge for the Web, the W3C has created the Web Ontology Language (OWL) [McGuinness & van Harmelen 2004].

OWL allows designers to describe the data and resources that are on the Web and provides mechanisms for these descriptions to be reasoned over. By reasoning over OWL ontologies one can infer additional knowledge about the resource, such as discovering new classifications of concepts within the resource based on logical assertions [Tsarkov & Horrocks 2006]. Reasoning can also discover if a resource is logically inconsistent, as the reasoner can discover contradictions within the model of the data. The contradictions may be due to human error in creating the model or may be because the model itself is incorrect. Either way, the reasoner can bring this to our attention allowing corrections and redesigns to be carried out as is necessary. Once reasoning over the OWL ontology has been performed, tools designed to manipulate and communicate with the resource can use any discovered knowledge to access and manipulate the data held within the Web resource automatically, without the need for human intervention.

2.2.3 Relationship To Our Research

As discussed in Section 2.1, Web documents contain implicit semantics that are only available to those who can see the rendering on the screen. Annotation mechanisms and ontologies provide a method for the implicit semantics of the page to be exposed so that they are available to assistive technologies and therefore visually impaired users. While embedding annotations within the XHTML document would enable us to explicitly define the roles that each of the elements of the page plays, a drawback of this approach is that document ownership is required. In addition, reengineering the large number of existing Web pages to contain metadata is an unrealistic proposition.

Associating annotations to Web pages without ownership is possible, although existing approaches have tended to use XPath in order to achieve this. As discussed previously, this can result in issues with synchronicity. If the structure of the document changes, then the XPath may break, resulting in the incorrect part of the document being annotated.

However, we can exploit the association between to the XHTML and the CSS to provide a mechanism for annotating the content of the page. Web developers associate XHTML tags to CSS elements to provide a specific rendering and layout for the tag. This is typically achieved using the `class` or `id` attribute. To ensure that elements have consistent styles across all the pages of a Website, XHTML tags that represent similar roles will have the same attribute values. This is similar to a Microformat, but rather than a publicly available global standard, it is localised to terms defined for the Website by the designer.

Websites will have a bespoke set of terms used to associate XHTML tags with CSS elements. However, we can use an ontology to provide a defined set of concepts for annotating the role of the CSS elements. The ontology can contain roles that we are interested in exposing, such as `MainContent`, and then this can be used to describe the role that the CSS element is implicitly playing on the Web page. The benefit of such an approach is that document ownership is not required as the ontology is a separate entity from the Web document. Furthermore, the annotations are not brittle as we are not annotating specific areas of the document using XPath. Instead, we are describing the role that a particular CSS element plays, which is itself embedded within the document using the `class` or `id` attribute. The structure of the document may change but the association between the ontology definition, the CSS element and the XHTML tag remains the same.

2.3 Web Accessibility

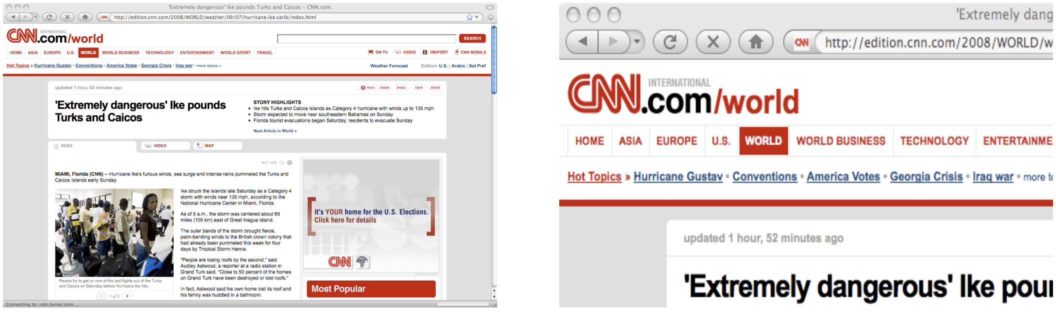
There are multiple definitions of the meaning of accessibility. Bergman & Johnson [1995] define accessibility as *"removing barriers that prevent people with disabilities from participating in substantial life activities, including the use of services, products, and information"*. The World Wide Web Consortium (W3C) narrows this definition to focus on accessibility of the Web, defining accessibility as *"allowing people, no matter what their disability, to perceive, understand, navigate and interact with the Web so they can contribute to the Web"* [Chisholm *et al.* 2000].

At the centre of these definitions lies the word "disabilities." When thinking of disabilities, it is usually the case that extremes of disability are considered, such as a person who is entirely without sight or a person who has complete loss of hearing. However there is no clear distinction between a person being disabled and not being disabled as disabilities can be a temporary phenomenon. A person with a broken hand is faced with the same difficulties in using a mouse as a person with a missing limb [Bergman & Johnson 1995]. Indeed the Disability Discrimination Act (DDA) 1995 defines a disability as a *"physical or mental impairment which... effects [a person's] ability to carry out normal day-to-day activities"* [UK Government 1995]. Therefore anything that can act as a hindrance to a person's activities can be considered as a disability. A person who has a slight visual impairment may have as much difficulty reading a Web resource as a person who is completely without sight. Their needs will be different. In order to read the page, larger font sizes may be sufficient for the person with a slight visual impairment, whereas a screen reader may be required for the person who is fully without sight. Nevertheless, both are still disabilities that may prevent people from accessing Web resources. Software systems and Web pages must be designed to cater for this and be adaptable in order to meet the different needs of people in different situations using a variety of assistive technologies.

2.3.1 Assistive Technologies

Assistive technologies are pieces of hardware or software that increase or improve interaction with computers for people with disabilities [Burks *et al.* 2006]. Assistive technologies include both input and output mechanisms and the functionality that they provide is dependent upon the needs of the user. Stead Clicks [Trewin *et al.* 2006], for example, is software that can assist users who have mild motor impairments and have difficulty with fine mouse movements. As the user is clicking on the mouse, the software freezes the cursor location on the screen so that the click action can be completed even if the user accidentally moves the mouse. For users with more sever motor impairments, such as spinal cord injuries, verbal input mechanisms have been developed. FreeDraw [Harada *et al.* 2007] allows users to control a mouse through non-speech sounds. By making continuous sounds in various pitches and tones, the user can move the mouse around the screen with more fluidity and accuracy than speaking words alone. Speaking the word "left" may cause the mouse to move but it is not clear to the system have far left it should move. By using sounds, the mouse moves until the user stops making the noise. By changing tone, the system can also change the direction of the mouse, allowing more fine grained control than is possible by speaking the four directions of "left", "right", "up", and "down".

People who have learning or communication difficulties can make use of text prediction techniques such as those developed by Baljko & Tam [2006], Felzer &



(a) A Story from the CNN Website viewed at normal resolution.

(b) A Story from the CNN Website viewed at 3× resolution.

Figure 2.2: Comparison of viewing a Website using normal resolution and a screen magnifier. The magnified version requires the user to scroll in both directions to gain an overview and context of the content.

Nordman [2006], and Wandmacher *et al.* [2007]. Such systems allow users to enter words into a system with a minimal number of key strokes and hands movements. The text can then be used in traditional text-based applications such as word processors or, if users have communication difficulties, used in conjunction with speech synthesisers to allow users to engage in conversations with other people.

For people with visual impairments, the most prominent forms of assistive technology available are screen magnifiers and screen readers. Screen Magnifiers "zoom in" on portions on the page so that areas are magnified and displayed using lager fonts and graphics [Alliance for Technology Access 2000]. Common commercial screen magnification software includes Lunar[2], ZoomText[3], and Lightning[4]. Screen magnifiers typically have three methods of zooming for the user: Lens magnification whereby the user moves the cursor around the screen and the area immediately below the cursor is enlarged; part-screen magnification whereby users move the cursor around the screen but the area to be enlarged is displayed in a dedicated part of the screen; and full-screen magnifiers which enlarge the entire screen [Blenkhorn *et al.* 2002]. Figure 2.2 on page 41 shows a comparison between viewing a CNN news story[5] using both normal resolution and the screen magnified to 3× the normal resolution. Note how in the magnified screen, shown

[2]URL: `http://www.yourdolphin.com/productdetail.asp?id=3`. Last accessed: 25th November 2009

[3]URL: `http://www.synapseadaptive.com/aisquared/`. Last accessed: 25th November 2009

[4]URL: `http://www.xpscreenreader.com/`. Last accessed: 25th November 2009

[5]URL: `http://www.cnn.com/2008/WORLD/weather/09/07/hurricane.ike.carib/`. Last accessed: 7th September 2008

in Figure 2.2b, users have to use the cursor to scroll the screen both horizontally and vertically to see all the content. While this magnification can generally help users with low vision, users can lose contextual information as only a small portion of the screen is viewed at any one time. In addition, alert boxes may appear that are not in the current viewing area. The application may become unresponsive while it waits for input from the user but the user may not be aware that the application requires their attention [Edwards 2008].

Screen Readers are applications that use speech synthesisers to read aloud everything on the screen, including content, menus and buttons [Alliance for Technology Access 2000]. Common commercial screen readers include JAWS[6], Window-Eyes[7], and Hal[8]. The first screen readers used low level graphics calls by applications to create an off screen model of what content was present on the screen [Ramen 2008]. While this provided users with access to content, it was not ideal, especially for Web access. Hyperlinks were difficult to identify [Asakawa & Itoh 1998] and if text was presented in columns or frames, it was difficult for users to gain a comprehensive and cohesive understanding of the page contents [Nguyen & Petty 1996]. As screen readers matured, they no longer read what was rendered on the screen but used the underlying HTML tags and document objet model to create an audio version of the page content [Burks *et al.* 2006]. By accessing the HTML tags, audio rendering could be generated from the markup instead of by guessing from visual representations allowing users to easily access hyperlinks and page headings [James 1998]. JAWS, for example, allows users to press the `F5` key to generate a "Heading List", which is a listing of all the headings that are available on the page. Users can then use this to gain an overview of the page content and quickly access different sections without the need to read through the entire page.

While screen readers directly accessing the underlying HTML structure have improved access to Web content for visually impaired users, problems still exist. Pages that are badly marked up prevent tools such as JAWS' Heading List from functioning correctly. In addition, Web technologies advance at a faster rate than assistive technologies. The Web is currently evolving from static pages into

[6]URL: `http://www.freedomscientific.com/jaws-hq.asp`. Last accessed: 9th November 2009

[7]URL: `http://www.gwmicro.com/Window-Eyes/`. Last accessed: 25th November 2009

[8]URL: `http://www.yourdolphin.com/productdetail.asp?id=5`. Last accessed: 25th November 2009

interactive Websites that include dynamically updating regions of content within the page. Evidence suggests that screen readers can not yet cope with dynamic content and any updates and content changes are lost to the user [Brown & Jay 2008]. Indeed studies that have investigated how different disability groups access Web content have noted that of all the disability groups surveyed, it is blind users using screen readers who have most difficulties as Websites are not designed with their needs in mind [Petrie *et al.* 2004; DRC 2004].

2.3.2 Accessible Web Content

In order to raise awareness of the issues of accessibility and to allow access to the Web for people with disabilities, a number of guidelines have been produced. These include Dive Into Accessibility [Pilgrim 2002] and See It Right [RNIB 2004] that focus, to a large extent, on provisions for visually impaired users. A generic set of accessibility guidelines, aimed at all user groups, has been created by the W3C Web Accessibility Initiative (WAI). These guidelines, known as the Core Techniques for Web Content Accessibility Guidelines 1.0 (WCAG 1.0) range from advice on the design process, such as separating the structure of the document from its presentation, right through to the testing process, such as testing the Web page on as many system configurations as possible [Chisholm *et al.* 1999]. Combined, there are 65 checkpoints with each checkpoint having a priority rating of either 1, 2 or 3, with 1 being issues that must be addressed and 3 being issues that may be addressed. The 65 checkpoints are aimed at a multitude of disabilities. For example, checkpoint 7.1 states that designers should *"avoid causing the screen to flicker. [Priority 1]. People with photosensitive epilepsy can have seizures triggered by flickering or flashing in the 4 to 59 flashes per second range"* However, a vast majority of the checkpoints are concerned with people with visual impairments. This is broadly in line with observations from previous studies that have noted that visually impaired users are the most disenfranchised group of Web users.

In December 2008 WCAG 2.0, a revised version of WCAG 1.0, was released as a W3C Recommendation [Caldwell *et al.* 2008]. Unlike WCAG 1.0, WCAG 2.0 does not have a list of checkpoints, but has a set of four principles that are necessary for Web accessibility. To be accessible, a page must be *perceivable*, *operable*, *understandable* and *robust*. Each one of these four principles has a number of guidelines. For example, to be *understandable*, a Web page must *"make text* *content readable and understandable"*, *"appear and operate in predictable ways"* and *"help users avoid and correct mistakes"*. Each of these guidelines is followed by both machine and human testable success criteria. For example *"appear and operate in predictable ways"* has five success criteria including *"when any component receives focus, it does not initiate a change of context"* and *"navigational mechanisms that are repeated on multiple Web pages... occur in the same relative order"*.

WCAG 1.0 focused on static HTML documents that could be determined as accessible by conforming to a set of checkpoints. However the Web has evolved to include dynamic content and a diverse range of technologies such as JavaScript, Flash and AJAX. By using principles and guidelines rather than a set of stringent checkpoints, WCAG 2.0 provides accessibility mechanisms that are applicable to all technologies [Reid & Snow-Weaver 2008]. However it should be noted that the checkpoints in WCAG 1.0 are mappable to guidelines and success criteria in WCAG 2.0. For example, the WCAG 1.0 checkpoint *"provide a text equivalent for every non-text element"* is defined in WCAG 2.0 as *"All non-text content that is presented to the user has a text alternative that serves the equivalent purpose"*, Unlike, WCAG 1.0, WCAG 2.0 then proceeds to list exceptions. For example Completely Automated Public Turing Test to Tell Computers and Humans Apart (CAPTCHA) is a method of ensuring that a user of a Website is human and not an automated Web crawler. This is achieved by asking users of the Website to enter the letters displayed in a distorted image that machines cannot process [von Ahn *et al.* 2003]. Providing text alternatives for CAPTCHA images would not only allow CAPTCHAs to be accessible to screen reader technology but also to automated tools and therefore allow Web crawlers to bypass the CAPTCHA security feature.

WCAG 1.0 has provided a solid basis for the implementation of accessible Web design and has helped to raise the awareness and profile of Web accessibility [Kelly *et al.* 2005]. Yet, for a variety of reasons, there are still Websites being designed that are inaccessible to people with disabilities. While WCAG 2.0 may further improve accessibility, the issues discussed below are still relevant for a number of reasons. Firstly, as shown by Chen [2008], the adoption rate of Web technologies significantly lags behind the release date of the standard or recommendation. As WCAG 2.0 has only recently been released as a W3C Recommendation, one can assume that it will be a number of years before there is widespread adoption amongst Web developers, if indeed at all. Some studies suggest that part of the problem with the uptake of accessibility may lie with designers not fully understanding the issues of accessibility [DRC 2004]. This confusion and misunderstanding manifests itself in the way various people have described accessible design. Some believe that accessible Web design is essential, others believe that it is absolutely impossible. There have also been suggestions that accessibility is a good idea whose time has not yet come whilst some designers flatly refuse to consider accessibility, arguing that their Web pages are not designed for blind people to use [Meyer 2005].

Secondly, while WCAG 2.0 attempts to be a more generic set of accessibility principles applicable to all Web technologies, it still builds upon and adopts a majority of the checkpoints from WCAG 1.0. Therefore many of the issues surrounding WCAG 1.0 are present in WCAG 2.0. Kelly *et al.* [2005] entered into discussions with Web designers on mailing lists and Web forums to allow them to express their thoughts and reservations with regards to WCAG 1.0. There was general acknowledgement amongst designers that the WAI has been successful in raising the importance of Web accessibility and that WCAG guidelines have been a successful tool. However, there were outstanding issues which the Web content authors felt the guidelines needed to address. These included the guidelines being too theoretical, too ambiguous, too complex and, in some instances, logically flawed.

There have also been a number of studies that demonstrate the weaknesses in guidelines. Petrie & Kheir [2007] conducted studies with sighted and visually impaired users and concluded that Web accessibility guidelines provide a valuable starting point for designers trying to ensure that visually disabled users are able to use their Web sites. Unfortunately, conforming to the guidelines in no way guarantees that a Web site is genuinely accessible.

Dalal *et al.* [2000] asserted that a reason for the guidelines failing to provide accessibility was that whilst there were numerous guidelines on Web design available, a large majority of the guidelines were based upon designers' intuition and common sense, as opposed to guidelines based upon theoretical and experimental validation. This contradicted some of the findings of Kelly *et al.* [2005] where it was believed that part of the problem with guidelines is that they are too theoretical. Dalal *et al.* [2000] believed that most of the problems associated with inaccessible Web pages were due to underlying design flaws, some of which included over-reliance on multimedia and meaningless link labels such as "Click here". Web pages that were structured, well organised and clear were much more usable and accessible than those that followed the more traditional guidelines [Dalal *et al.* 2000]. This conclusion was supported by observations by Harper *et al.* [2004] where it was noted that a Website could pass all the accessibility guidelines and standards but still be impossible for users to navigate and find the sort of information that they require.

2.3.3 Accessible Web 2.0 Content

Accessibility guidelines and established structure-driven transcoding techniques are targeted at traditional static Web content where information can be accessed in a linear fashion. While a majority of the Web still consists of static-based documents, there is a trend towards interactive Websites, known as Web 2.0, where users actively contribute to the content [Millard & Ross 2006]. Web 2.0 Websites include dynamically updating regions within the page that break the linear access assumption [Thiessen & Chen 2007]. To allow assistive technologies to communicate with updating regions, the W3C are currently developing Accessible Rich Internet Applications (ARIA) [Cooper *et al.* 2008].

ARIA is a mechanism that allows developers to enhance the accessibility of rich content contained within their Website by defining the roles, states, and properties of interactive elements contained within the page. These elements include user interface widgets, such as navigation menus, and updating content areas such as news tickers. By assigning roles to areas of the page, assistive technologies can provide information about the element to the user. Consider the the CNN navigation menu discussed previously in Figure 2.1 on page 33. Using ARIA, we could add an additional property to the element, such as `<ul class="cnnNavigation" role="navigation">`, to explicitly state that this is the navigation element of the page. Assistive technologies can then use this role property to provide users with easy access to the navigation area of the Website. ARIA can also assist users when they are using interactive elements contained within the page. Forms, for example, can have buttons that are either selected or not. ARIA can explicitly define the current state of the form element. For example `<input type="radio" role="radio" aria-checked="true"/>` informs the user that the current element is a radio button and that its current state is true. The user can then change the state by unchecking the radio button.

ARIA can also be used to allow users to interact with content that updates itself dynamically. For example news Websites, such as CNN, often have a "ticker-tape" at the top of the page that automatically displays the latest news stories that are available on the Website. These updates can occur without the page being refreshed and so users of assistive technology may not be aware that an has update occurred. With ARIA, not only can the live region be explicitly defined but a level of importance can be assigned. For example `<div class="ticker" aria-live="assertive">` states that the `<div>` element is a live region that can automatically update. The property `assertive` tells the assistive technology that when an update occurs, it should inform the user immediately regardless of what they are currently doing on the page. Other properties include `polite`, which tells the assistive technology to wait until the user is not currently doing anything before informing them of an update or `off` which ensures that the user will not be informed of any updates.

ARIA requires designers to modify their existing Web content and insert the additional semantic properties required for allowing assistive technologies to interact with the Web 2.0 resources. To insert ARIA statements into Web content, designers need to have ownership of the document and have a full understanding of how the ARIA markup affects assistive technology. For example, ARIA allows updates to be given a priority which effects when the assistive technology informs the user that an update has occurred. Informing the user immediately as all updates occur can be overwhelming to the detriment of the users enjoyment of the Website [Thiessen & Chen 2007]. To aid Web 2.0 developers, Google have developed the AxsJAX Framework [Chen & Ramen 2008]. This is an open source framework that allows developers to use high level patterns developed from the underlying ARIA markup. The framework provides open source JavaScript code that already contains the necessary elements required to insert ARIA into Web content. Designers merely define a series of Content Navigation Rules (CNR) that use XPath and XML statements to inform the framework where ARIA statements should be inserted in the page. For example, the content navigation rule `<cnr><list next='RIGHT j' prev='LEFT k'><item>//ul[@class='nav']</item></list></cnr>` would look for an an unordered list element in the HTML with a class attribute value of `nav`. As the CNR states that this is a list, the appropriate ARIA will be inserted to allow users to access the list and move through the elements using the `j` and `k` keys. The framework not only ensures that valid ARIA is inserted into the Web page, but it also allows for ARIA statements to be inserted into the page dynamically. This allows developers and other community contributors to enhance the accessibility of Web 2.0 based pages without the need for document ownership.

2.3.4 Relationship To Our Research

People with disabilities can access Web content through assistive technologies which are pieces of hardware or software that can improve interaction with computers [Burks *et al.* 2006]. While assistive technologies have been of great benefit in allowing access to Web content, there are pages that still remain inaccessible to disabled people, especially those who make use of screen readers. In order to rectify this situation, the W3C have developed the Web Content Accessibility Guidelines, which are designed to help developers create accessible Web content. Although these guidelines have been successful in raising the awareness of accessibility amongst designers and contributed to a more accessible Web, there are still Websites being designed that are inaccessible to people with disabilities. There are a number of reasons that have been proposed for this. These include a lack of awareness amongst developers to more deep flaws within the guidelines themselves, where it has been observed that a Website can pass all the accessibility guidelines yet still remain inaccessible [Harper *et al.* 2004]. Therefore, methods of transforming Web content into a format more suited to the sequential, audio representation that screen readers present to users are desired to further enhance access to Web content for visually impaired users.

2.4 Transcoding

Transcoding is a way of transforming Web content so that it can be accessed on a diverse range of devices [Ihde *et al.* 2001]. In adapting Web content, transcoding systems use a variety of architectures, a range of methods and have a diverse target user group.

2.4.1 Architecture

Figure 2.3 on page 49 shows the three general locations that transcoding systems can adopt when adapting page content. In a client-side architecture, indicated

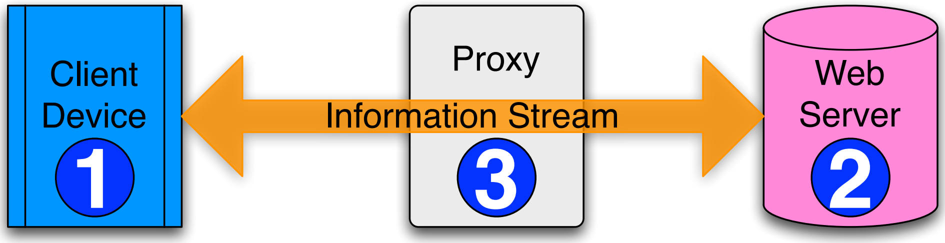


Figure 2.3: Transcoding tools can be implemented on the client, located at ①; the server, located at ②; or as a proxy, located at ③.

as ①, the transformations are executed on the client device, which provides the benefit that the Web content can be adapted to suit the needs of that specific device. It also allows users to set the software in such a fashion that it is tailored to their needs [King 2006]. The disadvantage is that every user that wants to use the transcoder must install a copy on their personal device. Additional processing of content is also required on the client-side as all the Web content is downloaded before transformations are applied.

With server-side transformations, indicated as ②, the transcoder operates on the server, adapting all the content for every user before it is sent across the network. This reduces processing on the client and allows different users to access different versions of the same content [Seeman 2004]. However, transformations are limited to the content that resides on that server [BBC 1999].

Proxy servers and intermediaries, indicated as ③, are software programs or agents that meaningfully transform information as it flows from one computer to another [Maglio & Barrett 2000]. Instead of accessing the Web page directly, a proxy server intercepts the communication between the client device and the Web server and adapts the content before it is rendered on the client device [Carr *et al.* 1998]. The proxy allows multiple users to access the transcoder, but the disadvantage is that some functionality, such as redirects and secure connections, may be lost [King *et al.* 2004].

2.4.2 Approach

Broadly speaking, transcoding solutions are based upon one of two approaches. The first, known as heuristic transcoding, analyses a page and adapts it based on a set of predefined rules. Chen *et al.* [2001] made use of heuristics where it was asserted that a Web page was composed of objects, such as presentation objects, interaction objects and hyperlink objects. By discovering what the functionality of the objects was, suitable transcoding rules could be applied. For example, small images were considered as presentation objects, and therefore removed.

Gupta *et al.* [2003] exploited the DOM to apply transcoding to the Web document. The document was parsed into a DOM tree and the nodes traversed in order to identify content. For example, table cell nodes that had a large link to paragraph ratio were considered to be “link lists” and removed from the page.

Parmanto *et al.* [2005] made the premise that humans interact with familiar environments better than non-familiar ones. This tendency means that Websites that provide similar functionality all use similar templates. News Websites all have a similar look and feel and e-commerce sites all use similar styles. By discovering the genre of Website that the user was accessing, the transcoding system could match a template to the site in order to exploit the underlying structure of the Web document so that the content could be rearranged. Jang *et al.* [2003] used templates to create pages for different devices. A template that suited the device capabilities was created and used as the basic model when adding and reorganising content from different Websites.

The second approach is semantic transcoding, which is the adaptation of a Web page by using the semantics of the structure or content to make better adaptation decisions [Hori *et al.* 2000]. Whereas heuristic transcoding makes educated guesses as to how the page should be adapted, semantic transcoding uses metadata that describes the structure of the page to make adaptation decisions. To help capture the metadata, many semantic transcoding systems annotate Web pages to capture the structure of the page and the knowledge held within it [Nagao 2000].

Piggy Bank, developed by Huynh *et al.* [2005], made use of RDF annotations of a Web page to allow users to extract data from the page and combine it with several other data sources. If RDF was not available for the Website, then Piggy Bank could make use of Really Simple Syndication (RSS) feeds that are available on a large number of Websites. These are XML-based documents that describe the content found within Websites, in particular Websites that have frequent content updates such as news Websites and Web Logs (Blogs) [Winer 2007]. By providing access to the data contained within the Web page, Piggy Bank allowed users to search and view different data sources simultaneously and then compare and combine the information that was available.

The transcoding proxy, developed by Takagi & Asakawa [2000], used three methods of annotation to provide users with access to page content: *volunteer-specified*, which identified visual fragments and assigned them a level of importance; *automatic*, which added alternative text to images and warned the user of JavaScript; and *user-specified*, which allowed the user to specify the starting point of the main content. With the annotations in place, transcoding could then be applied to the page. For example, users could simplify the page by reorganising the content into a more suitable format. However, for the transcoding to work, every Web page visited required annotations in order for the page to be adapted by the transcoding proxy.

In addition to identifying content importance, annotations can also identify roles elements play within the page [Mukherjee *et al.* 2003]. Towel compared the notion of travelling in the real world to travelling within a virtual environment [Harper *et al.* 2003]. To aid travel, the XHTML of the Web page was annotated so that the roles of the elements could be identified. For example, *cue contexts* were items that drew the user's attention to relevant pieces of information on the page. *Obs contexts* were obstacles that could cause problems to the traveller. By explicitly stating the purpose of the elements within the page, Towel could help navigate the user through the components and structure of the Web page, including links to other Web pages and resources.

2.4.3 Transformations

Transcoders apply a series of transformations to Web content to enable a variety of devices to access the content. Consider the BBC news story[9] shown in Figure 2.4a on page 53. One can assume that this page was designed to be accessed on a high resolution, high colour device. Therefore, transcoding engines have been developed to allow such content to be accessible to mobile devices [Chen *et al.* 2003; Yin & Lee 2004], screen readers [Takagi *et al.* 2002; Yesilada *et al.* 2004] and devices using low bandwidth connections [BBC 1999; Boldt 2000]. While the target devices have been diverse, the transformations applied have been similar and can be described as *Chunking*; *Filtering*; *Reordering*; *Scaling*; and *Text only.*

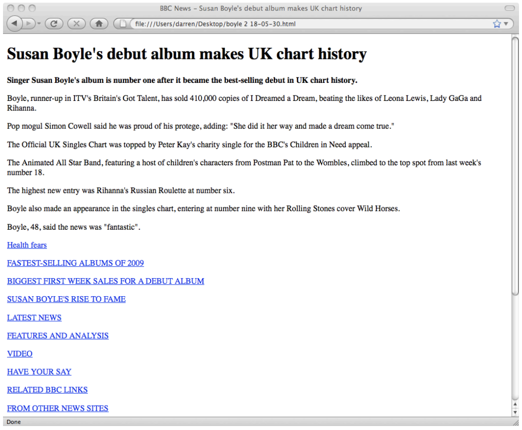
[9] URL: `http://news.bbc.co.uk/1/hi/entertainment/8385395.stm`. Last accessed: 30th November 2009

Chunking involves decomposing a Web page into a series of smaller pages or chunks that can be more easily accessed on small screen devices. Digester, developed by Bickmore *et al.* [1999], analysed a page and split the contents into smaller components that were accessed via an automatically generated table of contents page. For example, a Web page that had three headings with paragraphs of text underneath would be broken into three pages. Each page would consist of the heading and the text. To access the three new pages, a fourth page would be created consisting of a table of contents with links to the three pages. The idea behind the transformation was that mobile devices have small screens and therefore the user can only view a small portion of the page. Having a table of contents allowed users to easily access the part of the page that they required without the difficulty of scrolling for long periods of time. An example of chunking can be seen in Figure 2.4b. The first part of the news story has been left on the page, but subsequent headings have been converted into hyperlinks that point to pages that contain the content that was underneath those headings. This reduces the amount of content to be displayed on the screen, which can be beneficial when users are interacting with small screen devices.

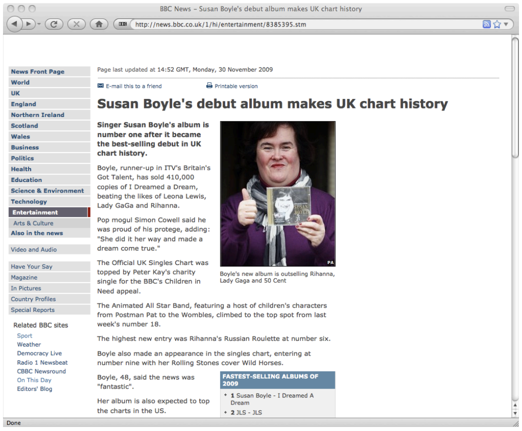
Filtering does not restructure a page like other transcoding methods, but merely removes content from the page that is deemed unnecessary. WebFilter, developed by Boldt [2000], was a tool that created filtering templates written in Perl. The Perl scripts looked for patterns in the HTML syntax that represented elements that the user did not want to see. For example advertisements could be removed by looking for the element within the document that inserted the advertisement into the page. Filtering out content reduced the download time of pages on narrow bandwidth connections and also allowed users to access content without the distraction of unnecessary clutter. An example of Filtering can be seen in Figure 2.4c. Here the masthead at the top of the page and the related stories column located on the right-hand-side of the page have been removed.

Reordering involves rearranging a Web page so that areas of the page that are considered to have more important information are placed in a position that is easier to access. Asakawa & Takagi [2000] used reordering as a means to transcode Web content using the Annotation-Based Transcoder. Fragments of the page were annotated with an importance value on a scale of 1.0 to -1.0. The transcoder rearranged the page so that fragments of importance level 1.0 were placed at the top of the page and fragments of importance level -1.0 were placed at the bottom.

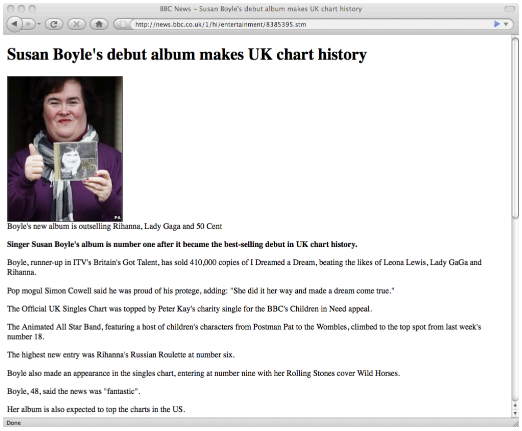
(a) An unmodified news story from the BBC Website.



(b) Applying chunking to a news story from the BBC Website.

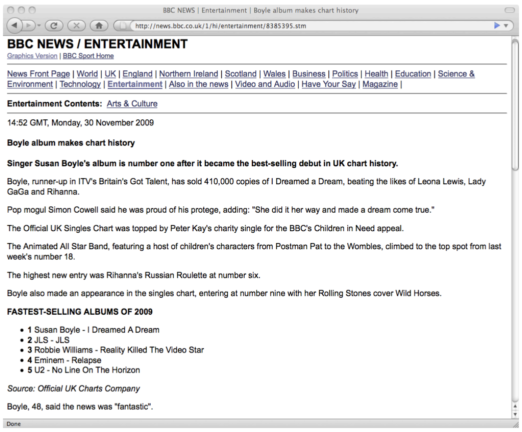


(c) Applying filtering to a news story from the BBC Website.



(d) Applying reordering to a news story from the BBC Website.

(e) Applying scaling to a news story from the BBC Website.



(f) Applying text-only to a news story from the BBC Website.

Figure 2.4: Comparison of different approaches to transcoding Web content.

This allowed users to have direct access to the main information contained within the page without having to navigate through additional content that they may not be interested in. An example of reordering can be seen in Figure 2.4d. Here the news story, which can be considered as the most important information on the page, has been moved so that it is located at the top of the page. When a screen reader user accesses the page, they are immediately presented with the main story, rather than having to navigate through the masthead and menu of the page first.

Scaling attempts to reproduce the desktop version of the Web page on an alternative small screen device. Gutwin & Fedak [2004] use four image transformation techniques applied to the page as a whole in order to allow mobile device users to have access to content designed for a large screen display. These techniques, including panning, zooming and fisheye, allowed users to see the page as a whole, albeit on a smaller scale, and then focus in on areas that were of interest. An example of scaling can be seen in Figure 2.4e. A scaled down version of the page is displayed to the user. However, as the user moves the mouse pointer over the content, the area directly below the cursor is magnified allowing users to read the content more easily, yet still have an overview of the page.

Text only transcoding results in a text only version of the Web page. Betsie, developed especially for the British Broadcasting Corporation (BBC) Website, removed all the images and Java from the page and replaced them with the value of the `alt` attribute of the tag if one was available [BBC 1999]. If there was audio on the page, then a link was inserted that pointed to the audio file so that users could access it if they wished to do so. An example of text-only transcoding can be seen in Figure 2.4f. The images and multimedia elements of the page have been removed. Columns and tables have also been collapsed so that users only interact with a linear version of the content.

2.4.4 Relationship To Our Research

The transcoding techniques discussed have been shown to improve access to Web content for users [Maeda *et al.* 2003; Yesilada *et al.* 2007b]. The techniques that employ heuristics can be applied to a wide variety of Websites without any additional information with regards to the page content [Asakawa & Takagi 2000]. While manipulations of the structure can aid navigation and display, they have two problems. The first is that the rules have to be general enough so as to be applicable to every possible Web page. Therefore they do not capture and exploit the semantic information of the page, which is usually only available implicitly through the visual presentation. Secondly, a specific user group is targeted, such as mobile Web users or visually impaired users. This means that for each user group or target device, a different set of heuristics and templates need to be discovered and applied to the page [Huang & Sundaresan 2000].

In contrast, semantic transcoding attempts to capture the meaning of the information on the page in order for this meaning to be exploited to suit a variety of different user groups [Asakawa & Takagi 2000]. This involves deeper levels of document understanding and therefore human intervention into machine understanding of documents is required [Nagao 2000]. Annotations can provide the additional information that machines use to perform semantic transcoding. This tends to produce higher quality transcoded documents because of the additional level of understanding. However, the cost of this is that annotations have to be created by hand, which can be extremely tedious and time consuming [Takagi *et al.* 2002].

As discussed in previous sections, the rendering of a Web page element provides additional implicit meaning that informs the user of its role within the Web page. This rendering information is usually defined once and any elements within the Website that have similar roles are associated with that rendering. We have also seen how we can exploit the association between the XHTML and the CSS to provide a mechanism for annotating the content of the page. Such annotations allow for a hybrid transcoding solution. The annotations allow us to accurately describe the role that each element on the Web page plays. This is akin to semantic transcoding where annotations are used to provide accurate adaptations. However, the annotation will only occur in one location, at the CSS level. The annotations will then propagate through to all the XHTML documents due to their association with the CSS. This provides the benefit of heuristic transcoding whereby a large number of pages can be adapted without the need for modifying or annotating every document.

There is also scope to improve the transcoding that is provided to the user. The transcoding techniques described above have been based on developers using intuition or previous works to solve perceived problems that users face when accessing Web content. These transformations are often based on identifying the structural elements of the content and then reengineering that structure to suit the linear output produced by screen readers. Most of the transformations provided are then justified through user evaluation and testing. While the Annotation-Based Transcoder [Asakawa & Takagi 2000] and Web Access Gateway [Brown & Robinson 2001] involved visually impaired users, it should be noted that these were expert users leading the development effort. To move towards behaviour-driven content adaptation that will further enhance access to Web content, an understanding of how users interact and cope with the Web is required.

2.5 Understanding User Behaviour

Understanding user behaviour within a Web environment revolves around usability studies and user evaluations that focus on how well users can interact with a given Website [Nielsen 2004b]. When understanding user behaviour, user studies can capture requirements and document existing problems of users performing a task; provide useful information on the design process; or gauge how well a user can achieve their goal [Jay *et al.* 2008]. Such studies provide insights into how users interact with computer systems. These insights can then be used to develop interfaces that support users as they complete tasks and which users feel comfortable using.

2.5.1 User Behaviour

Within psychology there are multiple definitions of what constitutes behaviour and it is beyond the scope of this work to provide a definitive explanation. However, one can abstract behaviour to an action performed by a person in response to a stimulus from the external environment [Chein 1972]. These behaviours can fall into two broad camps. The first is involuntary, reflexive behaviour, which can be considered to be purely physical and mechanical reactions to a stimulus that a person has little control over [Nelsson & Hayes 1986]. For example, shining a light into the pupil of an eye causes the pupil to contract. The person has very little control over the behavioural response, although there is some conditioning that can be applied [Hutt *et al.* 1966]. The seminal work on conditioning reflexive behaviour was conducted by Pavlov [McKellar 1989]. Pavlov used a variety of stimuli, including bells, to indicate to animals that it was time for feeding. This conditioned the animals to drool on receiving the stimuli in anticipation for being fed, even when food was not present [Pavlov 1927].

Voluntary, or complex behaviours, are attempts to achieve some goal that is influenced by motivation, emotional state and prior experiences [Hutt *et al.* 1966]. To achieve a goal, motivation is used to channel behaviour into specific tasks [Berelson & Steiner 1964]. Motivation to achieve goals can be physiological, such as the desire for food in order for the organism to survive or psychogenic, such as the desire to complete a university degree [Barrett *et al.* 1986].

The focus on understanding human behaviour on the Web lies in complex behaviours whereby a user has a goal that they wish to achieve. This may be high level, such as the desire to read email online, or more concrete tasks such as selecting a link with the mouse pointer. Understanding user behaviour when interacting with computer systems was first introduced in the early 1980s. The assumption made by developers was that evaluating how users interacted and used a computer system would demonstrate that the system was usable and so occurred at the end of the development cycle [Dumas & Redish 1999]. This approach is taken with modern day Websites with methods used to understand how users behave and interact with online content falling into two broad categories. The first is quantitative evaluations that involve the acquisition and analysis of measurable data that is obtained from the user as they interact with a Web page. As the results of quantitative evaluations are numeric, the results can generally be analysed using statistical techniques [Dix *et al.* 1993], allowing complex usability issues to be expressed as a number that can easily be compared and discussed [Nielsen 2004a].

Qualitative studies try to elicit users' implicit knowledge about, and perception of, the evaluated Website [Jay *et al.* 2008]. When visiting a Website, each user has a specific goal based on their needs and experience that influences the problems that they have while interacting with a Web page. Qualitative techniques help the evaluator to understand the individual problems, reactions, behaviour and expectations of the user.

These techniques usually occur after development has taken place and so only provide insight into how users behave with a given Website. To inform the design of interfaces before development begins, models of human cognitive behaviour have been developed. One of the first works to model human behaviour for computer systems was Goals, Operators, Methods, and Selection Rules (GOMS).

This was a framework developed by Card *et al.* [1983] that modelled how users achieved their goals in a given system. The four aspects of the method were defined as *Goals*, which are the tasks that the user wants to achieve; *Operators*, which are the actions that could be performed to achieve a goal; *Methods*, which are sequences of operators that accomplish a goal; and *Selection Rules*, which involve choosing what method to use when there are several choices for achieving the same goal [John & Kieras 1996]. The advantages of using such models to predict behaviour is that they can estimate how long a task will take to complete. They can also highlight areas where tasks may be completed with few Web pages, but are heavy with content and difficult to interact with, or take a large number of pages but are easier for the user to understand [Olson & Olson 1990]. However, the model assumes that users will not make any errors during the completion of their tasks. How a Website copes if a user deviates from this rigid execution of operations is not clear, which can adversely effect the site performance, especially when novice users are performing tasks.

The problem of task deviation is a particular problem when studying how visually impaired users interact with Web content. Many Websites still have accessibility issues, even when guidelines have been conformed to as designers often rigidly follow the guidelines without understanding the importance of the accessibility principles that the guidelines are composed of [Asakawa 2005]. Studies have shown that to access Web content, visually impaired users develop their own familiar scanning navigation methods. These methods are adhered to by the user for each Website that they visit rather than try and understand the logical organisation of each Website's content [Takagi *et al.* 2007]. Therefore developing a GOMS model would not be suitable as each user would require a different set of *Operators*, *Methods*, and *Selection Rules*, which they may deviate from if accessing the content became difficult.

User evaluations and GOMS models focus on how users achieve specific goals. More recent studies have focused not on goals, but on how users generally behave when interacting with Websites [Jay *et al.* 2008]. This is usually achieved through the use of eye-tracking, whereby users' eye movements can be captured and analysed to understand how users perceive the Web page or application interface. This analysis can provide key insights into user behaviour that can aid future design decisions of Websites and applications by allowing developers to understand how users really interact with their system and create presentation and interaction methods that support user behaviour. Work by Jay *et al.* [2006], for example, demonstrated that the layout and presentation of a Web page can assist users as they interact with the Website. The study tracked users' eye movements as they interacted with the text-only version of the BBC News Website[10] and the standard graphical version. Users made significantly more eye fixations on the text-only version suggesting that the text-only page required more of the user's attention to process the content. Therefore it is not adequate enough to present visually impaired users with a text equivalent of a Website. The layout and design guides users through the content and makes the Website easier to interact with. Therefore content needs to be presented in audio in such a way that users can easily reach the content and interact with the Web page.

2.5.2 User Coping

Within psychology, coping is defined as a *"constantly changing cognitive and behaviour effort to manage specific external and/or internal demands that are appraised as taxing or exceeding the resources of the person"* [Lazarus & Folkman 1984]. Coping can be viewed as an adaptive process whereby changes are made to meet the current threats that the person is facing [Lazarus 1993]. Lazarus [1966] described this adaptive process as a mechanism that consisted of three parts. The first was Primary Appraisal, whereby a person decided how much danger they were in due to the current situation. Secondary Appraisal was an analysis of how much the danger could be reduced from a given action. The final third phase was the application of the coping strategy itself to the threat that the person was facing.

In an effort to understand the coping process Folkman & Lazarus [1985] analysed how students coped with the pressures of examinations through the use of questionnaires. The coping processes were split into two groups. The first was Problem-Focused Coping Strategies whereby the student tried to alter the cause of the stressful event. Answers for this group included *"I'm making a plan of action and following it."* The second group was Emotion-Focused Coping Strategies whereby the student attempted to reduce the emotional stress caused by the situation. Answers for this group included *"I wish that the situation would go away and be over with."* The study concluded that participants rarely relied on a

[10]URL: `http://news.bbc.co.uk/`. Last accessed: 7$^{\text{th}}$ September 2008

single form of coping and instead used combinations of the two strategies to deal with the situation.

Carver *et al.* [1989] argued that while a distinction between problem- and emotion-focused coping was important, it was too simplistic to capture how people cope with situations. To support this a multidimensional coping inventory consisting of fourteen elements was devised. The elements ranged from actively coping with the situation through to alcohol and drug disengagement. The study asked 978 participants the ways in which they dealt with a variety of situations, and a ranking of the various strategies was created. Planning and active coping were the most common forms of strategy, however there was a significant number of participants who also sought social support when faced with a problem. It was interesting to note that there were several distinctions between gender, especially with coping strategies that tended to be more emotional-focused.

While both Folkman & Lazarus [1985] and Carver *et al.* [1989] identified ways in which people cope, they tended to be generic and applicable to all situations that can be deemed stressful. From the perspective of Web accessibility, we define coping strategies to be *routines that users employ when interacting with inaccessible Web content*. One of the first observations of this phenomena was by Yesilada *et al.* [2007b] in an evaluation of the DANTE transcoder. During the study, users developed a number of strategies, described as reading patterns, that were employed while reading the content on the page. For example, to skip the Google banner and avoid listening to the advertisements and other elements that surrounded the results, some users counted the number of `tab` key presses to access the content of interest more quickly.

While this is a well known example, there has been little research conducted into coping strategies from a Web accessibility or computer science viewpoint, although preliminary investigations have recently emerged. Using a proxy to record the Web pages that users visited and the keystrokes that they made, Bigham *et al.* [2007] attempted to determine what coping strategies visually impaired users employed compared to sighted users. The study concluded that visually impaired users probed pages more than sighted users and that visually impaired users occasionally used a mouse to access inaccessible page elements. However, the study had a number of weaknesses, in particular, the remote logging of the data without any follow up studies. Capturing data logs is a powerful technique when studying human computer interaction as it allows for the automatic acquisition of large amounts of data, from multiple users, over a long period of time [Nielsen 2004b]. However, log data analysis has difficulties of mapping low level system usage to high level user tasks [Ivory & Hearst 2001]. Evaluators can understand *what* users did when interacting with a Website, but cannot necessarily understand *why* they did it, and if they achieved their goal in a satisfactory manner [Nielsen 2004b]. Therefore the results of Bigham *et al.* [2007] provided information that probing occurred, but did not convey why users relied upon probing and why probing was necessary to complete a task.

In contrast to the wide ranging, remote study of Bigham *et al.* [2007], Shinohara & Tenenberg [2007] observed a single visually impaired user in their home for a total of twelve hours, spread out over six, two-hour sessions. The aim of the study was to understand how technology can be difficult to use for visually impaired users and the workarounds users employ when difficulties arise. It should be noted that Shinohara & Tenenberg [2007] used the term "breakdown" to indicate a failure to complete a task due to the design of technology and "workaround" as a strategy to overcome a breakdown. These are analogous to the author's use of the terms "inaccessible" and "coping strategy". An interesting coping strategy that was reported when using JAWS was that the user would often go back to a start page whenever they became lost within a Website. This allowed them to start again and retrace their steps in an attempt to find the content that was required. While Shinohara & Tenenberg [2007] provided interesting insights into a visually impaired user's Web browsing behaviour, the study itself was broad ranging. The participant was asked to demonstrate any technology that they wished, and so items such as braille printers and talking watches were also discussed. This created a study that was more akin to an ethnographic survey of a user and therefore lacked the focus we require, which is the coping strategies used when accessing content on Web pages.

2.5.3 Relationship To Our Research

There have been a number of techniques used to understand how users interact with Web content. One of the earliest methods was Goals, Operators, Methods, and Selection Rules (GOMS). This was a framework developed by Card *et al.* [1983] that models how users achieved their goals in a given system. The advantages of using such models to predict behaviour is that they can estimate how long a task will take to complete. However, the model assumes that users will not make any errors during the completion of their tasks. How a Website copes if a user deviates from this rigid execution of operations is not clear. This can be a particular problem for visually impaired users as studies have shown that to access Web content, visually impaired users develop their own scanning and navigation methods which they adhere to for each Website rather than try and understand the logical organisation of the Website content [Takagi *et al.* 2007].

Rather than predicting how users can achieve their goals on a Website, current studies have focused on how users generally behave when interacting with Websites [Jay *et al.* 2008]. In particular, recent work has focused on understanding how users come when access to Web content becomes difficult. One of the first observations of this phenomena was by Yesilada *et al.* [2007b] in an evaluation of the DANTE transcoder. Since then there have been a small number of studies with a small number of users that have looked into identifying coping strategies, however the results so far seem promising. Bigham *et al.* [2007] made use of logs to capture a large amount of data but there lacked an understanding of what users were trying to achieve and why they performed particular actions to achieve that goal. While this is capably if capturing a large amount of data, it can only identify *what* users did when interacting with a Website, but cannot necessarily understand *why* they did it, and if they achieved their goal in a satisfactory manner. In contrast, the work by Shinohara & Tenenberg [2007] has tried to understand at an in-depth level why users performed particular tasks, but this work has been limited to a single user and covered a wide manner of daily tasks. This study, which was similar to an ethnographic study, provided insights into how users coped when Web content and other technologies became challenging. There is therefore scope for a similar but more in-depth study with multiple users that focuses on Web usage. Such a study would enable us to develop a deeper understanding of the difficulties that users face when interacting with Websites and the coping strategies that they develop in an attempt to overcome those difficulties. This would then allow us to identify transcoding techniques that complement how users interact with the Web and provide transformations that further enhance access to the Web for visually impaired users.

2.6 Summary and Conclusions

In this chapter we have seen how HTML, the original language of the Web, conveyed the structure of the information within the document, not its presentation. The markup available for HTML documents was designed to create a structured document that could be displayed on any number of devices used to access the document and could be considered to be accessible. As the Web evolved, developers wanted more control over the presentation of their Web content and so presentational extensions were added to the HTML specification to the detriment of device independence and accessibility. However, there is a move towards a purely structural HTML specification, in the form of XHTML. This, in conjunction with the CSS style language, has allowed for the separation of presentation and structure. This not only benefits Web designers in the maintenance of their Websites, but also enhances access to the content for a variety of devices. The advent of annotation mechanisms, such as RDF, Microformats and OWL, have also facilitated the machine understanding of Web content and allowed a number of devices to consume data on the Web and process it automatically. When fully realised, the semantics added to Web content will be of great use to the field of accessibility as it will allow tools to identify and convey content to users in a manner that better suits their needs.

In order to promote access of Web content to all users, regardless of ability, the W3C Web Content Accessibility Guidelines have been produced. The guidelines are aimed at a multitude of disabilities, however, a vast majority of the checkpoints are concerned with people with visual impairments. While WCAG has helped to raise the awareness and profile of Web accessibility they are not sufficient to provide Web pages that are accessible. This has led to research into transcoding, whereby Web content is adapted into a format that is suitable for alternative access devices such as mobile phones and assistive technologies, such as screen readers. While a number of approaches have been taken to transcode Websites, broadly speaking the proposed solutions are either highly scalable to the detriment of accuracy, or highly accurate but lack scalability. Furthermore, the adaptations that are provided are structure-driven and based upon designers reformatting the structure of the page into one that is suitable for the linear audio output of a screen reader.

During user evaluations of transcoding devices, coping strategies have been observed. These are techniques that users employ in order to cope with the difficulties of accessing the Web. While coping is a well known phenomena in the field of psychology, there has been little investigation into understanding the phenomena within Web accessibility. While some studies have tried to identify coping strategies, they have been limited in their approach. This therefore poses the question as to what are coping strategies and can they be used as a means to enhance transcoding functionality and Web accessibility.

Chapter 3

The SADIe Experimental Prototype

Structural-Semantics for Accessibility and Device Independence (SADIe) is an approach to transcoding that combines the benefits of both heuristic and semantic transcoding first proposed by Harper & Bechhofer [2004] but never developed or evaluated. This chapter describes the work conducted to construct a SADIe experimental prototype in order to support Research Question 1. Implementing a prototype of SADIe as a transcoding proxy brought to light issues surrounding a two-part ontology approach used to annotate the roles of the elements found within the page. Addressing the problems associated with the ontology led to the development of an annotation method that extended the properties of the Cascading Style Sheet.

To evaluate the technical feasibility of the SADIe transcoding method, established structure-driven adaptations were incorporated into the transcoding engine. This allowed the outcome of the SADIe adaptations to be compared against expected results. Forty Websites were transcoded with the SADIe transcoder. Of the forty Websites, SADIe was able to successfully adapt 89%, with a true range lying between 99% and 79%. In this study, success was determined by the output of the transcoding being consistent with the predicted output of the established transcoding techniques. This demonstrated that SADIe was capable of transcoding a diverse range of Websites and was suitable for use as a basis for investigating structure-driven and behaviour-driven transcoding algorithms.

The advantage of constructing the SADIe transcoder was that it provided a mechanism for making consistent comparisons between structure-driven and behaviour-driven transcoding algorithms. When conducting user studies with both the structure-driven adaptations and the behaviour-driven adaptations, described in Section 5, any changes in behaviour while accessing Web content by the participant would be attributable to the transcoding itself. All other factors, including the annotation model and the underlying transcoding engine, would remain constant and therefore provide more robust results.

3.1 SADIe Transcoding Approach

Transcoders seek to provide adaptations to Web content that are both scalable and accurate. In this context, scalable can be defined as the ability to adapt a large number of Web pages with a minimal amount of human intervention. Accuracy can be defined as the ability to consistently apply transformations to a set of Web pages in a systematic and predictable manner. Previous approaches to transcoding, as discussed in Section 2.4.2, have broadly fallen into two categories. The first, known as heuristic transcoders, use techniques that are highly scalable so that a large number of Web pages can be adapted easily. However, in order to accommodate the large number of Websites, accuracy suffers leading to transformations that do not always produce the desired results [Asakawa & Takagi 2000]. Alternatively, semantic transcoders have been proposed that have high levels of transcoding accuracy. These use techniques that typically require Web pages to be annotated before any transformation can take place. Due to the workload placed on the human when adding annotations to Web pages, these systems have lacked the scalability to be applied to a large number of pages [Takagi *et al.* 2002].

SADIe is an approach to transcoding that combines the benefits of heuristic and semantic transcoding first proposed by Harper & Bechhofer [2004] but never developed or evaluated. The principle idea behind the approach is that the rendering of a Web page element is closely associated with its role. As discussed previously in Section 2.1.3, rendering information can be defined within the Cascading Style Sheet (CSS) and associated with the HTML via tag attributes such as `class` or `id`. The rendering information, which controls the colour, layout and presentation of Web page elements, can help distinguish the role that the content on the page plays. For example, allowing users to identify a list of links as the main navigation menu of the Website.

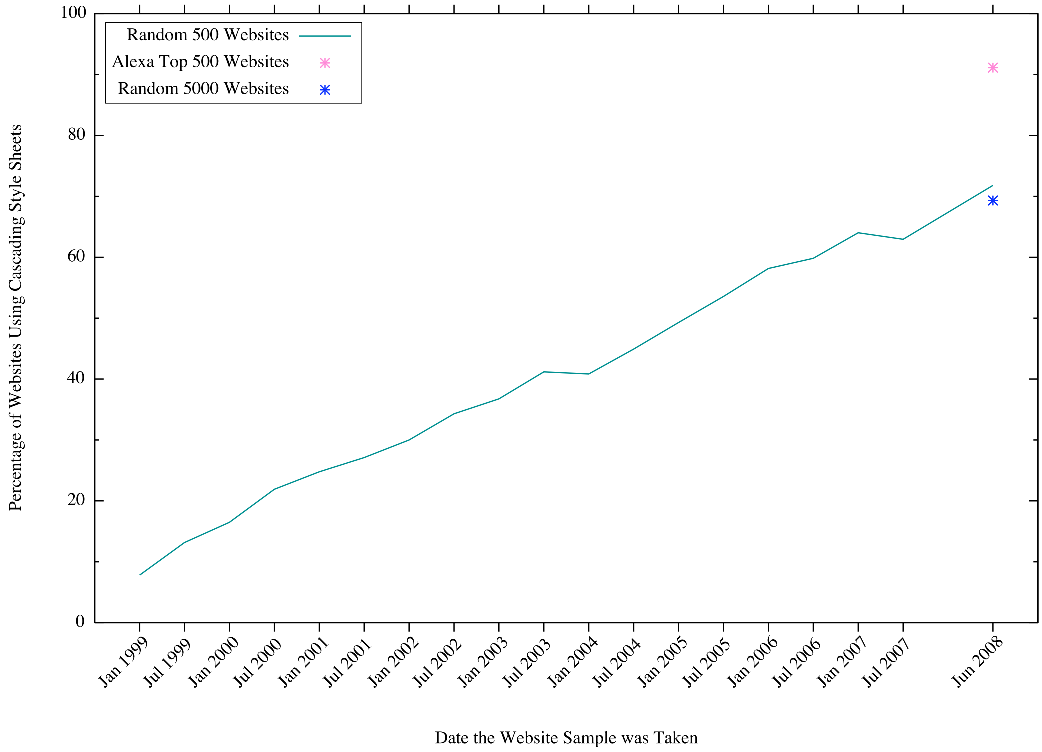


Figure 3.1: A graph showing the percentage of Websites that use Cascading Style Sheets from samples taken every six months. Data reproduced from [Chen 2008].

A Website typically has a homogenous cascade of styles containing all the style definitions that all pages within a Website refer to in order to maintain a consistent look and feel. The SADIe approach is to therefore annotate the CSS classes themselves rather than annotate every page contained within the Website. This reduces the annotation overhead as, rather than annotate every page within the Website, annotation occurs in one location at the CSS level. As the pages of the Website are associated with site-wide style definitions of the CSS elements, they not only inherit the rendering information but also the annotations that have been associated with the CSS. The CSS annotations allow semantic transcoding to take place as the role of each element is explicitly stated, improving the accuracy of the adaptations. In addition, scalability is achieved as defining the roles of CSS elements allows every page within the Website to be transcoded due to CSS containing site-wide style definitions that all pages use.

A limitation of the approach is that the visual rendering of a Website must be based on CSS. Legacy Websites that combine presentation with structure

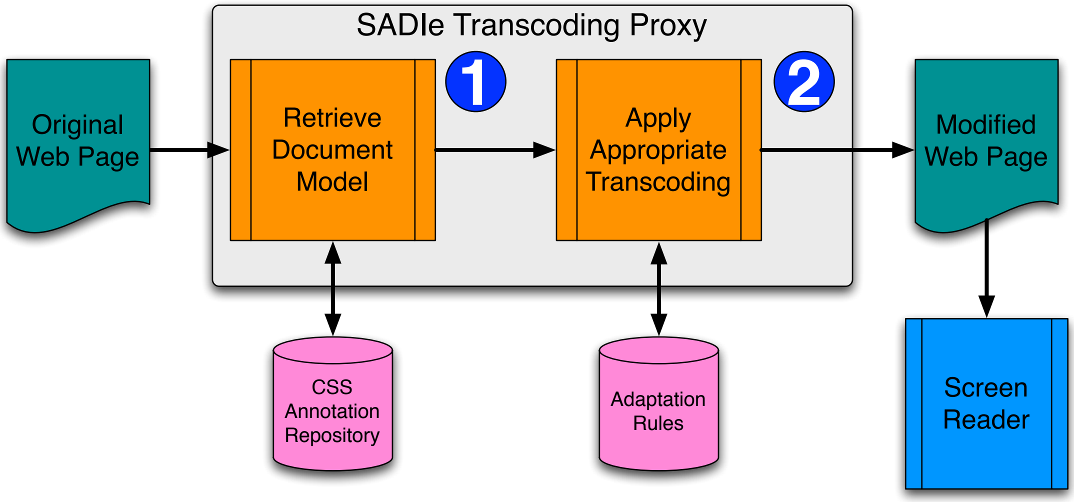


Figure 3.2: The experimental SADIe transcoding proxy architecture. Given a Web page, the proxy retrieves the appropriate document annotations and applies suitable transcoding before returning the modified Web page to the user.

are therefore not transcodable using this annotation method. However, recent investigations into the utilisation of W3C recommendations have shown that the adoption of Cascading Style Sheets is increasing [Chen 2008]. Figure 3.1 on page 67 shows the percentage of Websites using CSS based rendering at six month intervals from January 1999. As of June 2008, 72% of 500 random Websites made use of CSS rendering. In June 2008 two additional samples of Websites were taken. The first was the top 500 visited Websites as ranked by Alexa[1], a Web information and rankings Website. From this sample, 91% of the Websites tested made use of CSS. The second was a sample of 5000 random Websites showing that 69% of Websites made use of CSS for rendering. This shows a positive trend towards the use of CSS rendering and therefore a significant number of Websites that can potentially make use of CSS annotations and be open to transcoding using the SADIe method. While there still remains a number of non-CSS based Websites, we assert that over time these Websites will be refreshed and employ CSS as a presentation method, or the Websites will be left to fade and the URL expire.

The transcoding output of SADIe was independent of the annotation of the Cascading Style Sheets. Figure 3.2 on page 68 shows the architecture of the SADIe transcoder. Note how it is a proxy architecture similar to those that

[1] URL: `http://www.alexa.com/`. Last accessed: 7th September 2008

were discussed in Section 2.4.1 and shown in Figure 2.3 on page 49. The first module, indicated as ①, was developed in order to allow the transcoding engine to match the Web page URL with the appropriate CSS annotations stored in a central repository. As will be discussed in Section 3.2, the annotation model changed during testing due to issues that arose with the original two-part ontology solution. However, the modular approach allowed this to occur without adversely effecting the transcoder output.

The second module, indicated as ②, was developed to allow for the adaptations to be easily changed. One of the primary goals of building the SADIe prototype was to allow a fair comparison of structure-driven and behaviour-driven transcoding to take place. This could be achieved by changing the adaptation rules in the repository. The remainder of the platform remained constant, with the annotations, interface and transcoding engine remaining the same.

3.2 Annotating CSS Elements

The aim of the annotating the CSS is to explicitly state the role that elements on the page are playing in order to allow the transcoder to apply suitable transformations to those elements. Consider the CNN news story[2] shown in Figure 3.3 on page 70, which is used as a running example to show how both annotation approaches and transformations facilitate users in easily accessing the content of the Websites. The most important content of the page can be considered to be the story headline (indicated as ①), the highlights of the story (indicated as ②) and the article itself (indicated as ③). However these important elements are surrounded by elements that hinder the user such as a banner (indicated as ④) and advertisements (indicated as ⑦). The page also contains elements that are useful, such as a search box (indicated as ⑤) and the navigation menu (indicated as ⑥), but which the user does not necessarily need to access with every visit to the page. The designer of the page has used a combination of colour, white space and font effects to draw a sighted user's attention to the main area of the page, leaving the remaining elements on the periphery of the page where they do not interfere with the main content. However screen readers have no notion of visual rendering and so the clutter that surrounds the main article hinders the user as

[2]URL: `http://www.cnn.com/2008/WORLD/weather/09/07/hurricane.ike.carib/`. Last accessed: 7th September 2008

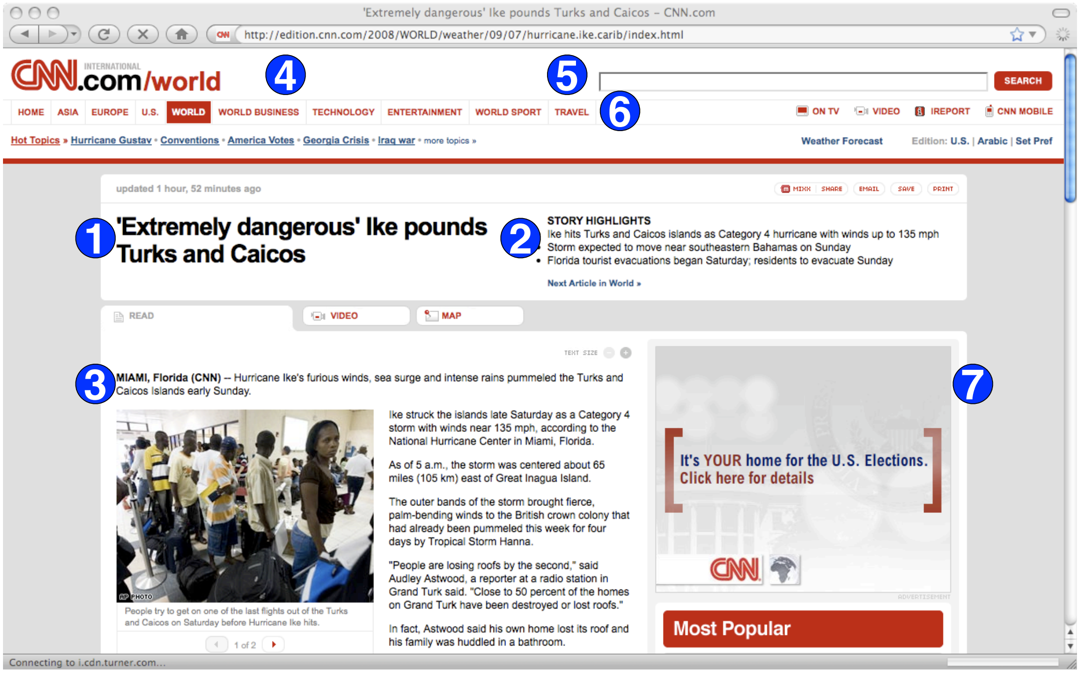


Figure 3.3: A news story from The CNN Website displayed in a visual manner as the designer intended. The main content of the page (title, key points and the article itself) are predominant and easily identified by the user.

they try and reach the content. The annotations expose these elements to the transcoder in order for the page content to be adapted into a format that is more accessible for the user.

During development of the prototype two approaches to annotating the role of the elements contained within the Web page were investigated. The first involved a two-part ontology that contained elements found within the CSS in the lower ontology and abstract role concepts from an upper ontology. During usage, weaknesses within the annotation method became apparent therefore a second approach was investigated. This approach involved extending the Cascading Style Sheet properties. This resulted in increased flexibility of the annotation approach and reduced the overhead of annotating Websites.

3.2.1 Annotation Via a Two-part Ontology

One approach to annotating Web elements is through the use of a combination of an upper and lower ontology. Figure 3.4 on page 71 demonstrates the underlying architecture of how the ontology can be used to annotate the CSS elements of

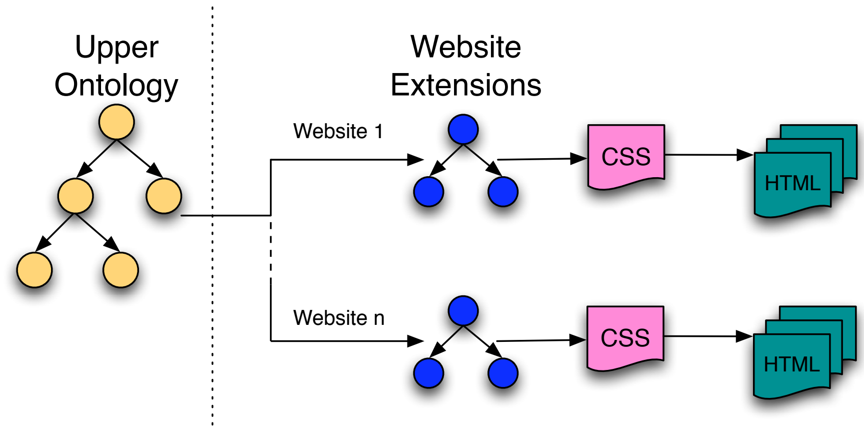


Figure 3.4: Annotation of Websites through and extension of the upper ontology for each Website that is to be transcoded. The upper level ontology contains properties and high level abstract concepts that represent the potential roles of Web page elements.

a Website. The upper level ontology contains properties and high level abstract concepts that represent the potential roles that Web page elements can play. These concepts provide a controlled vocabulary with which to classify the roles of the Websites that one might wish to transcode.

The second part of the ontology provides a Website specific extension to the upper ontology. This contains the elements found within the Cascading Style Sheet of the Website that is to be adapted. The Website ontology accesses the concepts found in the upper ontology by importing the entire upper ontology. This allows additional classes to be added in the upper ontology, which subsequently become available to all the Website extensions.

The benefit of using an upper ontology is that it acts as an interface between the application performing the transcoding and the page being transcoded. The roles of the CSS elements are defined by the abstract classes and properties of the upper ontology through a site specific extension. During the transcoding process, a transcoding application will require access to the roles that each element of the Website plays. As Websites use heterogeneous naming policies for the CSS classes, querying a site directly is difficult as there is no facility to associate the names used in each Website's CSS with the role that the elements play. By defining classes in terms of a consistent collection of concepts, an interface to a Website is provided, regardless of the names that are used to create the CSS elements. The transcoding application merely requests the names of the CSS

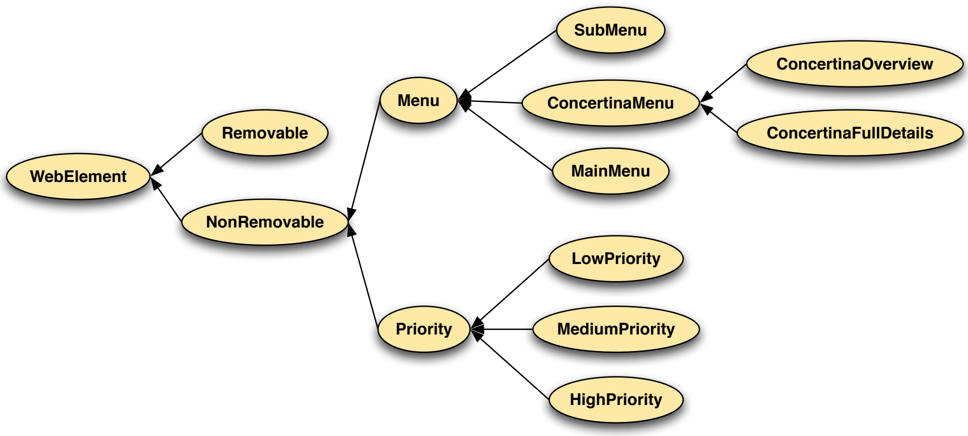


Figure 3.5: The original upper ontology used to act as a classification system of Web elements.

elements that fulfil the role type that is being manipulated, via the terms found in the upper ontology.

The upper ontology, known as the SADIe Upper Ontology, can be seen in Figure 3.5 on page 72. The implementation of the upper ontology consisted of a small number of disjoint concepts that were used to act as a classification system of Web elements. For example, a Web page element could be either `Removable` or `NonRemovable`. A `NonRemovable` element could be either a `Menu` or given a `Priority` and so on and so forth. Properties were added to a class in order to define what its role was. For example an element of high priority would have the property ∃ `hasPriority some Thirteen` added to the class. Even though SADIe used a three tier rating of High, Medium and Low priorities, elements were assigned a number between 1–15 inclusive, with 1–5 representing Low Priority items; 6–10 representing Medium Priority items; and 10–15 representing High Priority items. This was to add flexibility by having a more fine-grained measure of priority, although in practice the three tier approach was the only scale used. The properties of the Upper Ontology are provided in Table 3.1 on page 73. The complete SADIe Upper Ontology, written using Manchester Syntax [Horridge & Patel-Schneider 2009], can be found in Appendix A on page 172.

As an example of constructing an ontology for a Website, consider the CNN Web page shown previously in Figure 3.3. As has been discussed, the CSS rendering gives elements a look and feel that allows users to identify the role that that element is playing on the page. The CNN navigation menu indicated as ⑥ in Figure 3.3, for example, is an unordered list defined as `<ul`

Table 3.1: The properties used to define classes in the SADIe Upper Ontology

Property	***Range***	***Definition***
hasPriority	Integer	This allows for the priority of an element to be specified, on a scale of one to fifteen, with one being the lowest and fifteen indicating the highest priority.
isRemovable	Boolean	This allows a class to be defined as Removable.
isMainMenu	Boolean	This allows a class to be defined as Main Menu.
isSubMenu	Boolean	This allows a class to be defined as Submenu.
isConcertinaFullDetails	Boolean	This allows a class to be defined as Concertina Menu with all the information available to the user.
isConcertinaOverview	Boolean	This allows a class to be defined as Concertina Menu with only the minimum amount of information available to the user.

`class="cnnNavigation">` where `cnnNavigation` is a reference to a CSS element that defines the style for the list. Using the SADIe approach, an ontology is made to represent CNN CSS elements that imports the SADIe Upper Ontology. In the CNN ontology a class is created to represent each CSS element and suitable properties are added to the classes. For example, a class called `cnnNavigation` is created to represent the CSS element of the same name and the property ∃ `SADIe:isMainMenu some SADIe:True` is added to state that this element is the main menu of the page. The transcoder can then use this assertion when applying transformations. Every HTML tag with a property `cnnNavigation` is considered to be the main menu and suitable transformations are applied. This process is repeated for every CSS element in the Website. A complete listing of the CNN Ontology can be found in Appendix B on page 176.

3.2.2 Annotation Via a CSS Extension

During usage of the two-part ontology model, a number of weaknesses became apparent in the approach. The SADIe Upper Ontology was developed with a dual

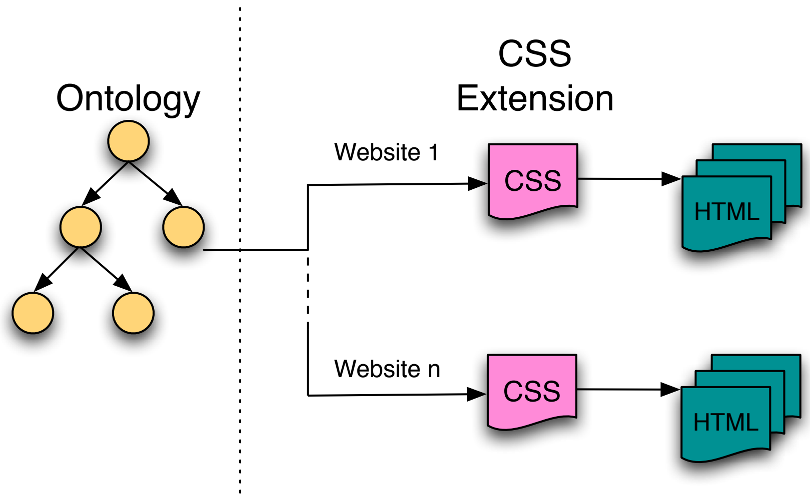


Figure 3.6: Annotation of Websites by adding terms from the ontology directly into the CSS. The ontology acts as an interface between the Website and the transcoding engine.

purpose. The main reason for its construction was to demonstrate that annotating Web documents via the CSS was a viable approach to exposing Web page semantics to transcoding applications. However, in order to demonstrate the feasibility of the approach, established structure-driven transcoding, discussed further in Section 3.3, was incorporated into the prototype tool in tandem with the annotations. As the structure-driven adaptations were based upon known techniques, the expected outcome of the transcoding was understood. This provided a method of evaluating the technical suitability of SADIe, the results of which are discussed in Section 3.4. However, as a result, the upper ontology lacked a distinction between concepts that described the semantics of the Web page and concepts that described the functionality of the structure-driven transcoding.

Consider the concept `Menu`. A menu is a property of a Web page element that describes its role and purpose. As highlighted in Figure 2.1 on page 33, it is a collection of links that allows the user to navigate through the pages contained within a Website. The concept `Removable`, however, does not reflect a semantic property of a Web page element but describes functionality to be executed on a particular element. A Web element is not inherently removable. In certain instances it can be deemed as superfluous to requirements for the user but in other situations it may have some purpose and need to be retained. The original SADIe Upper Ontology made no distinction between the two and merged both the structural semantics of the Web page and the structure-driven transcoding functionality into a single model.

Furthermore, the ontologies constructed to annotate the CSS terms were taxonomic that made no use of inference engines to discover new facts and relationships between the elements of the Website. They also took a considerable amount of effort to construct. Therefore, the annotation approach was adapted to use an alternative method that extended the keywords used within the CSS.

Figure 3.6 on page 74 demonstrates the underlying architecture of how an ontology can be used to annotate elements of a Website through an extension of the CSS. The ontology contains concepts which define roles that elements of the Web page can play. This is similar to the upper ontology shown in Figure 3.4 on page 71. However, rather than a lower ontology containing the classes that represent elements found within the CSS, this approach involves extending the keywords used within the CSS style definitions. For example, to declare that a CSS element is a menu, `-uom-structural-role: LinkMenu` can be added to the CSS definition. The value `LinkMenu` is a concept taken from the ontology to show that this CSS element is a list of links that represent a menu. When investigating annotating CSS elements through an extension of the Cascading Style Sheet, the Web Authoring for Accessibility (WAfA) ontology was used [Harper & Yesilada 2007]. This ontology contains terms to represent the meaning of content contained within a Web page and has no concepts to represent functionality. Using this ontology therefore removed one of the weaknesses of the SADIe Upper Ontology which was the combination of structural semantics and functionality into a single model. As the WAfA ontology had different terms to the ones present in the SADIe Upper Ontology, a mapping was necessary. Therefore any element classified as a `Priority` element in the SADIe ontology was annotated as a `Chunk` using WAfA and any element classified as a `Menu` was annotated as a `LinkList`.

It should be noted that there is no tight integration between the ontology and the keywords within the CSS. The terms within the ontology are merely used as a reference point for concepts that can be used to describe structure. Entering a keyword that is not part of the ontology results in the transcoding application being unable to utilise the structure of the page. However this is similar to the two-part ontology approach that used an ontology to store the names of the CSS elements. The ontology classes represented the CSS terms but there was no tight coupling between the two. Class names that had no equivalent CSS element name were ignored by the application as no instance could be found within the HTML document providing a graceful degradation of performance. Rather than attempt to apply transcoding with potentially unexpected consequences, the tool did not apply transcoding leaving the page in its original state.

By extending the CSS terms, we gain the benefits of the two-part ontology solution. That is, the ontology acts as an interface between the application performing the transcoding and the page being transcoded. The roles of the CSS elements are defined by the abstract classes and properties of the ontology through an extension of the CSS. However, we also gain a method of modelling Web page structure that requires less overhead. Rather than create an ontology and add to that classes representing the CSS terms found in the Website style sheet, during design time, a single `property: value` attribute can be added to the CSS terms. This creates a more pragmatic approach to annotation that requires less time and effort for the annotator. In addition, the validity of the style sheet is still maintained as the CSS specification allows for vendor-specific extensions to be added [Bos *et al.* 2007]. These extensions are ignored by user agents that cannot interpret the keyword and therefore cannot apply the appropriate rendering to the HTML element.

Again consider the CNN page shown in Figure 3.3. Using the CSS extension approach, an additional cascading style sheet is created for the Website that contains the CSS elements that we wish to annotate. To annotate the element `<ul class="cnnNavigation">` a new CSS class is created as `ul.cnnNavigation{ -uom-structural-role: LinkList; }`. When the transcoder parses the Web page to identify elements, it takes the CSS properties for each element and looks for the `-uom-structural-role` property. The value of this property informs the transcoder what the role of the element is and suitable transcoding an be applied. A complete extended Cascading Style Sheet for the CNN page can be found in Appendix C on page 180. Note how we only define the role properties in this style sheet. The rendering information remains in the style sheets originally associated with the page. The cascading property of style sheets is utilised so that the user agent combines all the styles sheets to create a complete set of properties for each element. Also note how less classes have been annotated using this method. As the WAfA ontology contains no functionality, only properties, there are fewer elements that require annotation as we can not specify what is `removable`.

A limitation of extending the CSS terms, however, is that elements can only be defined using a single `property: value`. Furthermore all possible values must be defined beforehand in order for tools to be able to interpret the values in a suitable manner. The ontological approach allows for multiple properties to be added to a CSS element in order to accurately define the role it plays on the Website. Extending the upper ontology also allows for additional properties and classes to be defined if they are required for describing the role of the CSS element. It is not clear if this flexibility would be of use in practice as large scale, in the wild testing of either approach has not taken place.

3.3 Adapting Web Page Elements

During development of the prototype two approaches to adapting elements contained within the Web page were investigated. The first involved established structure-driven transcoding techniques that involve reengineering the structure of the content to suit the requirements of the target user device. The second involved using the AxsJAX framework, discussed previously in Section 2.3.3, to inject Accessible Rich Internet Application (ARIA) statements into Web documents. This allowed users to interact with Web content in a manner similar to an application, as opposed to reading the content as a static document.

3.3.1 Established Structure-Driven Adaptations

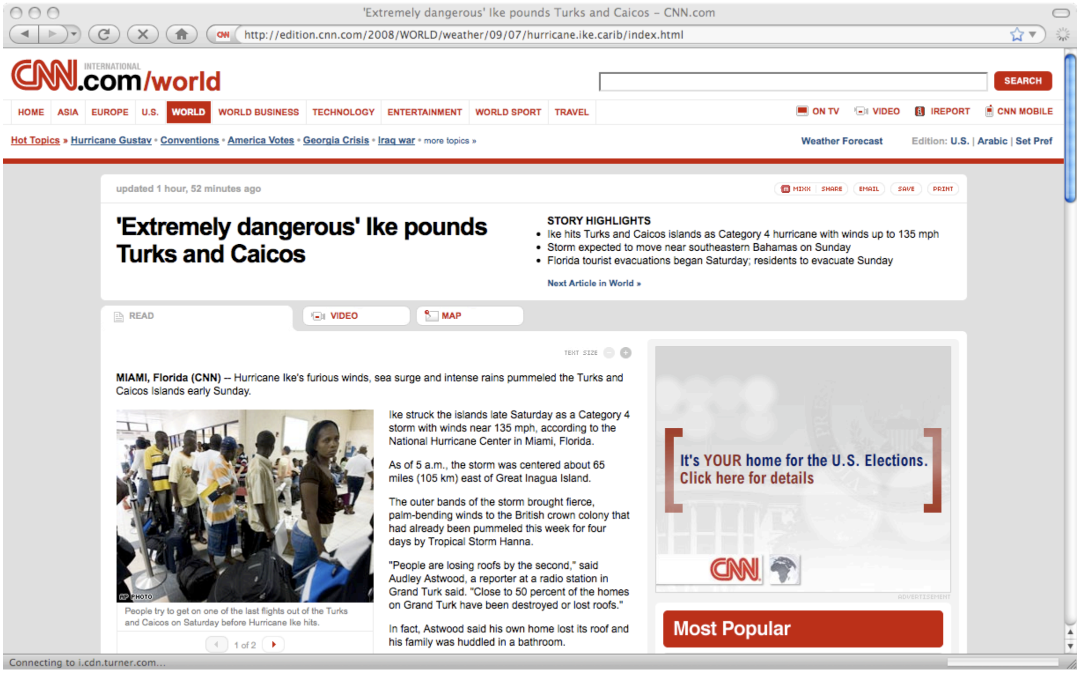
As we have seen in Section 2.4.3, a majority of the existing transcoding techniques are often based on identifying the structural elements of the content and then reengineering that structure to suit the linear output produced by screen readers. During development three existing structure-driven adaptations were employed that had been used in previous transcoding tools targeted at visually impaired users. The first was *Defluff*. In a similar fashion to the system developed by Gupta *et al.* [2003] and Web Access Gateway developed by Brown & Robinson [2001], this removed elements from the page that provided little or no information to the user and hindered their access to the content. Examples of content that could be considered as removable included footers, advertisements and parts of the banner that gave the Website its look and feel but did not add any additional information to the actual content of the page.

The second transcode that was added to SADIe was *Reorder*. This was similar to functionality available in the Transcoding Proxy developed by Takagi & Asakawa [2000]. Fragments of the page were annotated with an importance level.

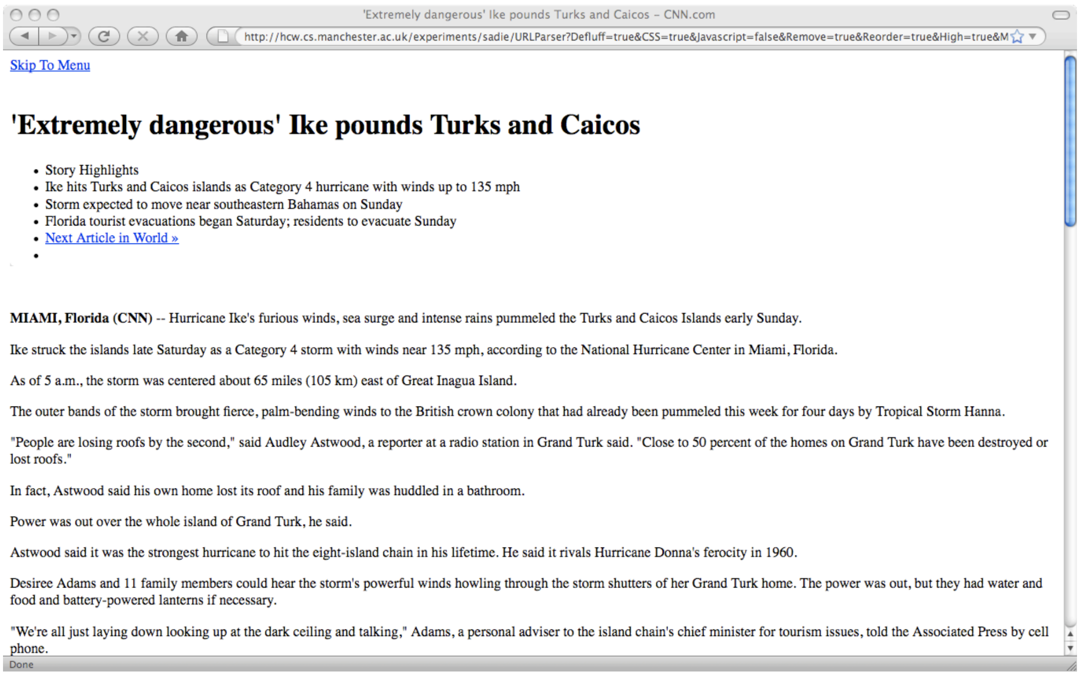
Takagi & Asakawa [2000] used a numerical scale between 1.0 and -1.0 inclusive but in the SADIe implementation a three tiered system of High, Medium and Low was used. SADIe rearranged the page so that fragments of high importance were placed at the top of the page and fragments of low importance were placed at the bottom. This allowed users to have direct access to the main information contained within the page without having to navigate through additional content that they may not be immediately interested in.

The final adaptation method to be incorporated into SADIe was *Menu*. As with the BBC Betsie transcoder [BBC 1999], *Menu* moved the menu of the Website to the bottom of the page. Menus are important aspects of Websites as they allow for easy navigation around the pages, but for users to hear the same menu on every page before they hear the actual content can be frustrating. Moving the menu to the bottom allowed it to be easily found without interfering with access to the main content when a user navigated to the page.

Figure 3.7 on page 79 shows an example of applying the three structure-driven SADIe transcodes to a CNN News story. Figure 3.7a shows the original version of the page. Using the CSS associated with the Website, an ontology was created, shown in Appendix B on page 176, to describe the roles that the elements play on the page. For example the menu is defined in the HTML as `<ul class="cnnNavigation">`, where `cnnNavigation` is a reference to a CSS element that defines the style for the list. This had a corresponding class in the CNN ontology with the property ∃ `SADIe:isMainMenu some SADIe:True` to state that this is the main menu of the page. The SADIe transcoding platform accesses the page and then parses the Document Object Model to identify the HTML elements contained within the page. For each element, the transcoder identifies any attributes that the tag may have, for example `cnnNavigation`, and then queries the ontology to see if an equivalent class exists. In this case `cnnNavigation` is discovered and found to be classified as the page menu. A suitable transformation is applied, such as placing the menu at the bottom of the page, and the process is repeated until the entire Document Object Model has been processed. The results of applying all three transformations are shown in Figure 3.7b. The page has had all the clutter removed and the story has been promoted to the top where it can be immediately accessed by a screen reader. The menu has been suppressed to the bottom of the page but is accessible via a link situated near the top. This allows users to easily find the menu when they



(a) A news story from the CNN Website. The CSS has been annotated using an ontology, which can be seen in Appendix B on page 176.



(b) The results of applying the *Defluff*, *Reorder*, and *Menu* transformations to the story shown in (a).

Figure 3.7: Example of a news story from The CNN Website being transcoded by SADIe using established structure-driven techniques. The content has been reengineered to allow the user immediate access to the main content.

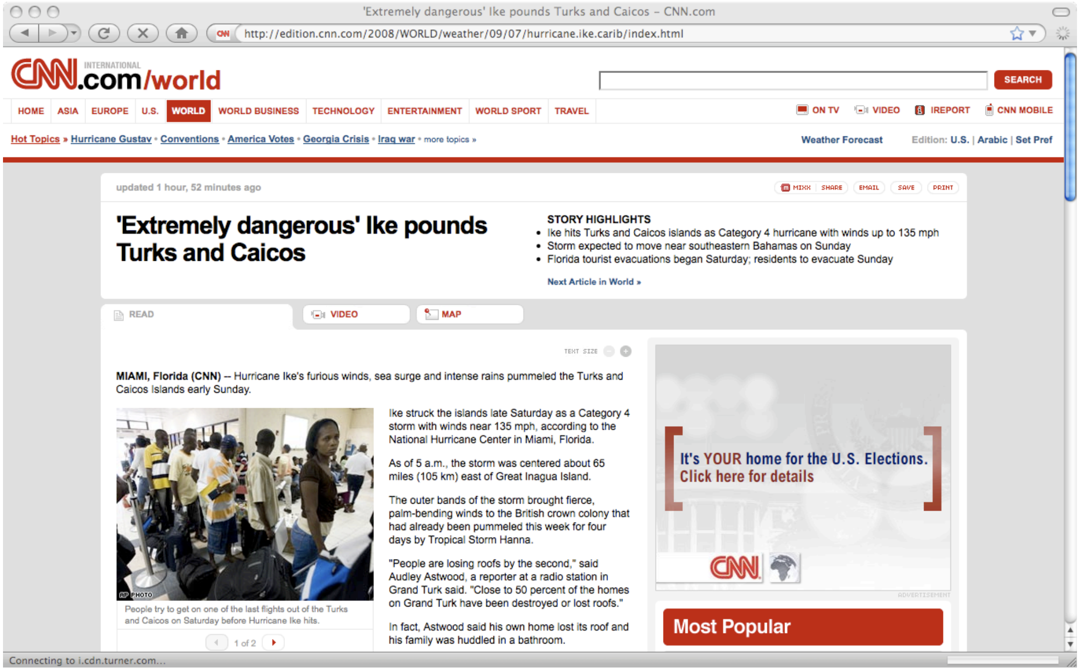
wish to navigate to other news stories.

3.3.2 AxsJAX Adaptations

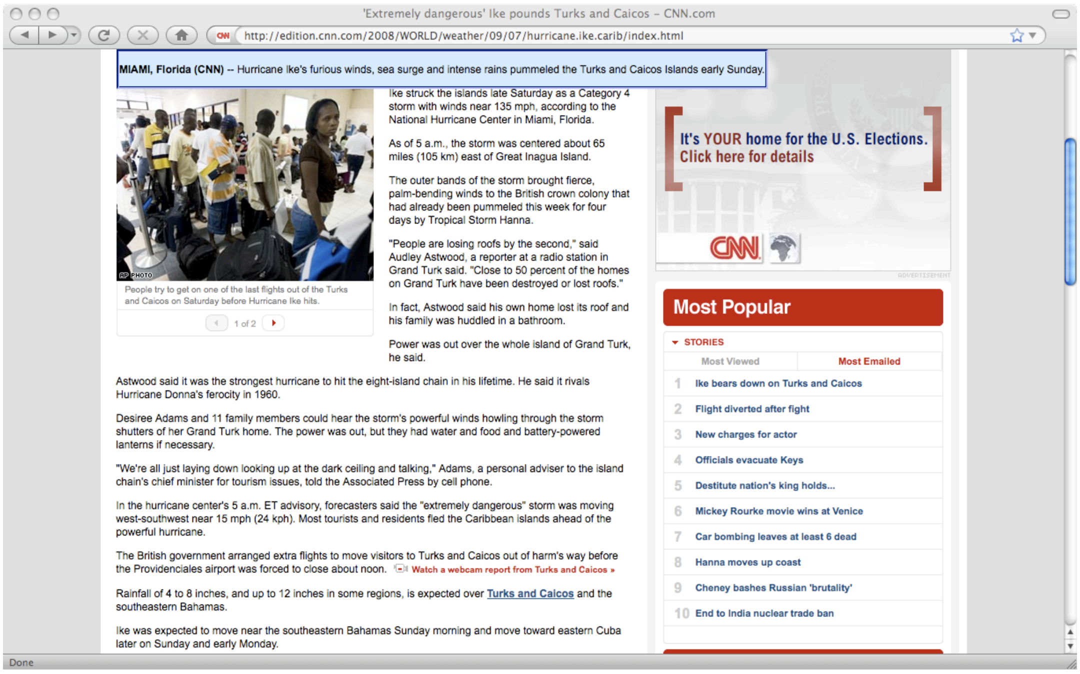
As discussed in Section 2.3.3, while a majority of the Web still consists of static-based documents, there is a trend towards interactive Websites, known as Web 2.0, where users actively contribute to the content [Millard & Ross 2006]. Technologies such as ARIA [Cooper *et al.* 2008] and the AxsJAX Framework [Chen & Ramen 2008] have been developed to allow these areas of rich content to be marked up and made accessible for people using assistive technologies.

A key part of the AxsJAX framework is the generation of Content Navigation Rules. This is JavaScript code that incorporates XPath and XML statements to inform the framework where ARIA statements should be inserted in the page and how the assistive technology should interact with the statement. XPath is a language that can be used to reach parts of a HTML document. XPath expressions can point to explicit nodes, such as `/html/head/meta[1]` which selects the first `<meta>` tag of the `<head>` element of a HTML document. Alternatively, XPath can identify elements with specific attributes, such as `//ul[@class='nav']` to select all unordered list elements with a class value of `nav`. Note how the XPath expressions use a similar mechanism to identify HTML elements as CSS. As `class='nav'` is a CSS element that can be annotated using the SADIe method, the adaptation rules of SADIe were modified so that, rather than manipulate and reengineer the page structure as discussed in Section 3.3.1, the CSS annotations were used to generate appropriate content navigation rules. These were then used in conjunction with the AxsJAX framework to insert ARIA statements into the document content. Injecting ARIA terms into a Web page allowed SADIe to treat all Web documents as interactive applications. Users could move around the page and interact with the content by using various key presses that were applied to the Web page.

Figure 3.8 on page 81 shows an example of applying AxsJAX transcodes. Figure 3.8a shows the original version of the page. Using the CSS associated with the Website, an extended Cascading Style Sheet was created that contained the CSS elements that were to be annotated. For example the element `<ul class="cnnNavigation">` is the HTML that creates the menu so a new CSS class was created as `ul.cnnNavigation{ -uom-structural-role: LinkList; }` so that we can explicitly state the role that this element plays. A full version



(a) A news story from the CNN Website. The CSS has been extended with a new structural role property, which can be seen in Appendix C on page 180.



(b) The results of applying the AxsJax transformations to the story shown in (a).

Figure 3.8: Example of a news story from the CNN Website being transcoded by SADIe using AxsJAX.The modified page appears the same as the original but the content has been injected with ARIA.

of the extended CSS can be seen in Appendix C on page 180. SADIe uses these properties to create the content navigation rules. `ul.cnnNavigation` can be expressed as `//ul[@class='cnnNavigation']` in XPath, which is then used as a variable in the AxsJAX framework JavaScript code, in addition to an assigned key press. In this example the letter `m` was associated with the term `LinkList` so that by pressing one key, users could be taken to the menu of the Website. The same process is repeated for each HTML element that has a `-uom-structural-role` property associated with it.

After SADIe has created the content navigation rules, the AxsJAX JavaScript is inserted into the page. An example of the AxsJAX JavaScript code associated with the CNN Website is provided in Appendix D on page 181. A majority of the code is consistent across all of the SADIe AxsJAX transformations. SADIe uses the template and inserts the XPath-based content navigation rules, which are dependent upon the Website's CSS annotations, into the appropriate areas of the code. The code contains a set of event listeners that monitor what key presses the user is making when they interact with the page. The key the user presses determines which ARIA statements are created. Returning to the CNN menu example, SADIe generates the XPath `//ul[@class='cnnNavigation']` and associates the letter `m` to the node that that path points to. When the user presses the `m` key to reach the menu, the AxsJAX event listener finds the menu node by using the `//ul[@class='cnnNavigation']` expression. Attributes are then dynamically added to the node such that <ul class="cnnNavigation"> becomes <ul class="cnnNavigation" aria-live="rude" aria-atomic="true">. The attribute `aria-atomic="true"` informs the assistive technology that it should read aloud the HTML node as a whole, including any descendent nodes. The attribute `aria-live="rude"` informs the assistive technology that it should read aloud the content of the node immediately, regardless of what the user is currently doing. As the Document Object Model has mutated, due to the updating of node attributes, the assistive technology uses the ARIA statements to identify the menu and allows the user to interact with that element.

Figure 3.8b shows the result of a user interacting with the page after the ARIA statements have been inserted. Visually, the page is identical to the original version, with the exception of the blue highlighted area. However, it should be noted that this is for demonstration purposes only. The blue area indicates the current focus of the screen reader. In this example, pressing the `n` key takes the user directly to the news story.

3.4 Technical Evaluation

A technical evaluation to investigate the validity of the SADIe approach was conducted. This evaluation was primarily concerned with establishing if the annotation of the CSS elements provided enough information to apply transcoding to a number of Websites in a consistent fashion. The transcoding used in the evaluation was the structure-driven transcoding that was incorporated into the prototype of the transcoder, described in Section 3.3.1. It should be noted that at this stage no attempt to establish the usefulness of the transformations for the user was being made. The structure-driven transcoding was used as a means to evaluate whether the transcoding output was as expected. For example, having annotated elements as `Removable`, did the transcoder indeed remove all of those annotated elements?

To ensure that the technical evaluation was conducted on a representative sample of Websites, the Website categorisation system proposed by [Amitay *et al.* 2003] was employed. This system takes into account the purpose of a Website's content and places it into one of eight categories. These categories include Corporate Sites such as Microsoft[3], Search Engines such as Google[4] and eStores such as Amazon[5]. For each of the eight categories, five Websites were chosen giving a total of forty Websites to be evaluated. A list of the forty Websites used and their categorisation is provided in Table E.1 in Appendix E on page 185.

The evaluation consisted of applying transcoding to the entry point of the Website, typically `index.html`. This was to ensure a fair comparison between the pages in the sample was made as the index page is the main page of the Website. It would be an unfair comparison if some Websites in the sample were tested using the main page that usually contained a high amount of content and other Websites evaluated using smaller pages with specific content such as contact details. In addition, the index is also the page that is interacted with by a majority of users therefore successfully transcoding this page is of benefit to more users.

[3]URL: `http://www.microsoft.com/`. Last accessed: 7th September 2008
[4]URL: `http://www.google.co.uk/`. Last accessed: 12th August 2008
[5]URL: `http://www.amazon.co.uk/`. Last accessed: 10th November 2008

For each of the forty pages, the three transcoding techniques of *Defluff*, *Reorder* and *Menu* were applied. After transcoding had taken place, the page was inspected to ensure that the expected outcome had been produced. For *Defluff* this involved asking *"have all obstacles annotated as removable been removed?"* and *"have any elements not marked as removable been removed?"* For *Reorder* this involved asking *"are the blocks of content in the correct order of high, medium and low priority?"*, *"are any high, medium and low priority elements missing?"* and *"have any additional elements not annotated as high, medium or low priority been inserted into the flow of text?"*. Finally for *Menu* this involved asking *"has the menu been placed at the bottom of the page?"* and *"are any menu elements missing?"*. If all three transcodes were applied correctly, and the desired outcomes were achieved, then the transcoding was considered to be a success.

During the evaluation, how the Website created its presentation was noted. Three types of rendering patterns were observed. The first, defined as "pure", used only CSS to create the rendering and layout of the page. The second, defined as "none", used no CSS and relied on presentation tags such as `<font>` and `<table>` to create the rendering and layout of the page. The third, defined as "mixed", used CSS to create the look and feel of the Website, such as font colour and sizes, but used tables to control the layout of the page. For example a two column table could be used to have the menu on the left-hand-side of the page with the content to the right of the menu. A summary of the results, broken down to show the success rate of the various classes of CSS usage, can be seen in Table 3.2 on page 85. The "Site Sample" is the number of pages that were evaluated for that class of presentation usage. "Site Failures" shows the number of Websites that failed to be transcoded correctly by SADIe and the "Sample Error" provides the percentage of Websites that failed to be transcoded from the sample used, in this case forty Web pages. The "True Error Range" uses the sample error and estimates how many Websites would fail to be transcoded using SADIe when applied to the entire population of Websites using a 95% confidence interval [Mitchell 1997]. This provides a more accurate performance level of SADIe. The Mixed CSS usage, for example, gave an error rate of 15% however it cannot be claimed that SADIe will not be able to transcode 15% of the Web. To do this the entire Web would need to be transcoded. However, it can be said that based upon the sample size and the error rate, SADIe will fail to transcode between 2% and 28% of all the Websites that use mixed presentation

Table 3.2: Summary of the results from the SADIe technical evaluation. As expected, SADIe was unable to perform transcoding on Websites that used no CSS. However, for Websites that did use CSS, SADIe was more successful.

CSS Type	***Site Sample***	***Site Failures***	***Sample Error (%)***	***True Error Range(%)***
Pure	9	0	0	0 – 0
Mixed	26	4	15	2 – 28
None	5	5	100	100 – 100
All	40	9	23	11 – 35
Pure/Mixed	35	4	11	1 – 21

styles.

It can be seen that the Websites that used no CSS failed to be transcoded. However this was as expected as the SADIe annotation method makes use of the association between HTML elements and the rendering of the CSS. If there is no CSS then by design, SADIe can not transcode the page. As discussed in Section 3.1, this is a limitation of the approach but as the uptake of CSS increases, more Websites will be exposed to our transcoding approach. All sites that used pure CSS were successfully transcoded. Mixed usage Websites used a combination of CSS and tables for presentation and were not always successfully transcoded. During the evaluation it was observed that a common approach adopted by these mixed presentation Websites was to give the entire table a CSS class value, which SADIe could make use of, but not give the elements within the cells of the table a CSS value. Therefore SADIe could remove or reorder the table as a whole, but the contents within the table were inaccessible to the transcoder.

The observation of "mixed" presentation mechanisms suggests that there needs to be an effort to improve designers' good practice when using CSS. CSS is able to control both the layout and rendering of Web elements and the need for using tables as a layout mechanism is redundant. A key benefit of using a pure CSS-based layout for developers is that it provides flexibility. With CSS, changing the look and layout of a Website can take place in one location and the changes propagate through to all pages. By using a table for layout, designers are locked into a particular layout that needs to be designed when the Website is first constructed. Changing the layout at a later stage requires editing all the pages contained within the Website, which for large organisations can constitutes thousands of pages.

To further summarise the results, the "All" row takes into account all forty Websites used in the evaluation, showing that nine of the forty Websites evaluated failed to be transcoded. The "Pure / Mixed" row factors out those Websites that did not use CSS at all as by design SADIe was be unable to transcode these pages. This improves results and shows that only four of the thirty five Websites evaluated failed to be transcoded.

The technical evaluation provided strong support for using the CSS-based annotation transcoding as a method for transforming Web content. Discounting those Websites that did not use any CSS for the presentation of the content, the technique was able to successfully transcode 89% of Websites, with a true range lying between 99% and 79%. These results were encouraging and gave a solid basis for using SADIe as the transcoding platform upon which to investigate both structure-driven and behaviour-driven transcoding from a user perspective.

3.5 Summary and Conclusions

In this chapter we have seen how an ontology can provide a concise set of terms to annotate the rendering information contained within the Cascading Style Sheet of the Website. Two approaches to annotating the structure of Web elements through the Cascading Style Sheet were investigated. The first involved a two-part ontology approach with the lower-ontology containing elements found within the CSS and importing the abstract terms from the upper ontology. The second involved extending the keywords used within the Cascading Style Sheet and using the abstract terms from the ontology to provide suitable values to the keywords. Both approaches allow a description of the elements to be provided, yet due to the association between Web pages and CSS, a single annotation propagates through to all the pages within a Website. As such, this provides the scalability of heuristic transcoding combined with the accuracy of semantic transcoding.

While both annotation approaches provided a mechanism of describing the roles of Web elements, they both had weaknesses that require further investigation. The two-part ontology blurred the application functionality and structural semantics of the Website. In addition it was taxonomic that made no use of inference engines to discover new facts and relationships between the elements of the Website. This took considerable time to construct and so, from a pragmatic perspective, was a large amount of effort for little gain. In contrast, the CSS extension provided the same level of annotations as the ontology but with less effort. However, there are issues that still surround the approach that need to be addressed. Since CSS was first proposed, there has been an effort to promote the separation of style and content due to the benefits that it provides [Regan 2004]. These benefits include the ability to easily redesign a Website and the opportunity for users to apply different rendering to the page in order to better suit their needs, for example a high contrast style sheet for those with low vision. Our proposal to extend CSS with the `-uom-structural-role` property has the potential to conflate the rendering of a Website with the semantics of the content, which could result in similar problems that the merger of style and content created. Consider a paragraph defined as `<p class="a">` and an associated CSS definition of `.a{font-size: 90%; margin: 4em;}`. Due to the paragraphs smaller font size and wider margins, one can assume that the designer intended for all paragraphs of class `a` to be the abstract of an online document. Explicitly stating this by adding `-uom-structural-role: abstract;` to the CSS definition can have benefits for tools such as SADIe that need to process and adapt the content of the page. However, consider a part of the abstract paragraph defined as `<span class="org">The University of Manchester</span>` with an associated CSS definition of `.org{-uom-structural-role: organisation;}`. No rendering information has been applied to the element and `organisation` is the semantic meaning of the phrase `The University of Manchester` and not a role that it plays on the Website. For such definitions mechanisms such as RDFa and Microformats, discussed previously in Section 2.2.1, are required. The CSS syntax is valid and will not cause any validation errors, but as more annotated data becomes available online, it may be necessary to have a clear separation of the role elements play on the page due to their rendering, and the semantic meaning of the content that is present within the page.

The annotations of the CSS were independent of the transcoding output. Two approaches to adapting elements contained within the Web page were investigated. The first involved established structure-driven transcoding techniques that involve reengineering the structure of the content to suit the requirements of the target user device. The second involved using the AxsJAX framework to inject Accessible Rich Internet Application statements into Web documents. Injecting ARIA terms into a Web page allowed SADIe to treat all Web documents as interactive applications. Users could move around the page and interact with the content by using various key presses that were applied to the Web page.

The construction of the SADIe experimental transcoding proxy provided two advantages. The first was to allow investigations into the technical feasibility of annotating the CSS as a method of exposing the structure of the Web content. A technical evaluation of the method provided strong support for the feasibility of the SADIe transcoding method. Discounting those Websites that did not use any CSS for the presentation of the content, SADIe could successfully transcode 89% of Websites, with a true range lying between 99% and 79%. These results support Research Question 1 by showing that the use of CSS annotations allows for the construction of a consistent and predictable transcoding transcoding platform that can be applied to a diverse range of Websites.

The success of the technical study provided a solid basis for using SADIe as the transcoding platform upon which to investigate both structure-driven and behaviour-driven transcoding from a user perspective. This provided the second advantage of providing a mechanism for making consistent comparisons between structure-driven and behaviour-driven transcoding algorithms. When conducting user studies with both the structure-driven adaptations and the behaviour-driven adaptations, any changes in behaviour while accessing Web content by the participant would be attributable to the transcoding itself. All other factors, including the annotation model and the underlying transcoding engine, would remain constant and therefore provide more robust results.

Chapter 4

Behaviour-Driven Transcoding

The adaptations used by previous transcoding systems have often been based on identifying the structural elements of the content and then reengineering that structure to suit the linear output produced by screen readers. For example some users may be hindered in their access to the main content of the page as it is surrounded by additional elements, such as menus and mastheads. One known approach is to therefore remove all the clutter from the page leaving only the main content. The behaviour-driven transcoding investigated in this thesis involves an understanding of user behaviour techniques captured through a series of summative experimental studies. These studies included ethnographic field studies of eleven participants over a total period of eleven, two hour sessions and the transcripts of twenty users participating in a Web usability study. The combined data of the thirty-one participants was analysed using thematic analysis techniques and led to the development, and validation, of a framework that captured the behavioural strategies that users employed when access to Web content became too taxing. This supported Research Question 2 and allowed for the identification and modelling of coping strategies.

The strategies identified from the summative experimental studies were low level instances of coping that provided insights into how users coped when interacting with Web content became taxing. Analysing the instances of coping strategies revealed that different strategies often had similarities, especially when they were used to achieve the same goal. Grouping strategies that had similar outcomes for the user revealed six patterns that abstractly described what strategies users employed on the Web to achieve their goal. These abstract patterns formed the basis for deriving behaviour-driven transcoding that was incorporated into the SADIe experimental transcoding proxy using the Google AxsJAX adaptation method in order to support Research Question 3.

4.1 Qualitative Experimental Study

To identify and understand the coping strategies users develop when interacting with the Web, a summative experimental study was conducted in conjunction with Henshaws' Society for Blind People (Henshaws') and with approval from the University of Manchester Senate Committee on the Ethics of Research on Human Beings. The study that formed the core of the data acquisition for the coping strategy identification took place during Henshaws' Skillstep to Success IT course for visually impaired users.

Two periods were spent at Henshaws' in total. The first was seven, two-hour sessions that consisted of eight participants four of whom were female and four of whom were male. The second was four, two-hour sessions that consisted of three participants, all three of whom were male. The class was aware that observers from the School of Computer Science were in the room to watch them browse the Web. The class was informed that should they require any assistance, both with the Web and other coursework, then help was available. During the study, the observers sat alongside the students of the class, typically on a one-to-one basis, and observed them use the Web. To ensure that students felt comfortable with the observers, notes were not written in the presence of the students but discreetly in-between watching different members of the class. The reason for this was to prevent students being placed under more stress than was necessary. Students on the course were learning new things that they found difficult and they were aware researchers from the University were present. Adding the additional stress of openly taking notes about the student was considered to be an additional stress factor that should be avoided.

Being both observers and classroom assistants created a conflict of interest that required the use of value judgement to create a balance between the two roles. As an observer, there was a desire to step back and watch browsing events occur naturally and without interference. As a classroom assistant, there was a requirement to assist students with any errors. As a balance, the observers sat quietly and watched the student, asking questions where appropriate. When students faced difficulty, help was not provided until the student explicitly requested assistance. Even when a request was made, students were encouraged to find a solution on their own until it became clear that the student would be unable to achieve their desired goal. At this point assistance was given.

While a majority of the data came from observations, informal discussions were also conducted with the class. This involved asking general questions about the assistive technology being used, as well as group discussions during designated coffee breaks. No student or member of staff was formally interviewed during the observation period and students were free to stop browsing the Web or request that the observations stop at any time. For example, during one session a student was using the Web for private work that they did not want the observers to see. The student offered to do some general browsing that could be observed, but it was felt that doing this would be imposing on the student. In addition, having the student conduct browsing for the observer's benefit would have been artificial and not provided accurate results or observations. Therefore the student's offer was declined and they were allowed to browse in private.

4.2 Using Thematic Analysis To Define Coping Strategies

The summative experimental study conducted at Henshaws' was akin to an ethnographic study where qualitative data was captured through participant observation [Maanen 1979; Pole & Morrison 2003]. Therefore techniques predominantly used in social science to analyse such data were drawn upon. Two common methodologies used in qualitative data analysis are grounded theory and thematic analysis. Both approaches involve coding notes taken from observations to identify similar phrases, relationships, patterns and themes. These are gradually reduced to a small set of generalisations that can be used to construct theories [Miles & Huberman 1994]. However there are differences between grounded theory and thematic analysis in that grounded theory includes theoretical sampling of the population from which the data was acquired [Rice & Ezzy 1999]. For the study described in Section 4.1 no population sampling took place. Any student of the course at Henshaws' was observed and entered into discussions with regardless of ability, age or sex. The only prerequisite was that the participant had to be visually impaired and required assistive technology to access the Web. However, it should be noted that the purpose of the research was to establish if coping strategies could be identified and used to construct behaviour-driven transcoding and not to develop an exhaustive corpus of coping strategies. Therefore thematic analysis was used as the basis of the data analysis.

Braun & Clarke [2006] provide a six-phase guide to ensure that thematic analysis is conducted in a systematic fashion. These are phase 1: familiarising the data; Phase 2: Generating Initial Codes; Phase 3: Searching for Themes; Phase 4: Reviewing Themes; Phase 5: Defining and Naming Themes; and Phase 6: Final Report of the Findings. The qualitative data captured during the Henshaws' study was analysed using this guide. The six phases of the analysis and the intermediary outcomes are described below.

4.2.1 Phase 1: Familiarising The Data

Before analysis can begin, the analyst needs to immerse themselves within the data in order to become familiar with the content and the data that has been captured [Braun & Clarke 2006]. For the Henshaws' data we were the observers who captured the data, providing a good prior knowledge of the data and the context in which it was captured before the analysis began. In addition, after each session, the data was converted from notes written in a log-book in the classroom to a collection of observations written in report style. This process provided a good understanding of what information was contained within the data.

Before Phase 2 began, the reports were reread to ensure that the contents were revised and seen as a completed data set. It should be noted that only data from the first Henshaws' session was analysed at this stage. This was because we wished to define what constituted a coping strategy and then validate that definition by applying the second set of data later.

4.2.2 Phase 2: Generating Initial Codes

Phase 2 involves the production of initial codes from the data. Codes identify features of the data that may be of interest during the analysis [Braun & Clarke 2006]. They can be discarded at a later stage or changed if required. At this stage the aim is to highlight key parts of the data and not produce definitive themes. To act as a guide, for every note recorded in the data, the following questions were asked:

1. What is the difficulty that the user is having?

Table 4.1: An example of the initial coding performed on the Henshaws' data. The data exerpt from the study notes has been condensed into two bullet points to capture what the difficulty and solution is.

Data Excerpt	***Initial Coding***
P7 navigated to Wikipedia as the Google search results had indicated this was where the UK Number 1 Singles could be found. P7 was initially confused with this [wikipedia] page as they were expecting a list of artists and singles, rather than a list of years. As P7 became confused by the page, they pressed the "back button" to go back to Google to ensure that they had selected the correct link. Satisfied... [Wikipedia] was the correct page to be reading, P7 navigated to it again and then looked around the page. Tentatively P7 clicked "List of number-one singles from the 1960s (UK)"...	• Confused and lost within a page. • Return to Google home page and check the correct link has been selected.

2. What, if anything, is the user doing to overcome this difficulty?

Table 4.1 on page 93 shows an example of how the initial coding process took place. The data was read through and for all notes contained within the data set, a brief summary of the difficulty the user faced and how they overcame that difficulty was noted down. In the example shown in Table 4.1, the user was disconcerted by the overwhelming number of links contained within the Wikipedia page and also confused as the data did not quite match what they were expecting. The user assumed that they had selected the wrong link from the search results and so returned to Google, checked the search results again and then moved back towards the Wikipedia page.

Continuing this process with all the data gave a set of thirty-six instances of problems that occurred during the first set of study sessions in-conjunction with how the users attempted to overcome those problems. The set of thirty-six instances of coping is provided in Table F.1 in Appendix F on page 187. It was encouraging at this point to note that some strategies were similar. For example in Table F.1 there are five occurrences of returning to Google and starting again. This suggested that strategies were repeatable and therefore more suitable to use as a basis for deriving transcoding solutions during the later stages of the analysis.

Table 4.2: The combined initial coding performed on the Henshaws' data. This shows the difficulties and strategies identified from the initial coding reduced from thirty-six to eighteen. Unlike the example in Table 4.1 the data is not shown.

	Difficulty	***Strategy***
1	User cannot access the content for various reasons.	Ask the observer for help.
2	Cannot enter text into a form.	Turn Jaws into 'Form Mode'
3	Confused as to whether this is the correct page.	Move around the page trying to identify cues (logo, title) to confirm that the correct Website is being displayed.
4	Cannot read a news story because of automatically changing content.	Wait for the story to roll around again.
5	Cannot find search result.	Press the 'tab' key several times to get to the first search result.
6	Data was lost in a form.	Physically abuse the keyboard and start again.
7	Difficulty finding a search button as it is placed before the search box, not after.	Use the 'tab' button to go forward several times then go backwards several times.
8	User cannot access the content for various reasons.	Give up.
9	Lost and confused within the page.	Return to Google and follow the same link.
10	Frustration at listening to the menu on every page.	Turn Jaws off and struggle with ZoomText.
11	Trying to find information contained within the page.	Rapidly scroll down to approximately where the results may be, re-orientate and then scroll some more.
12	Trying to find information contained within the page.	Sit and listen to everything.
13	What the user expected at the link destination and what they discover are not the same.	Click several links that may lead to the desired information.
14	Lost within the page.	Try to use the intra-page menu.
15	Confused by different types of menu contained within the page (internal links and Website links)	Navigate between the two menus several times before deciding which one to use.
16	Confusion because text looks like a link but it not.	Try and click links until something happens.

Continued on next page

Table 4.2 – continued from previous page

	Difficulty	***Strategy***
17	Unable to find the desired link in a page so return back to the top to look there.	Use prior knowledge of menu location to try and identify a link.
18	Unable to find information contained within the page that is rendered wider than the browser window.	Scroll vertically until the horizontal scroll bar and scroll the window horizontally.

Combining matching coping strategies reduced the number to eighteen, all with similar difficulties. This reduction is shown in Table 4.2 on page 94 which shows the difficulties and strategies identified from the initial coding but reduced from thirty-six to eighteen. Unlike the example in Table 4.1 the data is not shown.

While these provided a good insight into what users did when faced with problems with a Web page, they did not capture or define what a coping strategy was. Therefore the data was analysed again but this time identifying codes that identified potential properties and characteristics that the coping strategies may have. For example noting down the user, what assistive technology they were using, what Web page they were accessing and why they were on that Website.

Table 4.3 on page 96 shows an example of how the second coding process took place. The data was re-analysed and for all notes contained within the data set additional characteristics, no matter how trivial they initially appeared, were noted down. In the example shown in Table 4.3, the participant, P7 was accessing Wikipedia with a Screen Magnifier to find specific information about number one UK singles when they became confused by the overwhelming number of links contained within the Wikipedia page.

4.2.3 Phase 3: Searching For Themes

Phase 3 takes the initial codings from the data and tries to group them into some initial themes. There are many ways in which this can be achieved. Tables and mind-maps are common but it is also possible to use post-it-notes with each code written on it and then try and group them by moving the post-it-notes around [Braun & Clarke 2006]. For Henshaws', each of the eighteen strategies identified in Table 4.2 was taken in turn and the data, along with the codings, were analysed to try and find common features. This analysis allowed the first set of thematic maps for each strategy to be developed.

Table 4.3: An example of the secondary coding performed on the Henshaws' data. The data excerpt from the study notes has been condensed into two bullet points to capture what the difficulty and solution is. Characteristics of the data and the user have also been noted.

Data Excerpt	***Initial Coding***
P7 navigated to Wikipedia as the Google search results had indicated this was where the UK Number 1 Singles could be found. P7 was initially confused with this [wikipedia] page as they were expecting a list of artists and singles, rather than a list of years. As P7 became confused by the page, they pressed the "back button" to go back to Google to ensure that they had selected the correct link. Satisfied... [Wikipedia] was the correct page to be reading, P7 navigated to it again and then looked around the page. Tentatively P7 clicked "List of number-one singles from the 1960s (UK)"...	• Confused and lost within a page. • Return to Google home page and check the correct link has been selected. • Used by P7. • Accessing Wikipedia. • Using screen magnifier. • Browsing for five sessions. • Trying to find specific information.

Figure 4.1 on page 97 shows the thematic map for the return to Google and start again strategy that users employed on Web pages such as Wikipedia. To identify common themes from this strategy, all the codes were collated and then moved around so that they were grouped into common blocks. For example this strategy was used by both P5 and P7 therefore a common theme is that these are users of the strategy. The main themes are shown in the rounded boxes with examples of each theme placed into square boxes. It should be noted that not every instance of the example is shown. For example *"Confused and lost within a page so checked the page was correct"* occurred three times but is only shown once because we were not interested in how may times codes occurred. The interesting aspect was the different types of codes and how they could be fitted together.

4.2.4 Phase 4: Reviewing Themes

Phase 4 takes the candidate themes from Phase 3 and then refines them. This may involve merging some themes but also using the data to try and identify themes that may have been overlooked but add value to the final conclusions and definitions drawn from the data [Braun & Clarke 2006].

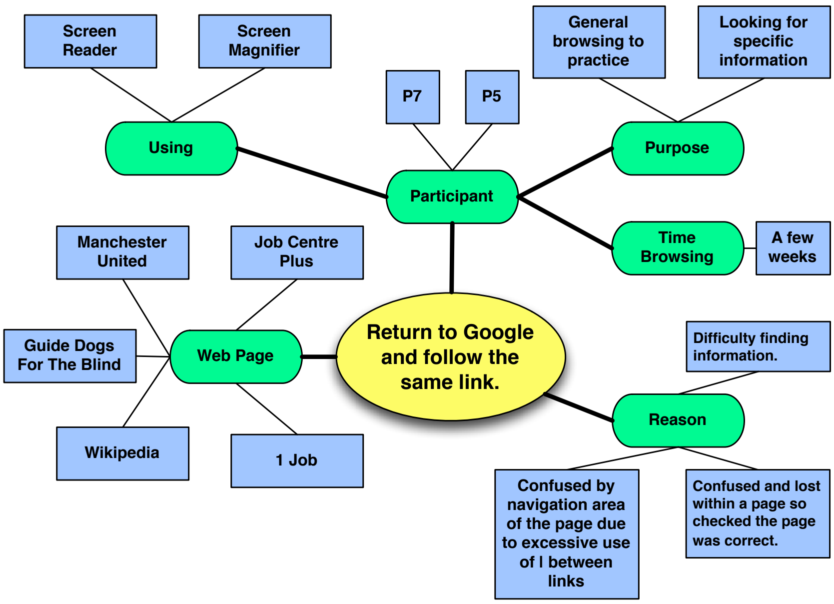


Figure 4.1: The first thematic map for the *"Return To Google"* strategy. The main themes are shown in the rounded boxes with instances in the square boxes. Note that not every instance is shown, only one of each type. *"Confused and lost within a page..."* occurred three times but here is only shown once.

As a first stage to the refinement process, the eighteen strategies along with the candidate themes were looked at to try and remove superfluous elements. For example, in the Return to Google Thematic Map in Figure 4.1, both users had been browsing for a number of weeks. Therefore this did not add any distinguishing features to this strategy as it was used by users with the same level of training. However the coding was retained because some strategies, such as Give Up were used by all levels of experience. Conversely the strategy of finding the search box and tabbing both before and after it to identify the button was used by an experienced user who had been using the Web for a number of years. They knew that a search box must have a button to press and used this knowledge to locate it on the page. While we have no evidence, it is conceivable that some of the less experienced users would have given up or asked for help if they had not immediately found the button they had to press to initiate the search.

The second stage was to try and refine the names of the existing themes. For example *"Using"* was the term used to illustrate what device users were employing when they had difficulty. However this was changed so that rather than indicate what assistive technology was used, it demonstrated what the strategy required in order to be applied. Some strategies, such as *"Return to Google"* could be used with any assistive technology. Others, such as *"turn JAWS into Form Mode"* could only be used with a screen reader. However *"Ask for help"* could not be used with any assistive technology. It required human intervention and so this was renamed *"Device Required for the Coping Strategy"*.

The final stage was to identify any differences between the strategies that had been overlooked. Again the themes for each of the eighteen strategies were analysed to see if further refinements could be made. One refinement made was to classify the strategies themselves. This was because some strategies appeared more useful than others. For example *"Retuning to Google and trying again"* seemed more productive to solving the problem than *"Physically abusing the keyboard"*. They were both valid strategies in response to a certain situation but the former appeared more productive in a browsing environment. Reviewing the literature of psychological coping strategies revealed that these different approaches had been previously identified and grouped as problem-focused coping strategies; emotion-focused coping strategies; and mixed problem- and emotion-focused coping strategies by [Folkman & Lazarus 1985]. This therefore provided a useful way to further distinguish between coping strategies.

A second refinement was the reason for the strategy. On the original thematic maps, the actual problem was recorded such as *"Confused by navigation area of the page due to excessive use of | between links"*. However some of these problems were not related. For example *"Failure to enter data into a form"* was due to the screen reader requiring the user to activate a special forms mode with special key combinations. This wasn't a problem with the page itself, but because of how the assistive technology interacted with the page. Conversely, designers inserting vertical bars between links could be easily rectified by removing those bars and using CSS to apply rendering to the links so that visually the page looks the same. Therefore the reason for the coping strategy being required was further defined as being a fault of the page design or a fault of the technology itself. A revised version of the Return to Google Thematic Map is shown in Figure 4.2 on page 99.

4.2.5 Phase 5: Defining and Naming Themes

Phase 5 takes the final themes identified from Phase 4 and refines each theme so that it defines what captures the essence of what the theme is about. Together

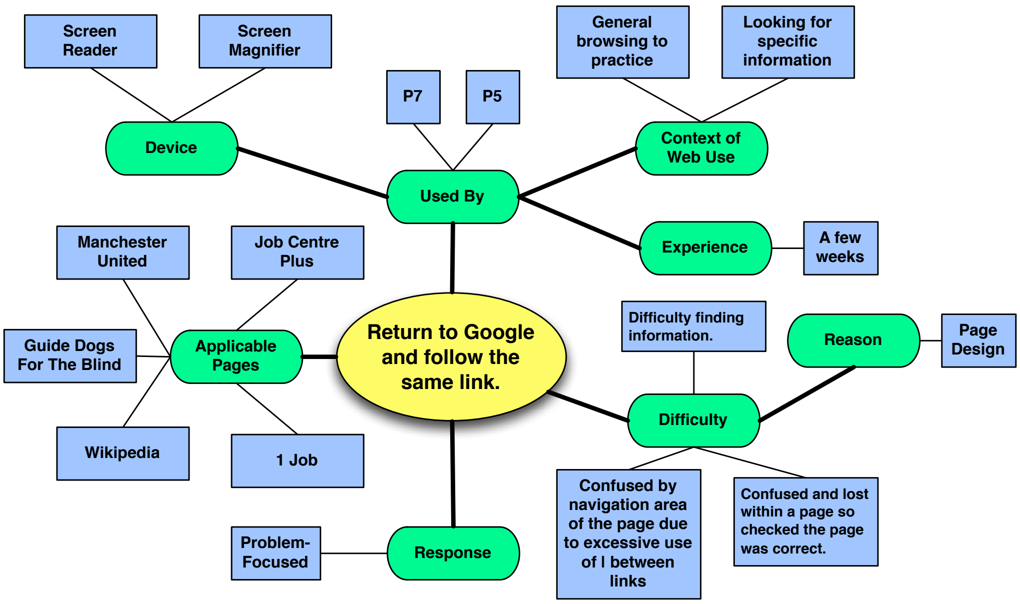


Figure 4.2: The revised thematic map for the *"Return To Google"* strategy. Note how the themes have been renamed to reflect the purpose of the theme as well as additional themes being added.

each theme provides the overall story of the analysis [Braun & Clarke 2006]. Therefore the themes identified from Phase 4 can be named and defined as:

Cause of Difficulty: A number of reasons why difficulties arose when accessing Web content could be identified. Poor layout and design was one cause of difficulty, such as placing a search button before the input form forcing participants to move backwards through the content to execute a search. A further area that caused participant problems was a lack of familiarity with the screen reader, preventing the user from inserting text into a form correctly.

Experience of the Coping Strategy Developer: The experience of the user helped determine the strategy that was employed. For example, an experienced user knew approximately how many times to press the `tab` key to move the focus of the cursor close to the search results of Google in order to avoid the advertisements and the masthead. A novice user, however, would allow the screen reader to continually read out content as they were unfamiliar with the page and unfamiliar with the shortcuts provided by the assistive technology.

Users of the Coping Strategy: During the study some users were using the same or similar coping strategies to access content within Web pages. While twelve of the eighteen strategies were performed by only a single user, it was still encouraging to note that six were performed by more than one user, suggesting that there were strategies which were common among visually impaired users. While this may have been as users became familiar with the pages through the course of the sessions, there was also an element of education as the course tutors provided advice to the students as they learnt how the access the Web.

Psychological Response of the Coping Strategy: In Section 2.5.2 a number of psychological coping strategies were discussed. One of these studies placed coping strategies into three groups — problem-focused coping strategies; emotion-focused coping strategies; and mixed problem- and emotion-focused coping strategies [Folkman & Lazarus 1985]. While a majority of the coping strategies observed during the Henshaws' study could be classified as problem-focused, for example pressing the `tab` key several times to get to the first search result, there were instances of emotional-focused coping strategies, such as physically abusing the keyboard.

Application of the Coping Strategy: In addition to the number of users that employed a coping strategy, strategies were also applied to a variety of Web pages. While eleven of the strategies were only performed on a single Website, it was encouraging to note that there existed strategies that were performed on multiple Websites. This implies that there are strategies that are generic enough to be applicable to a diverse range of sites.

Device Required for the Coping Strategy: Some of the coping strategies observed varied depending on the device that the participant was using. Some strategies were only applicable to screen magnification software. For example, one problem identified involved users attempting to understand the content through horizontal scrolling. A screen reader would not be affected by the need for horizontal scrolling as it would read the source code aloud and ignore the presentation. However there were a number of strategies that were neutral and could be used on a wide variety of clients. Indeed nine of the strategies observed at Henshaws' were used on both screen magnification software and screen reader software.

Context of the Coping Strategy: As the participants took part in the study, it was noted that different strategies were used depending upon the context in which the Web was accessed. Three contexts were identified from the data. Browsing occurred when the participants had no overall goal and were merely practising their Web browsing skills or seeing what information could be found. This was similar to information foraging theory where users navigate from hyperlink to hyperlink looking for information [Bates 1989]. Specific information search occurred when the participants had a goal, for example tasks given to the students by the course tutors to direct their learning or discovering the address of a company that participants wanted to apply for a job to. This was similar to the exploratory search identified in [White *et al.* 2006] whereby users want to find specific information in a domain that they have little knowledge of. Finally, detailed reading occurred when participants were interested in the information on the page and were happy to read all the text. It should be noted that the coping strategy study focused on what strategies users employed when using the Web and so only a broad categorisation is given. For a more in depth discussion of Web usage, see [Yesilada *et al.* 2007a].

4.2.6 Phase 6: Coping Definition

Phase 6 is the final report of the findings of the thematic analysis. By taking all the themes into account, the following definition of a coping strategy was constructed to allow for future identification of coping strategies to be used as a basis for behaviour-driven transcoding functionality:

> A technique based upon a *psychological response* used by *one or more users* to overcome a problem due to some aspect of the *client or content*, developed through *various levels of experience* that is applicable to *one or more Web pages* using *one or more devices* in a specific *context of use*.

Where we define: *psychological response* as either problem-focused, emotional-focused or mixed-response strategies; *One or more users* to be either an individual who uses the coping strategy or the wider community; *Client or content* to be either problems caused due to the design of the Web page or because of the access technology such as Jaws or ZoomText; *various levels of experience* to be the level of experience of the users who use that strategy. For example beginners, intermediate and expert users; *one or more Web pages* to be those strategies that are narrow in scope and apply to a single Web page or those that are more generic and applicable to a diverse range of pages; *one or more devices* to be the assistive technology that the user is using such a screen reader or screen magnifier; *context of use* to be the task that the user is currently performing on the Web, such as browsing, search or thorough reading of content.

Taking *"Return to Google and Try Again"* as an example, we can say that:

> Returning to Google to follow the same link is a problem-focused strategy used by a community of users due to being lost and confused within a Web page due to poorly designed pages. The strategy is used by intermediate users on multiple pages using all clients in multiple contexts of use.

Table 4.4 on page 103 shows the eighteen coping strategies identified in Table 4.2 classified using the definition of a coping strategy described above. Each column of the table represents a factor identified from the thematic analysis.

4.3 Validating the Coping Definition

The coping strategy framework defined in Section 4.2 provided a good foundation with which to identify and categorise coping strategies. While the data was captured from eight participants with a variety of skills over seven sessions, it was based upon a study designed for the purpose of identifying coping strategies. It therefore lacked validation from additional data sources to demonstrate that the definition was applicable to more than one data set. To redress this issue, two sets of validation tests were applied to the coping strategy definition.

The first validation method was to use the data obtained from the second Henshaws' study and apply the coping strategy definition described above to the data. The second Henshaws' data set had been collected in the same fashion as the first set but had not been used to construct the original definition. It could therefore act as a validation set. The initial analysis conducted in Phase 2 of the Thematic Analysis described in Section 4.2.2 was repeated. Therefore the data was analysed and for every note recorded in the data, the following questions were asked:

Table 4.4: Classification of coping strategies observed at the first Henshaws' study using the definition defined in Section 4.2

	Strategy	*Response*	*Used By*	*Difficulty*	*Cause*	*Experience*	*Application*	*Client*	*Context*
1	Ask the observer for help.	Mixed	Community	Content inaccessible beyond the skill of the user	Page Design	Beginner	Multiple Pages	Human	Multiple Contexts
2	Turn Jaws into 'Form Mode'	Problem-Focused	Community	Failure to enter text into a form	Screen Reader	Beginner	Single page	Screen Reader	Information Search
3	Move around the page trying to identify cues to confirm that the correct Website is being displayed.	Problem-Focused	Single User	Unsure if the correct page is being displayed	Page Design	Intermediate	Multiple Pages	All Clients	Multiple Contexts
4	Wait for the story to roll around again.	Emotion-Focused	Single User	Difficulty reading a news story as it switches between three different articles.	Page Design	Beginner	Single Page	Screen Magnifier	Browsing
5	Press the 'tab' key several times to get to the first search result.	Problem-Focused	Community	Difficulty finding search results.	Page Design	Expert	Single Page	Screen Reader	Information Search
6	Physically abuse the keyboard and start again.	Emotion-Focused	Single User	Loss of data in a form.	Page Design	Beginner	Single Page	Computer	Information Search
7	Use the 'tab' button to go forward several times then go backwards several times.	Problem-Focused	Single User	Finding the search button located behind the input form to instigate the search.	Page Design	Expert	Single Page	Screen Reader	Information Search
8	Give up.	Emotion-Focused	Single User	Finding and image with no alternative description.	Page Design	Expert	Single Page	Screen Reader	Information Search
9	Return to Google and follow the same link.	Problem-Focused	Community	Lost in the Web page.	Page Design	Intermediate	Multiple Pages	All Clients	Multiple Contexts
10	Turn Jaws off and struggle with ZoomText.	Problem-Focused	Single User	Frustration of listening to a menu repeated for every page.	Page Design	Beginner	Single Page	Screen Reader	Browsing
11	Rapidly scroll to approximate location of the results, re-orientate and then scroll again.	Problem-Focused	Community	Finding information within a page.	Page Design	Intermediate	Multiple Pages	Screen Magnifier	Multiple Contexts
12	Sit and listen to everything to Find Desired Information.	Emotion-Focused	Community	Finding information within a page.	Page Design	Intermediate	Multiple Pages	Screen Reader	Multiple Contexts
13	Click several links that may lead to the desired information.	Problem-Focused	Single User	Mismatch between link title and discovered information.	Page Design	Intermediate	Multiple Pages	All Clients	Multiple Contexts
14	Try to use the intra- page menu.	Problem-Focused	Single User	Lost within large chunks of text.	Page Design	Beginner	Single page	All Clients	Information Search
15	Repeatedly navigate between two menus before choosing.	Problem-Focused	Single User	Confusion caused by inter- and intra-page menu being together.	Page Design	Beginner	Single page	All Clients	Information Search
16	Try and click links until something happens.	Problem-Focused	Single User	Confusion between links and text that looks like a link	Screen Magnifier	Beginner	Multiple pages	Screen Magnifier	Information Search
17	Use prior knowledge of menu location to try and identify a link.	Problem-Focused	Single User	Unable to find a desired link.	Page Design	Intermediate	Single page	All Clients	Information Search
18	Scroll vertically until the horizontal scroll bar then scroll horizontally.	Problem-Focused	Single User	Unable to find information because the page is rendered wider than the browser window.	Page Design	Beginner	Single page	Screen Magnifier	Browsing

1. What is the difficulty that the user is having?
2. What, if anything, is the user doing to overcome this difficulty?

Once the data was read through and all the notes contained within the data set had been used to identify the difficulty the user faced and how they overcame that difficulty, a second phase of analysis was employed. This involved going through each coping strategy / difficulty pair and asking:

1. Can this difficulty and its solution be described using the existing coping strategy model?
    (a) Are there elements within the coping strategy model that cannot be answered by the data?
    (b) Are there elements within the data that cannot be applied to the coping strategy model?

These tests were successfully applied to the data and supported the suitability of the definition. The results of applying the coping strategy definition to the second Henshaws' study data can be seen in Table G.1 in Appendix G on page 190.

The second validation test was to use independent third-party data acquired from an evaluation of the Non-Visual Access to the Digital Library (NoVA) Project [Craven & Brophy 2003]. The evaluation was conducted in order to understand the difficulties users face when accessing Websites to search for information. As such, the data was independent from our work and unbiased towards the identification of coping strategies.

The coping strategy definition was applied to the data acquired from [Craven & Brophy 2003] in the same manner used for the second Henshaws' data set. Data regarding the experience level of the user was unavailable from the NoVA evaluation transcripts. For this reason, the column marked "Experience" had the value "NULL" inserted. However, with the exception of experience, the remaining elements of the definition could be identified from the NoVA data set. This demonstrated that the coping strategy definition was flexible and accurate enough to be applicable to multiple data sources. The results of applying the coping strategy definition to the data can be seen in Table H.1 in Appendix H on page 191.

4.4 Abstract Patterns of Coping Strategies

Using the framework of coping strategies defined in Section 4.2, a number of coping strategies were identified. These included eighteen from the first Henshaws' study, eight from the second Henshaws' study and twenty-two from the NoVA evaluation data. These strategies, however, represented all the instances of coping identified from the data. As such, they were described in a manner that was specific to the context in which they were identified and classified. To be of practical benefit to users, similar transcoding needs to be applied to as diverse a range of Websites as possible to allow users to interact with the content in a consistent fashion. Therefore closer inspection of the identified coping strategies was necessary in order to develop high level abstract patterns, extracted from those coping strategies already identified. By producing abstract patterns, generic strategies could be discovered, and hence generic transcoding functionality developed that was applicable to a wider range of Websites.

To identify abstract patterns, coping strategies that were used to achieve a specific task were grouped together. Table 4.5 on page 106 shows an example of how the strategies were grouped. The strategies in this table all had the similar goal of allowing the user to identify the area of the page that contained information they were interested in reading. While the goal was the same, the approaches taken often differed. For example, one strategy in this group involved rapidly scrolling down the page to approximately where the results may be, followed by a re-orientation and then more scrolling. An alternative strategy involved using the Heading List Dialogue Box built into the screen reader to gain an overview of the content and move the cursor to the section that seemed most relevant. While the two approaches were different the end goal was similar in that the user attempted to avoid the clutter contained within the page and reach content that was of interest to them.

Appendix I on page 193 shows all the strategies grouped from the three studies. Six groupings were identified which were defined as Candidate Chunk Discovery Strategy; Masthead Avoidance Strategy; Clustered Element Strategy; Probing Strategy; Backtracking Strategy; and Withdrawal Strategy. These strategies are discussed further in Section 4.4.1; Section 4.4.2; Section 4.4.3; Section 4.4.4; Section 4.4.5; and Section 4.4.6 respectively. A seventh grouping of Unclassified is also shown. This grouping was for strategies that did not fit with any of the other strategies available. It should be noted however, that the purpose of the

Table 4.5: An example of grouping similar coping strategies. The coping strategies presented in this table were employed when the user attempted to identify the area of the page that contained information they were interested in reading. The ID of the patterns is of the format data.number where data represents the data set that the strategy came from with H1 representing the Henshaws' Period One data set; H2 representing the Henshaws' Period Two data set and N representing the NoVA evaluation data set. Number is the ID of the strategy from the given data set. For example H1.2 is coping strategy two from the Henshaws' Period One data set

	Strategy	*Response*	*Used By*	*Difficulty*	*Cause*	*Experience*	*Application*	*Client*	*Context*
H1.11	Rapidly scroll to approximate location of the results, re-orientate and then scroll again.	Problem-Focused	Community	Finding information within a page.	Page Design	Intermediate	Multiple Pages	Screen Magnifier	Multiple Contexts
H1.14	Try to use the intra- page menu.	Problem-Focused	Single User	Lost within large chunks of text.	Page Design	Beginner	Single page	All Clients	Information Search
H1.15	Repeatedly navigate between two menus before choosing.	Problem-Focused	Single User	Confusion caused by inter- and intra-page menu being together.	Page Design	Beginner	Single page	All Clients	Information Search
H2.8	Use Heading List Dialogue Box	Problem-Focused	Single User	Significant amount of content on a page.	Page Design	Beginner	Multiple Pages	Screen Reader	Browsing
N.10	Quickly tab down the page and memorise links that may be useful so they can be returned to and investigated thoroughly later	Problem-Focused	Community	Finding Information	Page Design	NULL	Single Page	Screen Reader	Information Search
N.18	Quickly tab down the page listening to partial words as the user knows what to expect	Problem-Focused	Single User	Finding information on a familiar page	Page Design	NULL	Single Page	Screen Reader	Information Search
N.22	Press "Alt + Enter" to skip links and move to next block of text	Problem-Focused	Single User	Overwhelmed by the mount of information	Page Design	NULL	Single Page	Screen Reader	Information Search

research was to demonstrate the feasibility of the approach of behaviour-driven transcoding and not to produce a complete corpus of coping strategies. Further investigations may reveal additional groups to which strategies currently deemed as Unclassified belong.

Having grouped the strategies, the next phase of the analysis was to find common elements in the approach that the users had taken. To do this, each strategy of the group was taken and the states that the user followed noted. For example, when rapidly scrolling down the page, users were trying to reach the next area that may contain relevant information. When using the Heading List Dialogue Box, users were trying to reach the next area that may contain relevant information. The approaches were different but abstractly the interaction with the page was similar. Looking at the stages each strategy entailed allowed abstract patterns to be devised that described how the strategies helped users interact with the page and for what reason.

4.4.1 Candidate Chunk Discovery Strategy

A chunk, or semantic textual unit, is a coherent block of information contained within a page [Buyukkokten *et al.* 2001]. When searching for information, users will try to identify the chunk of the page that contains the information that

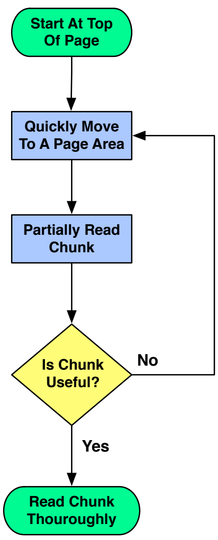


Figure 4.3: The Candidate Chunk Discovery Strategy enables users to identify areas of the page that contain the information that they are interested in reading.

they are interested in reading. In large pages, this will involve quickly identifying candidate chunks that may or may not contain the information and scanning these to see if they have discovered an area of interest. The process of identifying chunks and scanning them continues until either the correct information is discovered, in which case a thorough reading occurs, or the user has no more chunks to inspect and a new page is visited. For example, during the NoVA evaluation this strategy was observed in Test 3 where the user was looking for a Website that contained the weather forecast for Manchester. The user tabbed between each link but only listened to a partial description of the link so that they could get a feel for the content at the link destination. If the link was of interest, then the user listened further, eventually settling on BBC Weather as they knew that they could get weather forecasts from the BBC Website. The pattern users employ when using the Candidate Chunk Discovery Strategy is abstractly defined and shown in Figure 4.3 on page 107.

4.4.2 Masthead Avoidance Strategy

A majority of Websites have masthead elements that sit at the top of the page. The masthead is typically the same for every page and contains the banner, which informs the user what the Website is, and the menu. The Masthead Avoidance Strategy allows users to avoid the frustration of listening to the masthead repeated for every page that is visited within a Website. For example, in the first Henshaws'

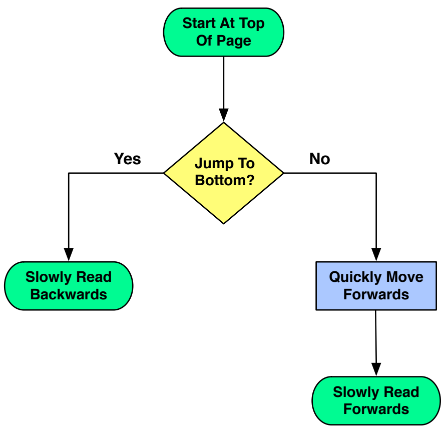


Figure 4.4: The Masthead Avoidance Strategy allows users to avoid the frustration of listening to the same content repeated for every page of a Website.

study this strategy was observed in Session 1 when the user was trying to reach the Google search results. The user rapidly pressed the `tab` key to avoid the banner and the adverts, stopping just after they had reached the first search results. The pattern users employ when using the Masthead Avoidance Strategy is abstractly defined and shown in Figure 4.4 on page 108.

4.4.3 Clustered Element Strategy

When interacting with Web elements, users are aware that some elements typically occur close to each other. For example a search form will typically have a text entry field and a search button close to each other. The Clustered Element Strategy is used by users to find elements that they know should be located close to each other. Users will identify an element that they wish to interact with and then move around that element looking for related items. Users will also use a cue to inform them that they have moved too far away from the element and will use the cue to search in alternative places. For example, in the first Henshaws' study this strategy was observed in Session 1 when the user was searching for a graph on the Progress Media Website. The user entered search terms into a text

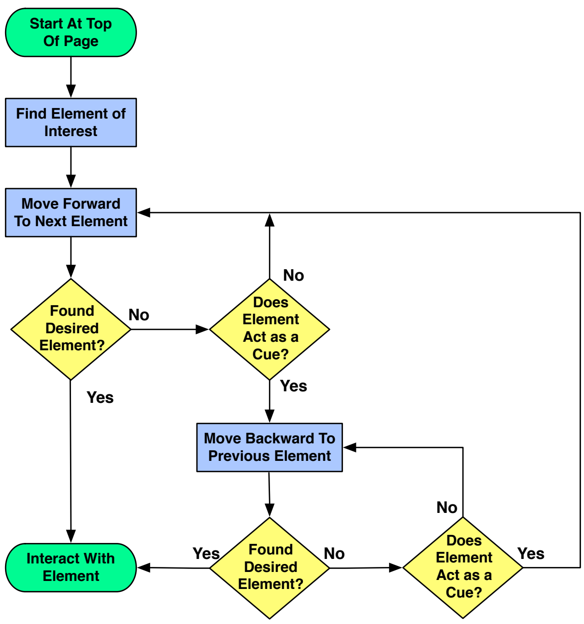


Figure 4.5: The Clustered Element Strategy helps users find elements that they know should be located close to each other.

field and then pressed the `tab` key to move forward to find the button to press that would instigate the search. The user knew the button would be close to the search box but after moving forward several elements could not find it. The user moved backwards and found the button located before the text box. The pattern users employ when using the Clustered Element Strategy is abstractly defined and shown in Figure 4.5 on page 109.

4.4.4 Probing Strategy

When interacting with Web elements, users are sometimes unsure as to which link they should select. Usually this is due to badly labelled links or links that have similar names but have different content at the link destination. In cases like this, users typically look for candidate links and then select the ones that they think will take them to the desired page. A user then scans the page and if the content seems appropriate, reads the content thoroughly. If not, they return back to the previous page, or the menu if it was a menu item, and then select other links to click. For example, in the first Henshaws' study this strategy was observed in Session 4. The user was interested in finding flight information but there was no link that seemed appropriate. The closest link was entitled *"Book*

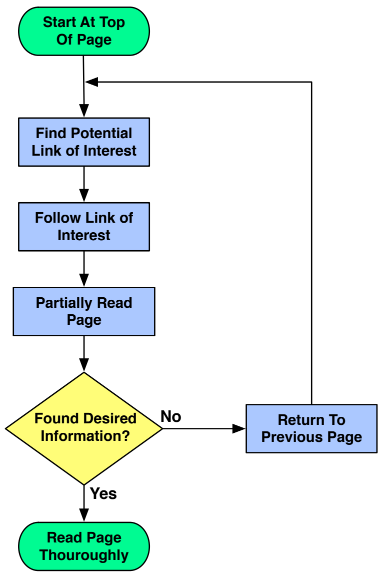


Figure 4.6: The Probing Strategy allows users to look for candidate links and select the ones that they think will take them to a desired page.

Now" but the user was unsure about whether to go to that page as they wanted to see flight costs and times and not book a flight. With no other potential links to follow, the user clicked on the *"Book Now"* link and found the information they desired. The pattern users employ when using the Probing Strategy is abstractly defined and shown in Figure 4.6 on page 110.

4.4.5 Backtracking Strategy

When browsing Web pages, users occasionally get lost within the page. This can occur either because the user clicked a wrong link or because the page at the destination of a link did not contain information that was expected. The Backtracking Strategy is used when users want to recover from states where they are lost within the Web. Instead of going backwards to the point at which the confusion began, users will go back to a safe state, typically the search page, and retrace their steps. This occurs until the user reaches the point at which they became lost and then a different decision is taken. For example, in the NoVA evaluation data this strategy was observed in Test 19 where the user was using AltaVista to find search results. The user was trying to find links to the UK weather and clicked on a link they thought was appropriate. On the new page,

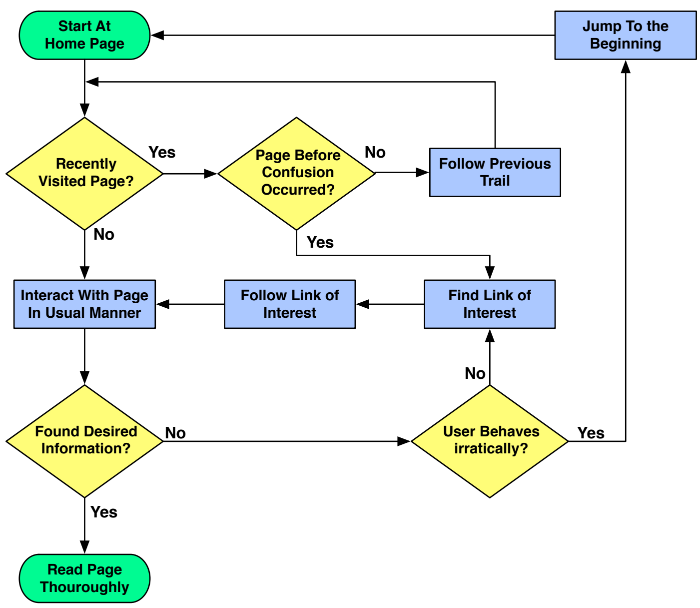


Figure 4.7: The Backtracking Strategy is used when users want to recover from states where they are lost within the Web.

the user had to use a combo box to choose the location. By accident the user chose Africa. Rather than go backwards one page to select a different location, the user returned to the AltaVista home page and retraced some of their steps. The pattern users employ when using the Backtracking Strategy is abstractly defined and shown in Figure 4.7 on page 111.

4.4.6 Withdrawal Strategy

There are occasions that users become so confused within a Web page that they can no longer cope with the frustration and stress of browsing the Web. Under such circumstances, users typically have three responses:

1. Become agitated and vent their frustrations through emotional and physical abuse;
2. Resign themselves to not being able to succeed in their task and give up;
3. Ask for help.

For example, in the first Henshaws' study, this strategy was observed in Session 7 when a user was entering their personal details in an online job application form. The user's details were lost during the session, possibly due to a time out

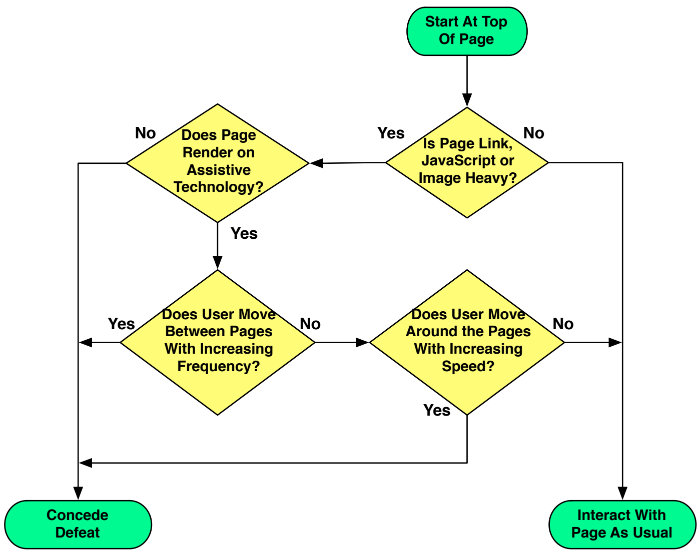


Figure 4.8: The Withdrawal Strategy is used when users can no longer cope with the frustration and stress of browsing the Web.

feature of the Website, and as the user was unable to retrieve the details, they physically abused the keyboard of the computer. The pattern users employ when using the Withdrawal Strategy is abstractly defined and shown in Figure 4.8 on page 112.

4.5 Coping Strategy Based Transcoding

From the patterns identified in Section 4.4, a number of transcodes that would assist visually impaired users' access to Web content emerged. These transcodes directly addressed the Candidate Chunk Discovery Strategy, Masthead Avoidance Strategy, Probing Strategy, and Backtracking Strategy. Furthermore, the Withdrawal Strategy was indirectly addressed as one could assume that providing transcodes to support users' access to content would reduce the number of times that they "gave up" completing a task. The transcodes therefore had the broadest benefit to users when implemented as they directly supported twenty, and indirectly support ten, of the thirty-three coping strategies that were categorised. They therefore acted as a good test of the behaviour-driven transcoding approach.

Provide Access To Important Chunks: In both the Candidate Chunk Discovery Strategy and Masthead Avoidance Strategy, users attempted to quickly and easily reach the main content of the page. Identifying the main chunks of information and providing a consistent manner in which to access them would aid users as they arrived at a page and interacted with the content.

Provide a Summary of the Page: In the Probing Strategy, users navigated to a page that they believed contained information of interest. However, due to poor link descriptions, this could not be certain. Providing a summary of the content of the page would facilitate the user in deciding if the page contained information of interest. If it did, then the user could interact with the page or return to the previous page and try alternative links.

Provide Access to the Menu: From the Probing Strategy, users who decided that the page did not contain the information they required often had difficulty returning to the menu to return back to previous pages. Providing a method to easily reach the menu would aid in the navigation of the Website. Furthermore, the Backtracking Strategy occurred when users became so lost in the page that they had to start again from scratch. Providing a mechanism the reach the menu and move from page to page within the Website more easily would alleviate this requirement.

To evaluate the effectiveness of the behaviour-driven transcoding, the SADIe experimental prototype, described in Section 3, was modified. Using the AxsJAX adaptations described in Section 3.3.2, ARIA terms were inserted into the page to allow users to interact with the content with a consistent set of key presses in a manner similar to an application.

To support *Provide Access To Important Chunks*, users pressed the `n` key to move the focus of the screen reader to the next most important chunk. Further presses of `n` skipped the focus to the next chunk, allowing users to easily skim through the text in order to gain an insight into the structure of the page and reach the important content more quickly. Pressing the `p` key reversed the process allowing users to go to the previous chunk. Using the two-part ontology annotation method, important chunks were identified from the property ∃ `SADIe:hasPriorityValue some SADIe:Thirteen`. Using the extended CSS approach to annotation, important chunks were identified from the property `-uom-structural-role: Chunk`. In both annotation models, areas of content defined as being the part of the page that contained the most important information was based upon the judgement of the annotator to correctly identify what they considered to be the most useful content.

To support *Provide Access to the Menu*, users pressed `m` to move the focus of the page to the menu. This moved the screen reader to the first menu item, regardless of its location within the page, allowing for easier navigation between pages within the Website. Using the two-part ontology annotation method, menus were identified from the property ∃ `SADIe:isMainMenu some SADIe:True`. Using the extended CSS approach to annotation, menus were identified from the property `-uom-structural-role: LinkList`.

To support *Provide a Summary of the Page*, users pressed `s` to receive a summary of the page content. Summaries could be generated by a specific chunk of text within the page that provided an overview, as is the case with CNN (see the area indicated as 2 in Figure 3.3 on page 70). Summaries were identified from the SADIe CSS annotation through the property `-uom-structural-role:`

`PageSummary`. There was no equivalent property in the two-part ontology approach. Alternatively, if no `PageSummary` annotation could be found, the `<meta name="Description" content="..." />` tag could be used. This provides a summary for search engines and is typically hidden from the user as it resides in the `<head>` element of the HTML document. However, this tag was utilised to inform the user what the page was about without them having to interact with the content. Summary read aloud the overview of the page without the user losing the current focus of the screen reader, allowing users to move around the page and still gain and overview without loosing their position.

For a full discussion of how the two-part ontology and the extended CSS were used to annotate Web content, see Section 3.2. An example ontology and extended CSS for the CNN Website can be found in Appendix B on page 176 and Appendix C on page 180 respectively.

4.6 Summary and Conclusions

In this chapter we have discussed how user interaction techniques on Websites can be identified through a framework of coping defined from a series of summative experimental studies. Using thematic analysis techniques the data was analysed to identify seven factors that can be used to define coping strategies employed when access to Web content becomes too taxing. This framework was validated using two additional data sets and demonstrated that the definition was flexible and accurate enough to be applicable to multiple data sources. The framework derived from the study, and validated with external data, supports Research Question 2 by showing that it is possible to identify coping strategies in a consistent manner as users interact with the Web.

We have also seen how using the coping strategies identified from the two Henshaws' studies and the NoVA evaluation data allowed six abstract patterns to be extracted. These patterns, identified as Candidate Chunk Discovery Strategy; Masthead Avoidance Strategy; Clustered Element Strategy; Probing Strategy; Backtracking Strategy; and Withdrawal Strategy, allowed for the development of behaviour-driven transcoding functionality. The transcoding identified transformed static Web content into interactive content by allowing users to navigate between key elements of the page through a consistent set of key presses. The development of the transcoding from the abstract coping strategy patterns supported Research Question 3 by demonstrating that identified coping strategies can be used to define transcoding solutions. These solutions were then incorporated into the SADIe experimental transcoding proxy to form the basis of the transcoding user evaluation.

Chapter 5

Transcoding Evaluation

This chapter describes how the structure-driven transcoding and the behaviour-driven transcoding, derived from the coping strategies in Section 4, were evaluated with visually impaired users in order to support Research Question 4. The initial evaluations compared the effects of structure-driven transcoding to the original version of the Website, which demonstrated that the structure-driven approach did benefit users. However during a pilot evaluation of the behaviour-driven transcoding, it became apparent that ARIA was not fully supported by common commercial screen readers. The original structure-driven transcoding evaluation allowed participants to use their own equipment. For behaviour-driven transcoding, not only would the transcoding change but also the equipment, which would not allow for a fair comparison of results. It would be difficult to establish if any differences in task completion time was attributable to behaviour-driven transcoding or the change in equipment. Therefore the structure-driven transcoding evaluation had to be repeated on the open source Firevox. This is an extension to Mozilla Firefox that converts the Web browser into a self-voicing application and supports ARIA. The results of the repeated study mirrored the results of the original study and demonstrated that the SADIe structure-driven transcoding reduced the time it took users to access information on Web pages. The behaviour-driven transcoding evaluation also revealed reduced task completion times, supporting the use of behaviour-driven transcoding as a method of improving access to content for visually impaired users.

5.1 Evaluation Setup

Quantitative evaluations involve the acquisition and analysis of measurable data that is obtained from the user as they interact with a Website. We opted to perform a quantitative evaluation as the outcome of such studies can generally be analysed using statistical techniques, allowing us to judge if anything significant occurred. In the case of SADIe, we wanted to establish if SADIe transcoded pages significantly reduced the time it took users to access information on a Web page. Having decided on a quantitative measure, a choice between performance measures or logging user actions had to be made. Logging user actions allows the evaluator to capture a continuous stream of quantitative data on tasks that are performed “in the wild”. Capturing data logs is particularly useful as it allows for the automatic acquisition of large amounts of real usage data, from multiple users, over a long period of time [Nielsen 2004b]. However, log data analysis faces the problem of mapping low level system usage to high level user tasks [Ivory & Hearst 2001]. Evaluators can understand what users did when interacting with a Website, but cannot necessarily understand why they did it, and if they achieved their goal [Nielsen 2004b].

Based upon this, we opted for a performance measure task whereby an evaluator was present. This ensured that the user was clear in the task that they were being asked to perform and also ensured that the number of confounding variables was reduced. With remote logging the user could stop during the evaluation and it would be unclear if this was due to the task being too difficult or the user taking a break. With an evaluator present, these concerns could be reduced and the results would be a more accurate reflection of the times taken to perform tasks. In addition, with the users and evaluators together, there was an opportunity for discussion and feedback after the session to try and understand why users had performed tasks in certain ways and what the user found difficult and easy, both with the original and the SADIe transcoded pages.

With this in mind the evaluations were based upon the following hypothesis:

H0: The time it takes to complete a fact-based task on a Web page is the same regardless of which version of the Web page is used.

H1: The time it takes to complete a fact-based task on a Web page using SADIe structure-driven transcoding is less than the time it takes to complete a fact-based task using the original version of the Web page.

H2: The time it takes to complete a fact-based task on a Web page using SADIe behaviour-driven transcoding is less than the time it takes to complete a fact-based task using the SADIe structure-driven transcoding version of the Web page.

In total four studies were conducted. The first study, labelled as *Evaluation 1: Structure-Driven Transcoding*, involved visually impaired users conducting tasks on both original versions of the page and versions that had been transformed using known structural-driven techniques. The second study, labelled as *Evaluation 2: Sighted User*, involved sighted users completing tasks on the same pages from Evaluation 1 in order to compare the difference between visually impaired and sighted users. The third study, labelled as *Evaluation 3: Structure-Driven Transcoding Repeat*, involved visually impaired users conducting tasks on both original versions of the page and versions that had been transformed using known structural-driven techniques but using the Firevox browser. This was due to incompatibilities between common commercial screen readers and ARIA, which was used to implement the behaviour-driven transcoding. The original structure-driven transcoding evaluation allowed participants to use their own equipment. For behaviour-driven transcoding, not only would the transcoding change but also the equipment, which would not allow for a fair comparison of results. It would be difficult to establish if any differences in task completion time was attributable to behaviour-driven transcoding or the change in equipment. Therefore the structure-driven transcoding evaluation had to be repeated on equipment that would be the same for both this study and the behaviour-driven transcoding study. The final study, labelled as *Evaluation 4: Behaviour-Driven Transcoding*, involved visually impaired users conducting tasks using Firevox to access pages that had been modified using the behaviour driven transcoding.

While all four studies are presented, it should be noted that only Evaluations 2, 3 and 4 are compared with each other in Section 5.2. This was because Evaluation 1 had a number of differences when compared to the other three studies including an older demographic (discussed further in Section 5.1.1), different study equipment (discussed further in Section 5.1.2), and a higher number of tasks (Section 5.1.3).

5.1.1 Participants

The evaluations required both visually impaired users who required a screen reader to access the computer and sighted users. Both groups were required to have some computer experience. Participants were recruited from Access SUMMIT[1]; the University of Manchester Disability Support Office[2]; University of Manchester research volunteer Web pages and mailing list[3]; and the University of Manchester's School of Computer Science. Potential participants were screened before the evaluation took place to ensure that they were suitable for the evaluations. A number of participants applied who, whilst visually impaired, solely used screen magnification software to access their computer. The visually impaired user group taking part in the study was limited to those people who used screen readers as this was the target audience of the SADIe transcoding features. As screen magnification and screen reading are different interaction paradigms, it was felt that combining the two groups in the evaluation would not yield fair comparisons.

The Websites chosen for the evaluation were English based. While nine of the participants were not native English speakers, they were sufficiently fluent in English to take part. While we did not conduct a language assessment ourselves, the participants in question were either students or staff at the University. Taking this into account, and conversations between the participants and the evaluators before the study took place, supported their fluency in English and their suitability to take part in the study. All the participants gave their informed consent to take part in the evaluation and were paid £10 for their time and effort.

Demographics of the users who participated in the four studies are given in Table 5.1 on page 121. All the participants used the Web daily, for a variety of reasons. Those who required assistive technology accessed the Web via a screen reader. It should be noted that the volunteers for Evaluation 1 were all over the age of 36, whereas for Evaluations 2, 3 and 4 there was a range of ages from 16–45. However, Evaluation 1 was not used when comparing the results between each evaluation group. The number of participants for each group is also relatively small, with only four visually impaired users in Evaluations 1, 3 and 4, and six sighted users in Evaluation 2. The small number reflects the

[1]URL: `http://www.access-summit.org.uk/`. Last accessed: 9th July 2008
[2]URL: `http://www.manchester.ac.uk/disability/`. Last accessed: 9th July 2008
[3]URL: `http://www.studentnet.manchester.ac.uk/volunteer/`. Last accessed: 9th July 2008

Table 5.1: Demographics of the participants that took part in the four transcoding evaluations.

Group	#	*Sex*	*Age*	*Job*	*Disability*	*Client*	*Use*
1	P1	Male	36–45	Lecturer	Profoundly Blind	JAWS	Daily
	P2	Female	36–45	Disability Trainer	Partially Sighted	JAWS & Zoom Text	Daily
	P3	Male	36–45	Student	Profoundly Blind	JAWS	Daily
	P4	Male	36–45	Student	Profoundly Blind	JAWS & Keynote Gold	Daily
2	P5	Male	16–25	Student	None	N/A	Daily
	P6	Female	26–35	Student	None	N/A	Daily
	P7	Male	26–35	Student	None	N/A	Daily
	P8	Male	16–25	Student	None	N/A	Daily
	P9	Male	16–25	Student	None	N/A	Daily
	P10	Male	26–35	Research Associate	None	N/A	Daily
3	P11	Male	36–45	Student	Profoundly Blind	JAWS	Daily
	P12	Female	16–25	Student	Partially Sighted	JAWS & Zoom Text	Few Times per Week
	P13	Female	16–25	Student	Profoundly Blind	JAWS	Daily
	P14	Male	36–45	Student	Profoundly Blind	JAWS	Daily
4	P15	Male	36–45	Student	Profoundly Blind	JAWS	Daily
	P16	Male	36–45	Student	Profoundly Blind	JAWS	Daily
	P17	Female	16–25	Student	Partially Sighted	JAWS & Zoom Text	Few Times a Week
	P18	Female	16–25	Student	Profoundly Blind	JAWS	Daily

difficulty that is faced when recruiting visually impaired users. The participants required for these studies needed to be visually impaired, have experience at using the Web, be competent using a screen reader, live in the local area and be willing to come into the usability lab to conduct the evaluation. These requirements reduced the number of potential recruits but is an issue that is faced by many studies that involve visually impaired users. Previous studies, for example, have been conducted with nine [King *et al.* 2004], five [Takagi *et al.* 2007], and three [Mahmud *et al.* 2007] visually impaired participants. Therefore, while our study lacks a large number of participants, it is not uncommon within the field and is an issue that needs to be addressed in the research community as a whole.

5.1.2 Equipment

For *Evaluation 1: Structure-Driven Transcoding*, participants performed the tasks using their own equipment in familiar surroundings. The reason for this was that the purpose of the evaluation was to study the effects of SADIe and it was felt that having the participant become accustomed to new surroundings and unfamiliar equipment was an unnecessary burden. All four participants used a laptop PC running Microsoft Windows XP[4]. The browser used was Microsoft Internet Explorer[5] and the assistive technology was JAWS screen reader[6]. For P1 the evaluation was conducted in their office. P2, P3 and P4 performed the evaluation in a study room at Access SUMMIT. All three participants were regular visitors to the Access SUMMIT offices and knew the people and the surroundings well.

For *Evaluation 2: Sighted User*, *Evaluation 3: Structure-Driven Transcoding Repeat*, and *Evaluation 4: Behaviour-Driven Transcoding*, both the location and equipment changed based on experience gained from running the first evaluation. All the participants were familiar with the University and many asked if the study could be conducted on campus. Therefore the location changed from a location of the participant's choice to the School of Computer Science Usability Lab.

In addition, participants were required to use our equipment in order for the behaviour-driven transcoding to function correctly. This was because during pilot studies of the behaviour-driven transcoding evaluation, it became apparent that ARIA was not fully supported by commercial screen readers such as JAWS. Therefore any participants of the behaviour-driven transcoding evaluation had to use Firevox. This is a free extension to the Mozilla Firefox Web browser that converts the browser into a self-voicing application that fully supports ARIA and AxsJAX code. This scenario raised the prospect that the behaviour-driven transcoding evaluation would have two variables: The transcoding of the page and equipment that users would not be familiar with. Therefore, the structure-driven transcoding evaluation was repeated, but with the Firevox self-voicing browser. This was to allow a fair comparison to be made with the results obtained from the behaviour-driven transcoding evaluation. As a result, all participants used equipment in the Usability Lab that consisted of a desktop PC running Microsoft

[4]URL: `http://www.microsoft.com/windows/`. Last accessed: 9th July 2008

[5]URL: `http://www.microsoft.com/windows/ie/`. Last accessed: 9th July 2008

[6]URL: `http://www.freedomscientific.com/jaws-hq.asp`. Last accessed: 9th November 2009

Windows XP. The sighted participants who took part in Evaluation 2 used Microsoft Internet Explorer Web browser and the visually impaired participants used Mozilla Firefox[7] Web browser coupled with Firevox[8].

To validate the results of the evaluation, the entire session was recorded using an audio tape recorder installed at the back of the room. The tape recorder was also used to record participant feedback and comments after the evaluation had been completed.

5.1.3 Tasks

During the evaluation users were asked to perform a series of tasks on Web pages. In selecting the tasks, there were several conflicting interests that needed to be addressed. The first was the number of pages used in the evaluation. Too many pages and the study would take too long resulting in the user becoming bored and frustrated, which would adversely affect the results. However too few pages and there would not be enough data to provide adequate results. Secondly, the tasks had to be comparable in terms of difficulty for the user. This could have been achieved by using a selection of pages from the same Website, assuming that the style and layout remained consistent and only the content changed. However, to accurately represent Web usage, a diverse range of Websites needed to be tested. Finally the user had to be familiar with the pages as this was an experiment that tested the hypothesis that SADIe improved the speed of access to content. At the same time, the content could not be so familiar that the user could guess the answer to the tasks.

For *Evaluation 1: Structure-Driven Transcoding*, the number of tasks used in the experiment was 20 as it was felt that this struck a balance between gathering enough data to perform statistical testing, yet not providing too many tasks that would frustrate the user and require too much time. While 20 pages may not seem like a representative sample of pages found on the Web, it would be impractical to build a corpus of Web pages which represents a greater number than this. According to Netcraft[9] there are approximately 150,000,000 Websites. Testing 10,000 Websites, rather than 20, means that only 0.007% more of the Web is tested, and this is not statistically significant. Indeed, we would need to

[7]URL: `http://www.mozilla.com/firefox/`. Last accessed: 10th November 2008

[8]URL: `http://firevox.clcworld.net/`. Last accessed: 10th November 2008

[9]URL: `http://news.netcraft.com/archives/web_server_survey.html`. Last accessed: 9th July 2008

test 1,500,000 Websites to achieve just a 1% testing. Given the impossibility of testing any significant proportion of the Web, the key thing is good sampling. A sample must reflect the population: major categories should be represented and this can be narrowed down further by choosing sites which are of a typical representation (sites with large numbers of visitors, and longevity).

The sample of pages used was taken from the top 100 visited Websites[10] so that popular, well used sites were being used in order to provide realistic Web usage for the experiment. The sample consisted of Websites that were written in English as this was the language used by the participants. The pages also contained predominantly text-based content due to the fact that the participants were visually impaired, therefore asking questions with regards to images was inappropriate and would adversely affect the experimental data. To ensure that users could not pre-empt the answer to the task, all the Websites provided content that was updated frequently, for example news pages, search engine results and blogs. This helped reduce the familiarity problem as the user would be familiar with most of the sites chosen but, as the content often changed, they would not be able to guess the answer to the task. Finally the page that was used in the experiment was from the entry page to the site where appropriate so that a fair comparison was made. For search engine tasks the entry point was not appropriate as the front pages were so familiar that the user would have been able to provide an answer to the question immediately. Instead the first search results page was used. A full list of the Web pages used as stimuli in the study and the tasks that users were asked to perform on the page is provided in Table J.1 in Appendix J on page 199. It should be noted that to ensure that the answers to the questions were not solely the opinion of the evaluator, a sighted user was asked to come into the lab and complete all twenty tasks to ensure that the answers they provided matched those that the evaluator expected from the participants when the evaluation was run.

The Web pages used as stimuli in the study were saved onto a USB memory stick and accessed locally through the Web browser during the evaluation. This ensured that the Web pages did not change between evaluation sessions and allowed the tasks to be taken to the participant's own computer as Evaluation 1 allowed participants to use their own equipment.

When Evaluation 1 was conducted (see Section 5.2.1), only one participant

[10]URL: `http://www.alexa.com/`. Last accessed: 7th September 2008

completed all 20 tasks, with the remaining participants completing between 6–16. Therefore the number of tasks for *Evaluation 2: Sighted User*, *Evaluation 3: Structure-Driven Transcoding Repeat*, and *Evaluation 4: Behaviour-Driven Transcoding*, was reduced to eight. These were *About*, *BBC News*, *CNN*, *Google Search*, *New York Times*, *Wordpress*, *Xanga*, and *Yahoo Search*. These were chosen as a comparison could be made between these eight websites from Evaluation 1 and they also contained diverse content, such as blogs, news stories, search results and portals. The same page and tasks from Evaluation 1 were used.

The pages used in Evaluations 1 and 3 were transcoded using the structure-driven transcoding while the pages used in Evaluations 4 were transcoded using the behaviour-driven transcoding. The annotations that drove the transcoding were completed manually in the same fashion that was described in Section 3.2 and Section 4.5. Therefore what constituted a menu and the most important information on the page was the judgement of the annotator, which had the potential to be different from that of the user.

5.1.4 Procedure

For *Evaluation 1: Structure-Driven Transcoding*, the Web pages were randomly ordered in two ways. The first was the order in which the pages were displayed to the participant. The second was the pages that changed from those that were transcoded with SADIe and those that were left in the original state. A page was only used once per participant. Therefore a user either interacted with the original version of a page or the transcoded version. A participant never interacted with both. To ensure that participants interacted with the pages in a consistent manner, a task page was created. This was a simple HTML document that had a list of links, labelled “Page 1” through “Page 20” inclusive, pointing to the twenty tasks that had been created. For each participant, the tasks were randomised so that they did not perform the tasks in the same order. However, after reordering, the tasks were renamed “Page 1” through “Page 20” so that from the perspective of the user, they were performing a sequence of twenty tasks.

For *Evaluation 2: Sighted User*, the tasks were randomised so that users did not perform the tasks in the same order. As with Evaluation 1,the tasks were renamed, “Page 1” through “Page 8”, so that from the perspective of the user, they were performing a sequence of eight tasks.

Evaluation 3: Structure-Driven Transcoding Repeat required two parts to the procedure. The first part of the evaluation was designed to allow users to become familiar with the new screen reader commands. Firevox had not been used by any participants previously and so time was allocated to allow users to become accustomed to the different keyboard commands and different voice used by the browser. There was no time limit placed on this task. The evaluator guided the participant as they used the Firevox commands until the participant was comfortable with the new equipment. For the second part, the Web pages were randomly ordered in the same fashion as Evaluation 1. Both the order of the tasks and the page that was displayed (transcoded or original) were randomised. For each participant, the tasks were randomised so that they did not perform the tasks in the same order. However, after reordering, the tasks were renamed "Page 1" through "Page 8" so that from the perspective of the user, they were performing a sequence of eight tasks.

Evaluation 4: Behaviour-Driven Transcoding required two parts to the procedure. As with Evaluation 3, the first part of the evaluation was designed to allow users to become familiar with the new Firevox screen reader commands. There was no time limit placed on this task. The evaluator guided the participant as they used the Firevox commands until the participant was comfortable with the new equipment. For the second part, the eight behaviour-driven transcoded pages were randomised so that participants did not perform the tasks in the same order. After reordering,the tasks were renamed "Page 1" through "Page 8" so that from the perspective of the user, they were performing a sequence of eight tasks.

To avoid an element of bias being introduced into the results, all evaluations were conducted as a double-blind experiment. The evaluator had not been involved in the design of the experiment and was not conducting work on the SADIe project. At the start of the study the evaluator read the information sheet to the participant and answered any questions that the user may have had with regards to the experiment. The user then sat in front of their computer with the task page loaded in the Web browser. The evaluator gave the participant instructions for the task for Page 1. When the participant was ready, they clicked on the link that took them to the page required to perform that task. The timer started as soon as the participant had clicked the link. The time stopped when the participant had completed the task. The participant was then allowed to return back to the main page in their own time. When they indicated that they were ready, the evaluator gave instructions for the next task and the process was repeated. It should be noted that the user was not aware of which version of the page they would be interacting with. The user was merely given a task and asked to complete it.

5.2 Results

The result all four studies are presented below. Both qualitative analysis using t-tests and quantitative observations were made to determine the success of the transcoding systems. All three of the evaluations that were conducted with visually impaired users shown that the transcoding can improve access to the content of Web pages. However, while the results are encouraging, a number of factors should be addressed in future investigations. The first factor is the annotations of the pages. As shall be discussed in Evaluation 4 in Section 5.2.4, some of the users had difficulty completing the tasks due to not being able to easily reach specific parts of the content. Furthermore, the number of participants was small so that any variance in the results would be greatly magnified. However, the results, comments and observations from the evaluations were positive and support the use of behaviour-driven transcoding as a method of assisting visually impaired users access Web content.

5.2.1 Evaluation 1: Structure-Driven Transcoding

Table 5.2 on page 128 shows the results of the evaluation for all four participants including the average completion time per page for both versions. The cells of the table that are greyed out indicate that that version of the page was not part of the evaluation. As discussed in Section 5.1.4 participants only performed the task on one version of the page, never both. The symbols *(s)* and *(f)* indicate if the user successfully completed the task or if the user failed to complete the task respectively. *N/A* indicates where a task was not attempted by the user.

Figure 5.1 on page 129 shows a comparison between the average time taken to complete the task on both the original and transcoded versions of the page. Not all participants completed the tasks, therefore only comparisons of the eight

Table 5.2: Results of the structure-driven evaluation for all pages. Greyed out cells indicate this version of the page was not part of the study. *(s)* indicates successful task completion. *(f)* indicates failed task completion. *N/A* indicates task not attempted. The times are given in seconds.

	Original					***SADIe***				
Page	***P1***	***P2***	***P3***	***P4***	***Mean***	***P1***	***P2***	***P3***	***P4***	***Mean***
About	117 *(s)*	179 *(s)*	N/A		148				57 *(s)*	57
AOL	17 *(s)*	55 *(s)*		114 *(s)*	62			N/A		N/A
BBC News	43 *(s)*				43		129 *(s)*	N/A	43 *(s)*	86
Blogger			N/A	N/A	N/A	46 *(s)*	23 *(s)*			34.5
CNET	36 *(s)*	424 *(f)*	N/A	N/A	230					N/A
CNN		51 *(s)*	N/A	N/A	51	09 *(s)*				9
Digg	34 *(s)*	343 *(s)*	N/A		219.5				N/A	N/A
Go	24 *(s)*	182 *(s)*	N/A	N/A	103					N/A
Google		424 *(f)*			424	26 *(s)*		N/A	117 *(s)*	71.5
Information	24 *(s)*	54 *(s)*	344 *(f)*	N/A	130.666					N/A
Live Search	109 *(s)*	N/A	285 *(s)*	50 *(s)*	50					N/A
Microsoft	24 *(s)*	N/A	N/A		24				N/A	N/A
MSN		N/A			N/A	10 *(s)*		201 *(s)*	279 *(f)*	163.33
New York Times	25 *(s)*	333 *(f)*		N/A	179			52 *(s)*		52
Rediff	133 *(s)*	147 *(f)*		N/A	220			N/A		N/A
Reference	31 *(s)*	190 *(s)*	N/A	N/A	110.5					N/A
Slashdot	11 *(s)*	N/A	238 *(f)*	237 *(f)*	162					N/A
Wordpress	26 *(s)*		N/A	N/A	26		87 *(s)*			87
Xanga	216 *(s)*		436 *(f)*	N/A	315		58 *(s)*			58
Yahoo! Search			N/A	120 *(f)*	120	10 *(s)*	16 *(s)*			13

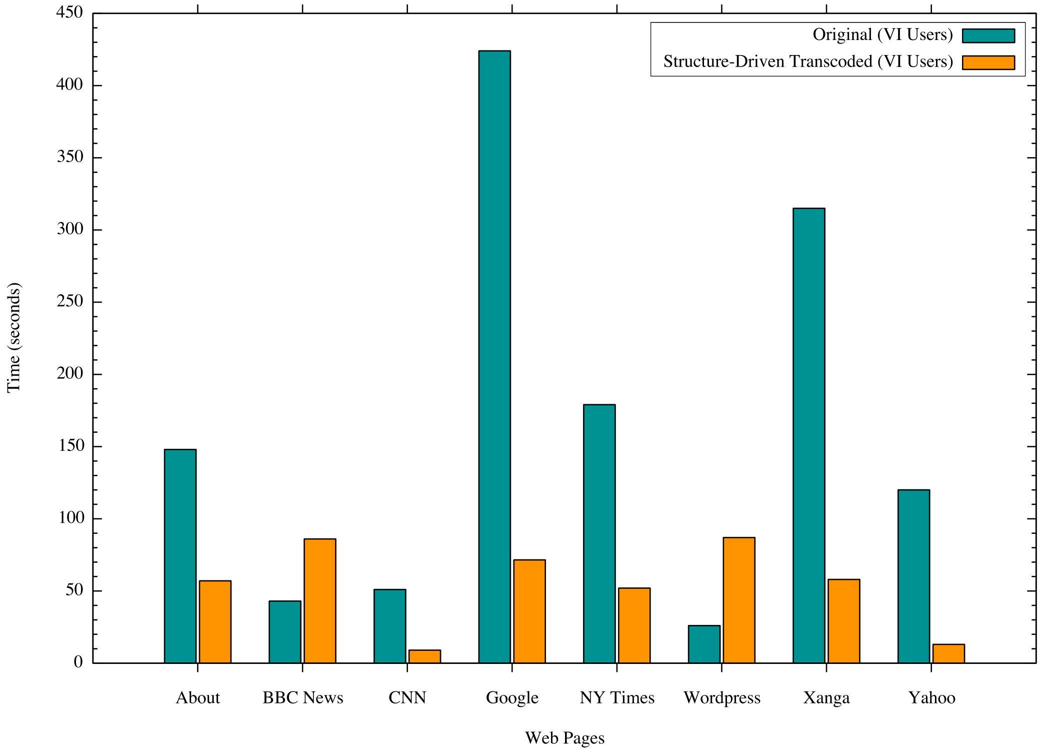


Figure 5.1: Comparison of the average time to find information on a page using SADIe structure-driven transcoding and the average time to find information on a page using the original version.

pages where tasks were performed on both the original and transcoded version of the pages by at least one user are presented. Where a task was attempted but not completed, we use the time at which the user gave up and moved onto the next task. For the original version of the page, there were four instances of the tasks not being completed and for the structure-driven transcoded versions, there were no instances where the user failed to complete a task. On average, participants took longer to complete tasks with the original Web pages ($mean$ = 163.25, $standard\ error$ = 49.87) than they did to complete tasks using the SADIe structure-driven transcoded pages ($mean$ = 54.19, $standard\ error$ = 10.48). Analysing the results using the t-test gave a value of $t(7.6) = -2.14$, with $p = 0.033$. As $p < 0.05$, the result was significant which supported hypothesis H1 that the time taken to find information within a Web page is reduced when accessing the content through SADIe structure-driven transcoding. A box plot illustrating the average task completion times for both transcoded and original versions of Web pages can be seen in Figure 5.2 on page 130.

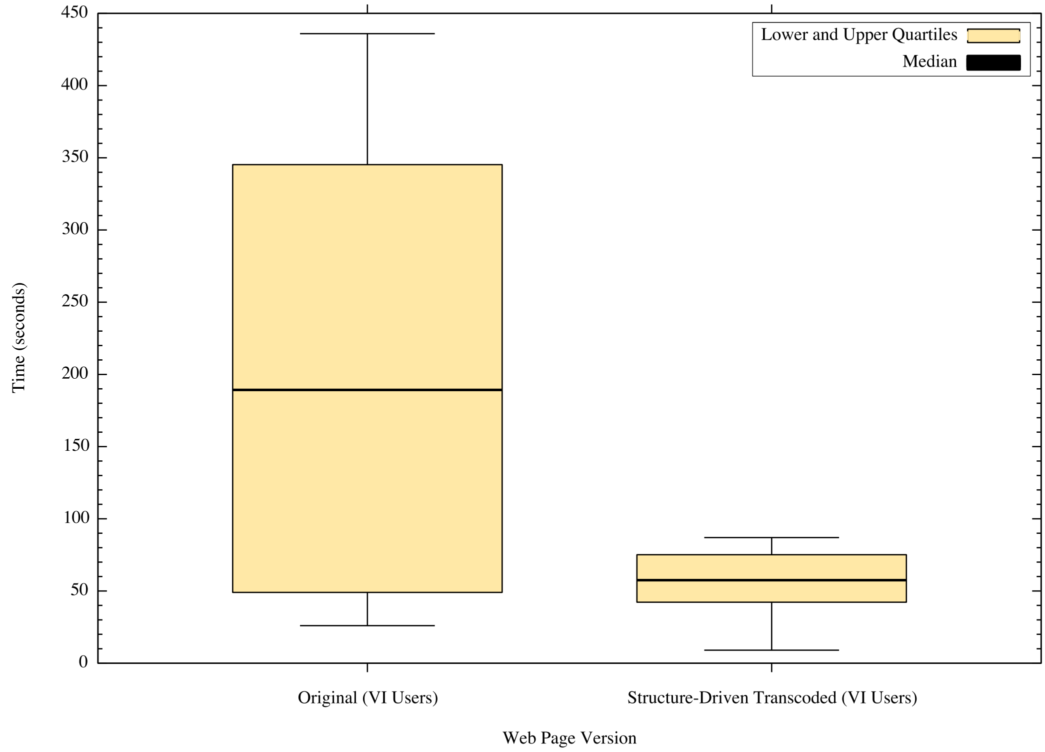


Figure 5.2: Comparison of the average time to find information on a page using SADIe structure-driven transcoding and the average time to find information on a page using the original version.

With the exception of the BBC News Web page and the Wordpress Blog page, the structure-driven transcoding decreased the average time it took users to find information within the Web pages. One reason for the transcoding increasing task completion time might be due to the layout and structure of the page changing which therefore made the page different from what participants expected. When accessing the transcoded version of the BBC News page, P2 for example, actually reached the main headline story within three seconds, but as they were not expecting the heading to appear so quickly, rapidly pressed `tab` to move down the page. On the BBC News Website, the main news is followed by news grouped by world regions. P2 was aware of this and it was not until they heard the phrase “Africa” that they started to `tab` backwards as they realised that they had gone too far. The same was also true for the Wordpress task. P2 had difficulty distinguishing between the blog owner’s name and the title of the latest entry of the blog after transcoding had taken place.

P3 also touched on the issue of structure during a post-evaluation discussion.

P3 commented that during the MSN task they would have preferred to have "Entertainment" has a heading followed by a numbered list of links to the entertainment stories. That way the participant could find the stories and know how many stories there were. When the participant wanted to read a story, then they could click the link and be taken only to that story and gain immediate access to it.

P1, when discussing search engine results after the session had finished, stated that *"if you told me these were the results, such as a heading stating 'The results are here' then that would help me."* This claim was supported by P3 who also commented that *"headings should proclaim what is coming"*. This is consistent with how P3 interacted with and navigated around the Web page. P3 relied heavily on headings, using functionality provided in JAWS to skip between heading sections of the page. In most cases SADIe assisted this functionality as the only headings left were those relevant to the task. However, if the user was familiar with the page, then the change in heading structure, caused by the transcoding, could be confusing,

5.2.2 Evaluation 2: Sighted User

Table 5.3 on page 132 shows the results of the evaluation for all six participants including the average completion time per page. It is interesting to note that the completion time for sighted users is relatively consistent across all pages with a standard deviation of 3.76 between task completion times. Visually impaired users however, completed tasks with a much higher standard deviation. In Evaluation 1, the standard deviation between the original pages was 106.92 and for the structure-driven transcoded pages it was 44.29. While one should be wary of the different demographics of the two groups and the different equipment that was used, it still provides an interesting insight. One reason for this difference might be the quality of the underlying markup of the pages. Some pages, such as the Wordpress blog, use well defined heading levels to structure the page and guide users through the content. Google, on the other hand, makes use of tables to define the layout of the content which hinders visually impaired users as the try and access the search results. Visually, however, both pages provide a layout that guides users to the main areas of the page's content.

Table 5.3: Results of the sighted user evaluation for all pages. *(s)* indicates successful task completion. *(f)* indicates failed task completion. *N/A* indicates task not attempted. The times are given in seconds.

Page	***P5***	***P6***	***P7***	***P8***	***P9***	***P10***	***Mean***
About	18 *(s)*	23 *(s)*	8 *(s)*	13 *(s)*	5 *(s)*	6 *(s)*	12.167
BBC News	7 *(s)*	18 *(s)*	6 *(s)*	4 *(s)*	3 *(s)*	3 *(s)*	6.833
CNN	6 *(s)*	9 *(s)*	11 *(s)*	8 *(s)*	4 *(s)*	10 *(s)*	8
Google	6 *(s)*	8 *(s)*	6 *(s)*	3 *(s)*	4 *(s)*	3 *(s)*	5
New York Times	11 *(s)*	7 *(s)*	13 *(s)*	8 *(s)*	5 *(s)*	9 *(s)*	8.833
Wordpress	4 *(s)*	4 *(s)*	6 *(s)*	3 *(s)*	2 *(s)*	4 *(s)*	3.833
Xanga	7 *(s)*	35*(f)*	6 *(s)*	6 *(s)*	4 *(s)*	30 *(s)*	14.667
Yahoo! Search	4 *(s)*	9 *(s)*	6 *(s)*	3 *(s)*	4 *(s)*	4 *(s)*	5

5.2.3 Evaluation 3: Structure-Driven Transcoding Repeat

Table 5.4 on page 133 shows the results of the evaluation for all four participants including the average completion time per page for both versions. The cells of the table that are greyed out indicate that that version of the page was not part of the evaluation. As discussed in Section 5.1.4 participants only performed the task on one version of the page, never both. The symbols *(s)* and *(f)* indicate if the user successfully completed the task or if the user failed to complete the task respectively. Some users attempted a task twice, which is shown by the two values in the same cell. *N/A* indicates where a task was not attempted by the user.

The symbol *(p)* indicates that this is the penalised time for failing to complete the task. In the original study, users failed to complete four tasks. However, the tasks that they failed had times that were considerably higher than all the successfully completed tasks and so no adjustment was made to the data. In the repeated study, some users failed a task and gave up in a time that was less than some of the successful times. For example P4 told the evaluator that they wanted to stop the Wordpress task at 415 seconds. However, P3 continued with the Yahoo! Search task for 528 seconds before succeeding. While failure is generally a negative result, using the raw data from the study resulted in failure to complete a task having a positive effect on the results. To rectify this, the times of the tasks were adjusted so that tasks that users failed to complete were penalised and the value raised to the maximum recorded time a user spent completing the task.

Figure 5.3 on page 134 shows a comparison between the average time taken

Table 5.4: Results of the repeated structure-driven evaluation for all pages using Firevox. Greyed out cells indicate this version of the page was not part of the study. *(s)* indicates successful task completion. *(f)* indicates failed task completion. *(p)* indicates that this is the penalised time for failing. *N/A* indicates task not attempted. Where there are two values in the cell, this indicates the participant asked to return to the task to try again later. The times are given in seconds.

	Original					***SADIe***				
Page	***P11***	***P12***	***P13***	***P14***	***Mean***	***P11***	***P12***	***P13***	***P14***	***Mean***
About	86 *(s)*			23 *(s)*	54.5		15 *(s)*	26 *(s)*		20.5
BBC News	172 *(s)*			509 *(f)* 528 *(p)*	350		92 *(s)*	80 *(s)*		86
CNN	264 *(s)*		33 *(s)*		148.5		23 *(s)*		56 *(s)*	39.5
Google	436 *(f)* 352 *(s)*		390 *(s)*		371		116 *(s)*		237 *(f)* 313 *(p)*	214.5
New York Times		145 *(s)*		297 *(s)*	221	313 *(s)*		33 *(s)*		173
Wordpress		411 *(s)*		415 *(f)* 528 *(p)*	469.5	581 *(f)* 43 *(s)*		120 *(s)*		81.5
Xanga		356 *(s)*	39 *(s)*		197.5	49 *(s)*			87 *(s)*	68
Yahoo! Search		142 *(s)*	528 *(s)*		335	20 *(s)*			124 *(s)*	72

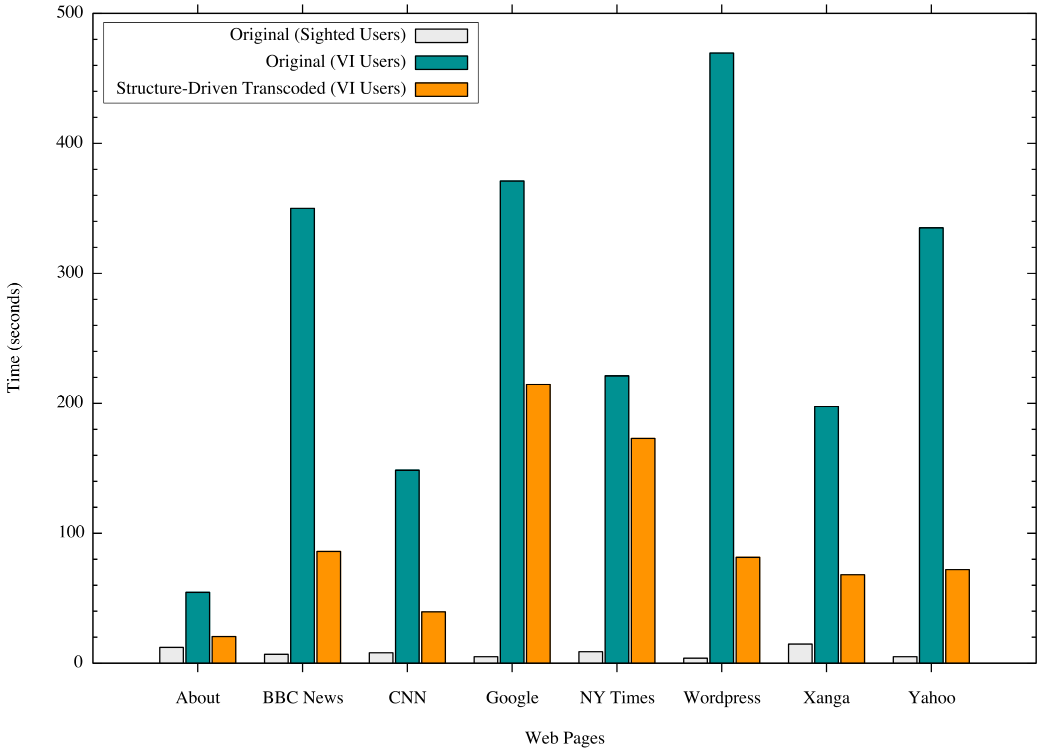


Figure 5.3: Comparison of the average time to find information on a page using SADIe structure-driven transcoding and the average time to find information on a page using the original version using Firevox.

to complete the task on both the original and transcoded versions of the page. The results shown include the adjusted values that penalise failure. The graph shows that the tasks were completed quicker using the SADIe structure-driven transcoding than the original version for all pages. The graph also shows the results of *Evaluation 2: Sighted User*, described in Section 5.2.2, to highlight that while SADIe can improve access to content, work still needs to be conducted to achieve access rates comparable to sighted users.

On average, participants took longer to complete tasks with the original Web pages ($mean = 268.375$, $standard\ error = 48.08$) than they did to complete tasks using the SADIe transcoded pages ($mean = 94.375$, $standard\ error = 23.34$). Analysing the results using the t-test gave a value of $t(14) = 3.255$, with $p = 0.003$. As $p < 0.05$, the result was significant and supported our hypothesis H1 that the time taken to find information within a Web page is reduced when accessing the content through SADIe transcoding. A box plot illustrating the average task completion times for both transcoded and original versions of Web

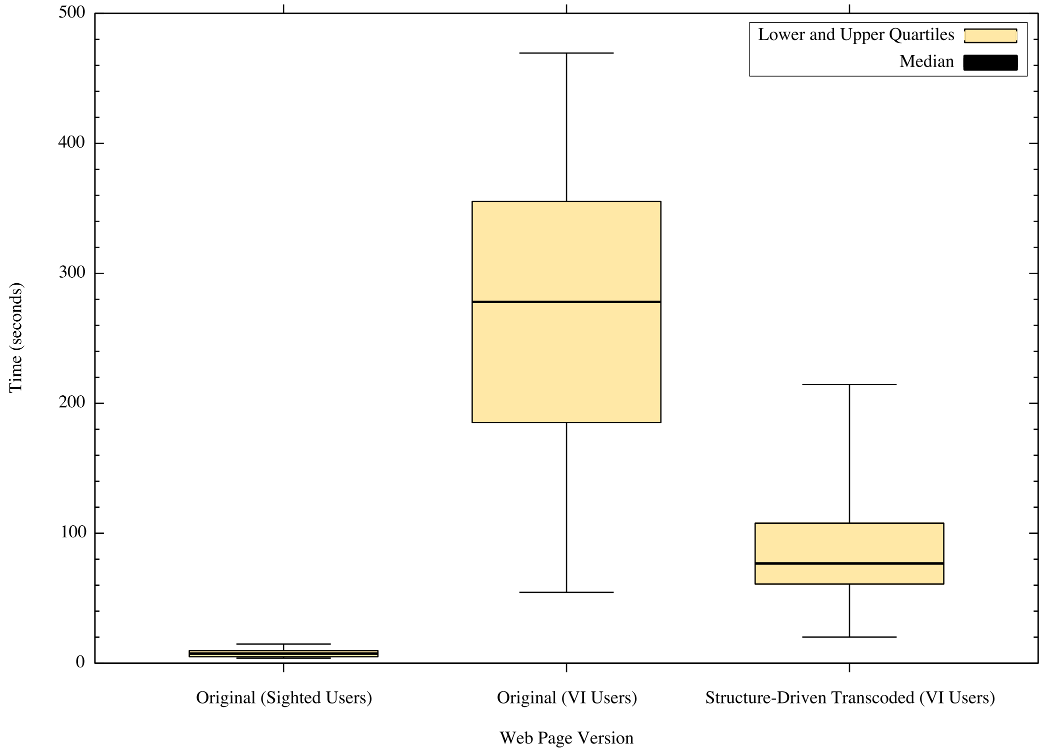


Figure 5.4: Comparison of the average time to find information on a page using SADIe structure-driven transcoding and the average time to find information on a page using the original version using Firevox.

pages can be seen in Figure 5.4 on page 135.

It is interesting to note that unlike Evaluation 1, SADIe improved access times for all pages. This may be a result of users having to use unfamiliar equipment. In Evaluation 1, users were familiar with the short cuts and options that their assistive technology could provide. For example, P2 was comfortable rapidly pressing `tab` to move between sections as they knew how their screen reader would behave. Indeed this is what resulted in P2 missing the answer to the task because they tabbed too far after the page had been transcoded. Even though Firevox provided these functionalities, all four participants were more tentative when interacting with the page as they were unfamiliar with the tool. As such, participants spoke aloud what they were doing, as though they were seeking assurance. For example, during the New York Times task, P14 said *"OK. Let me see. List of headings..."* as they were performing the task. During Evaluation 1, no participants spoke aloud the functionality. They only commented on the page in general or to provide the answer to the task. As users were not familiar with

Firevox, they were not reliant upon the structure of the page or the functionality that would allow them to exploit the structure to navigate through the content. As a result, participants accessed each page as though they were unfamiliar with the content and as SADIe transcoded the content so that the most important information was readily available, the participants completed the SADIe tasks more quickly.

It may be the case that one of SADIe's strengths lies in helping users interact with content that they are not familiar with. Users were unsure about interacting with the eight tasks as they were unfamiliar with the equipment and the SADIe transcodes appeared to help the users. Similarly, during Evaluation 1, two participants suggested that asking them to perform a task on a blog was unfair as they had never used a blog before. It was observed that during the task on Xanga, which involved a blog, P1 became very frustrated because they became lost in the page and unable to identify any elements or information. Indeed, P1 described Xanga as *"rubbish on the screen"*. When the same page was transcoded using SADIe, the participants who performed the task did not have any difficulty with the page. By restructuring the content, SADIe can assist users in reaching the main information on the page, even when users are not familiar with the content. SADIe ensures that content is consistently near the top and the menu is consistently near the bottom, which can be problematic if users are familiar with the page as their mental model of the content is broken. However, for unfamiliar pages, the rearrangement of content can be of benefit, although further investigation is required to validate this claim.

5.2.4 Evaluation 4: Behaviour-Driven Transcoding

Table 5.5 on page 137 shows the results of the evaluation for all four participants including the average completion time per page for both versions. The symbols *(s)* and *(f)* indicate if the user successfully completed the task or if the user failed to complete the task respectively. Some users attempted a task twice, which is shown by the two values in the same cell. *N/A* indicates where a task was not attempted by the user. The symbol *(p)* indicates that this is the penalised time for failing. This time was the maximum successful completion time in the same manner as the repeated structure-driven transcoding study.

Figure 5.5 on page 138 shows a comparison between the average time taken to complete the tasks on the original, structure-driven and behaviour-driven

Table 5.5: Results of the behaviour-driven transcoding evaluation for all participants. *(s)* indicates successful task completion. *(f)* indicates failed task completion. *(p)* indicates that this is the penalised time for failing. *N/A* indicates task not attempted. Where there are two values in the cell, this indicates the participant asked to return to the task to try again later. The times are given in seconds.

Page	***P15***	***P16***	***P17***	***P18***	***Mean***
About	8 *(s)*	23 *(s)*	21 *(s)*	92 *(s)*	36
BBC News	55 *(s)*	126 *(s)*	17 *(s)*	41 *(s)*	59.75
CNN	27 *(s)*	17 *(s)*	72 *(s)*	18 *(s)*	33.5
Google	27 *(s)*	63 *(s)*	21 *(s)*	99 *(f)* 11 *(s)*	30.5
New York Times	13 *(s)*	11 *(s)*	20 *(s)*	5 *(s)*	12.25
Wordpress	18 *(s)*	27 *(s)*	241 *(s)*	56 *(s)*	85.5
Xanga	88 *(s)*	197*(f)* 241 *(p)*	17 *(s)*	13 *(s)*	89.75
Yahoo! Search	25 *(s)*	31 *(s)*	6 *(s)*	208 *(f)* 57 *(s)*	29.75

transcoded versions of the page. For the behaviour-driven transcoding there was only one instance of a participant failing to complete a task, however this has been adjusted in the same manner as previous studies to allow a fair comparison to be made. On average, participants took longer to complete tasks with structure-driven transcoded Web pages ($mean = 94.375$, $standard\ error = 23.34$) than they did to complete tasks using behaviour-driven transcoded pages ($mean = 47.125$, $standard\ error = 9.95$). Analysing the results using the t-test gave a value of $t(9.466) = 1.862$, with $p = 0.047$. As $p < 0.05$, the result was significant and supported the hypothesis H2 that the time taken to find information within a Web page is reduced when accessing the content through behaviour-driven SADIe transcoding than structure-driven SADIe transcoding. A box plot illustrating the average task completion times for both transcoded and original versions of Web pages can be seen in Figure 5.6 on page 139.

Five of the eight tasks were completed quicker using the behaviour-driven transcoding than the structure-driven transcoding and all of the behaviour-driven transcoded pages were completed quicker than the original versions. One possible explanation for the failure of the three tasks to increase time over the structure-driven transcoding could be due to annotation of the page elements. Most pages had headings and text blocks separated, therefore users could press the `n` key to read the heading. A second press of the `n` key would then commence the

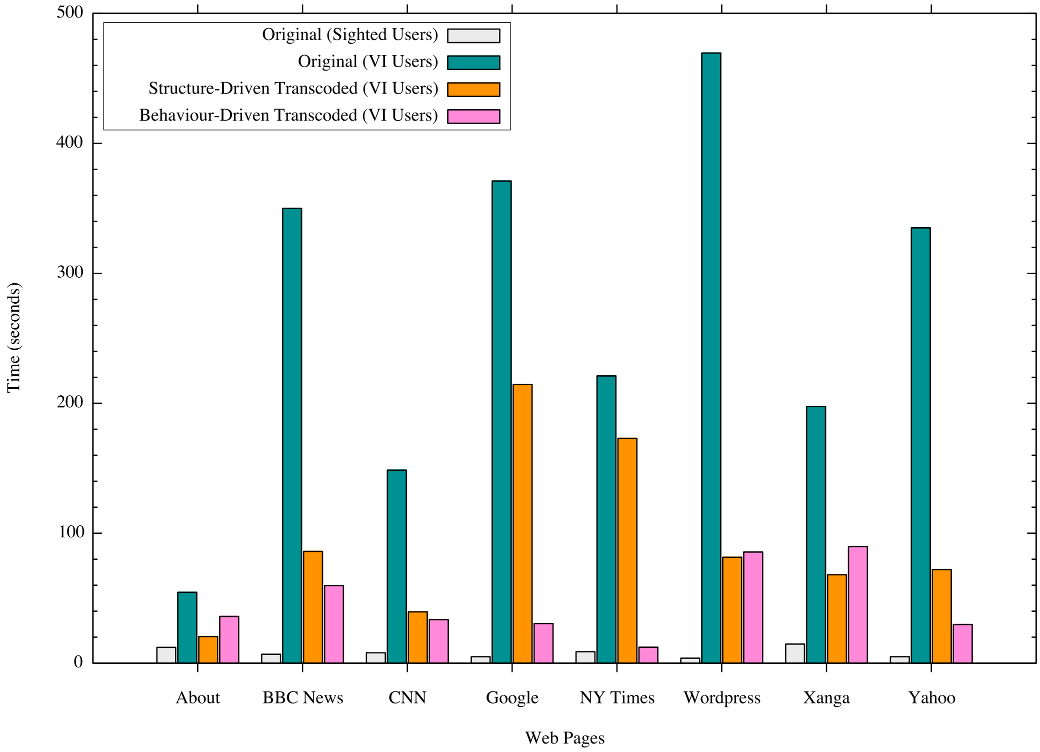


Figure 5.5: Comparison of the average time to find information on a page using SADIe behaviour-driven transcoding, SADIe structure-driven transcoding and the original version using Firevox.

reading of the text below it. For the Xanga and Wordpress blogs, plus the About page, the headings and text were wrapped in a `<div>` element. As the `<div>` element contained the CSS element that SADIe had access to, pressing the `n` key resulted in the heading and the text being read together in a continuous chunk as the behaviour-driven transcoding reads aloud the entire element automatically. Indeed P18 observed this and after identifying the answer to the Wordpress task stated *"last time it took you straight to the title at the top of the block so I couldn't understand why it wasn't doing it that time either."* Similarly for the Xanga blog task, P17 informed the evaluator that *"... because it started reading the paragraph straight away I probably couldn't distinguish between what was the title and what was the paragraph."*

The increased times may have also been due to participants unfamiliarity of the some of the pages. Again some participants were unfamiliar with the concept of a blog. P16 started the Xanga task and used `n` to access some of the content. After listening to the first blog entry (and therefore the answer to the task)

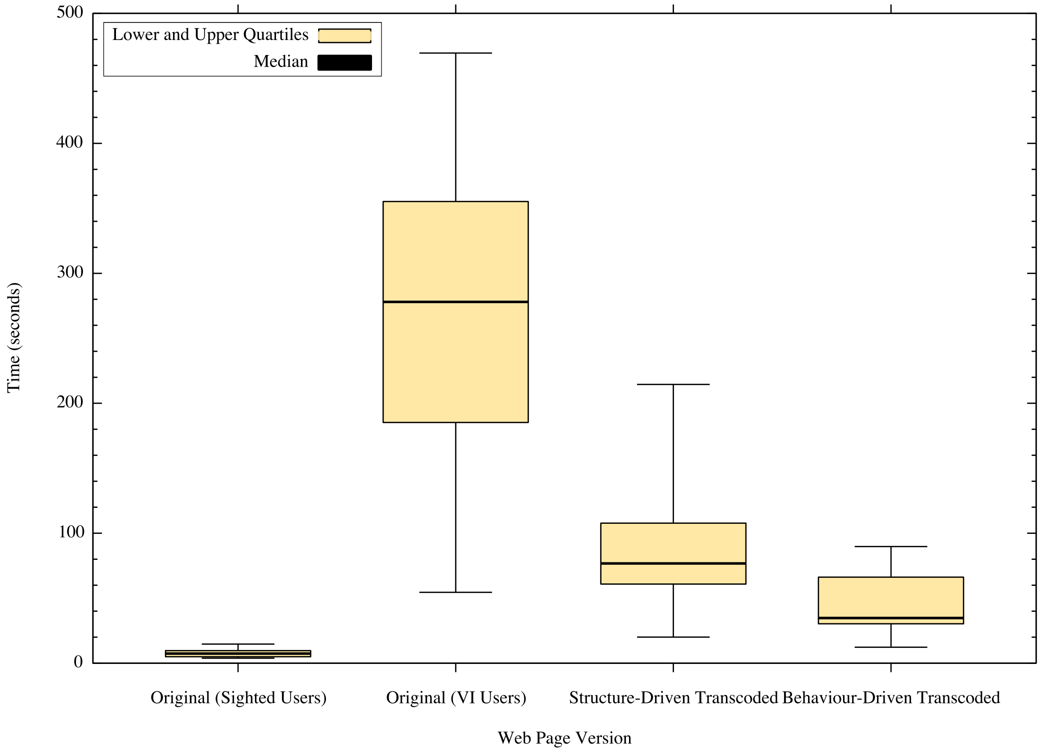


Figure 5.6: Comparison of the average time to find information on a page using SADIe behaviour-driven transcoding, SADIe structure-driven transcoding and the original version using Firevox.

P16 said *"what is a blog entry if I may ask?"* Therefore a combination of the annotations not being fine-grained enough and a lack of familiarity meant that participants required additional time to establish where the heading ended and where the text began in order to provide the answer to the task.

Overall participants seemed to find the behaviour-driven transcoding commands useful. P17 commented during the evaluation that they were *"really getting the hang of this now"*. P16 also used the commands to help with the blog task once they had understood what a blog actually was, saying *"I pressed 'm' and it gives [sic] me like the main menu isn't [sic] it?"* and also, when skimming through a news story, they commented *"I used 'p' thinking I would go to the top of the page."*

Figure 5.7 on page 140 shows a line graph of the results of all the user studies performed, both using Firevox and the participants' own equipment. The baseline shows the sighted users' task-completion time which the SADIe transcoding

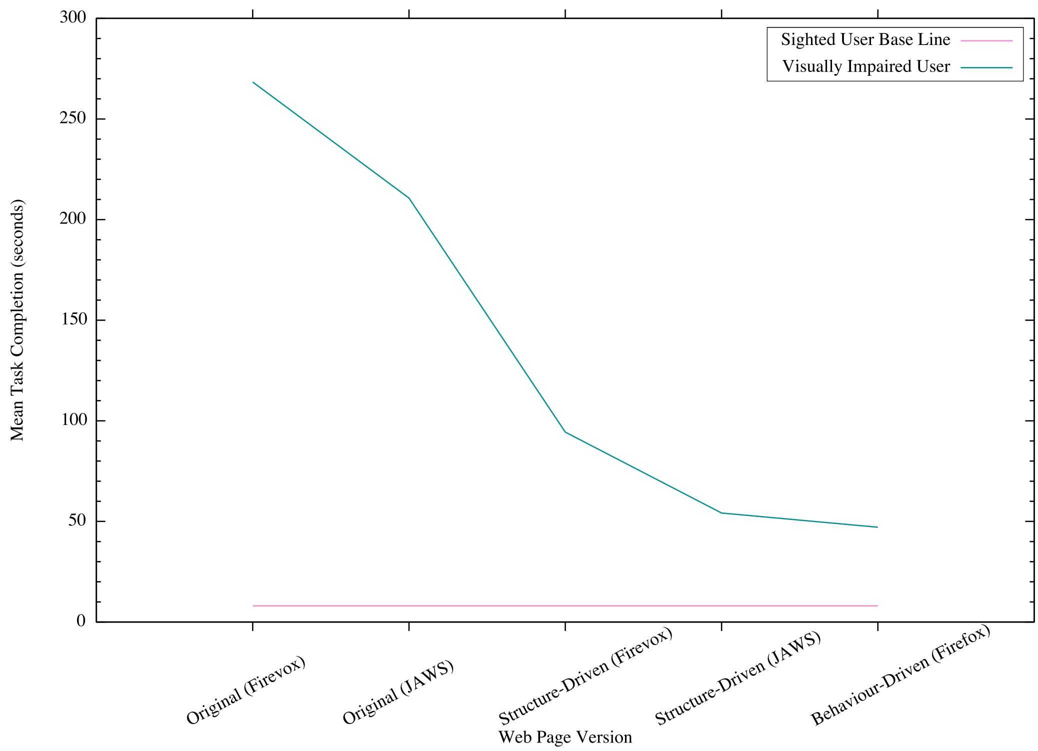


Figure 5.7: A graph showing how the different transcoding versions reduces the average task completion time for visually impaired users closer to the average task completion time of sighted users.

strives to achieve. As expected, Figure 5.7 shows that the results for studies conducted using Firevox were higher than the JAWS counterparts but a likely cause of this is that users were familiar with JAWS and had it set to their own preferences. Firevox was an unfamiliar application and users took some time to become comfortable with the commands. While making statistical comparisons between the four evaluations would be unfair due to the difference in location, user demographics and equipment, we assert that running the behaviour-driven transcoding evaluation with users' own equipment, when commercial screen reader software fully supports ARIA, will result in a further decrease in average completion time and move SADIe transcoding closer to the the sighted users' task completion time.

5.3 Summary and Conclusions

In this chapter we have seen how the behaviour-driven transcoding can aid users find information contained within a page. During a pilot evaluation of the behaviour-driven transcoding, it became apparent that ARIA was not fully supported by common commercial screen readers and so the evaluation had to be performed on the open source Firevox, an extension to the Mozilla Firefox Web browser that converts the Web browser into a self-voicing application. The original structure-driven transcoding studies allowed participants to use their own equipment. However, due to limited support of ARIA, the behaviour-driven transcoding evaluation had to be performed using Firevox. This change introduced additional variables into the results as not only was the transcoding altered but also the equipment. This would not have allowed for a fair comparison of results. Therefore the previous structure-driven transcoding study was repeated, using Firevox. The results of the repeated study mirrored the results of the original study and shown SADIe transcoding to reduce the time it took users to access information on Web pages. The behaviour-driven transcoding evaluation has also shown reduced task completion times that were less than the structure-driven transcoding task completion times. This result supported Research Question 4 by demonstrating that the use of behaviour-driven transcoding can improve access to content for visually impaired users.

While the results are encouraging, a number of factors should be addressed in future investigations. The first factor is the annotations of the pages as some of the users had difficulty completing the tasks due to not being able to easily reach specific parts of the content. Furthermore, the number of participants was small so that any variance in the results would be greatly magnified. However, the results, comments and observations from the evaluations were positive and demonstrated that behaviour-driven transcoding can assist users when accessing Web content.

The evaluations have also supported the use of SADIe as a transcoding platform. All the evaluation pages that were transcoded were done so in a manner that was consistent and predictable and was applied to a diverse range of Websites in a systematic fashion. While some users had difficulty completing tasks due to not being able to easily reach specific parts of the content, this was an issue with the manual annotations and not the SADIe transcoding method. A further refinement of the annotations used would help alleviate this problem.

Chapter 6

Summary and Conclusions

People with visual impairments are hindered when accessing information on the World Wide Web (Web) as it is not designed with their needs in mind. Visually impaired users can make use of screen readers that access the underlying structure of the page to create a sequential, audio rendering of the content. Most designers, however, are mainly concerned with how content is presented, rather than its structure and meaning. Consequently, implicit information available through the visual rendering of the page is lost to screen readers, and therefore the user. As discussed in Section 2.4, to address this problem, tools that transcode Web content into a format more suitable for screen readers have been developed. While these tools have improved access to the Web through structure-driven adaptations based upon anecdotal evidence, the approaches taken have either been scalable but inaccurate, or accurate but unscalable.

This thesis has presented research that has successfully demonstrated methods of improving transcoding systems based upon the four research questions set out in Section 1.1. These questions investigated the use of a hybrid transcoding solution that had the benefits of both heuristic and semantic transcoding. Furthermore, this thesis has demonstrated the feasibility of identifying user coping strategies as a means to derive behaviour-driven transcoding. The results of user evaluations have shown that behaviour-driven transcoding can enhance access to Web content beyond that provided by structure-driven transcoding. While a number of outstanding issues still remain, the work can now be used as a foundation for further research in the field.

6.1 Contribution of the Thesis

The four research questions set out in Section 1.1 have been successfully addressed and have made the following contributions to the field of Web accessibility.

6.1.1 SADIe Can Consistently Apply Transcoding to a Diverse Range of Websites Systematically

In Section 2 it was shown that transcoding approaches fall into two broad categories. The first is heuristic-based transcoding that can be applied to a wide variety of Websites without any additional information with regards to the page content. While such approaches are highly scalable, they lack accuracy and are prone to producing transcoding that does not always yield desired results [Asakawa & Takagi 2000]. The second is semantic-based transcoding, which uses annotations to provide additional information that can produce higher quality transcoded documents. However these approaches lack scalability due to the time required to annotate all the documents that are to be adapted [Takagi *et al.* 2002].

As described in Section 3 the SADIe approach annotates the Cascading Style Sheet elements as opposed to traditional annotation approaches that have focussed on annotating the Web page structure at the document level. This reduces the annotation overhead as, rather than annotate every page within a Website, annotation occurs in a single location at the CSS level. As the pages of the Website are associated with site-wide style definitions of the CSS elements, they not only inherit the rendering information but also the annotations that have been associated with the CSS. Such an approach provides a solution that combines the benefits of both heuristic and semantic transcoding. As annotations are being added to the elements, the role that they play is explicitly stated allowing for accurate transcoding to take place. However, a single annotation propagates through to every page within the Website due to the HTML elements being associated with the CSS. This therefore gives the scalability that previous semantic transcoding systems have lacked [Harper *et al.* 2006a, b].

A technical evaluation provided strong support that the SADIe transcoding method could consistently and predictably apply transcoding to a diverse range of Websites in a systematic fashion. Discounting those Websites that did not use any CSS for the presentation of the content, SADIe was able to successfully transcode 89% of Websites, with a true range lying between 99% and 79%. These results supported Research Question 1 by showing that the use of CSS annotations allows for a consistent and predictable transcoding platform to be constructed that can be applied to a diverse range of Websites in a systematic fashion. They also provided a solid basis for using SADIe as the transcoding platform upon which to investigate both structure-driven and behaviour-driven transcoding from a user perspective.

6.1.2 Coping Strategies Can Be Identified as Visually Impaired Users Access Web Content

As discussed in Section 2.4, the aim of transcoding is to adapt Web content to suit the needs of the user as they interact with the Web page. Previous transcoding approaches have used intuition-based adaptations to drive the transcoding functionality. For instance, removing the clutter that surrounds the main content may render the page more accessible for users. While such techniques may improve access to content, to further enhance Web accessibility, a behaviour-driven level of understanding is required based upon how users interact and cope with Web content. There had been limited research conducted into coping strategies used on the Web. As such, there lacked a framework with which to identify and define coping strategies that users employ during their interaction with the Web.

This research has shown that user interaction techniques on Websites can be identified though a framework of behavioural strategies. This framework was defined by data obtained from a series of summative experimental studies conducted in conjunction with Henshaws' Society for Blind People. These studies identified the coping strategies that users employed when access to Web content became too taxing, resulting in a framework consisting of seven dimensions. By taking these dimensions into account, a coping strategy can be described as a technique based upon a *psychological response* used by *one or more users* to overcome a problem due to some aspect of the *client or content*, developed through *various levels of experience* that is applicable to *one or more Web pages* using *one or more devices* in a specific *context of use*.

This framework was validated using two additional data sources, one of which was obtained from third-party sources and independent from any bias we may have introduced into the data acquisition process. The validation process demonstrated that the framework was flexible and accurate enough to be applicable to multiple data sources. This supports Research Question 2 by showing that it is possible to identify coping strategies in a consistent manner as users interact with the Web.

6.1.3 Coping Strategies Can Be Used to Define Transcoding Solutions

As discussed in Section 4, using the framework defined as part of Research Question 2, a number of coping strategies were identified from the two Henshaws' studies and the NoVA evaluation data. These strategies, however, represented all the instances of coping identified from the data. As such, they were described in a manner that was specific to the context in which they were identified and modelled. To be of practical benefit to users, similar transcoding needed to be applied to as diverse a range of Websites as possible to allow users to interact with the content in a consistent fashion.

Closer inspection of the identified coping strategies allowed six abstract patterns to be extracted. These patterns, described as Candidate Chunk Discovery Strategy; Masthead Avoidance Strategy; Clustered Element Strategy; Probing Strategy; Backtracking Strategy; and Withdrawal Strategy, allowed for the development of behaviour-driven transcoding functionality that would assist visually impaired users' access to Web content. The transcoding allowed for the easy navigation between important chunks of information on the page, easy access to the menu, and the ability to gain an overview of the page's content. These adaptations directly addressed the Candidate Chunk Discovery Strategy, Masthead Avoidance Strategy, Probing Strategy, and Backtracking Strategy. Furthermore, the Withdrawal Strategy was indirectly addressed as one could assume that providing transcodes to support users access content would reduce the number of times that they "gave up" completing a task. The transcodes therefore had the broadest benefit to users when implemented as they directly supported twenty, and indirectly support ten, of the thirty-three coping strategies that had been categorised and acted as a good test of the behaviour-driven transcoding approach.

The transcoding transformed static Web documents into interactive content by allowing users to navigate between key elements of the page through a consistent set of key presses. These included pressing `n` and `p` to provide access to important chunks; pressing `m` to provide access to the menu; and pressing `s` to receive a summary of the page's contents. The development of the transcoding from the abstract coping strategy patterns supported Research Question 3 by demonstrating that identified coping strategies can be used to define transcoding solutions. These solutions were then incorporated into the SADIe experimental transcoding proxy to form the basis of the behaviour-driven transcoding user evaluation.

6.1.4 Behaviour-Driven Transcoding Can Improve Access to Web Content

As discussed in Section 5, During a pilot evaluation of the behaviour-driven transcoding, it became apparent that ARIA was not fully supported by common commercial screen readers and so the evaluation had to be performed on the open source Firevox, an extension to the Mozilla Firefox Web browser that converts the Web browser into a self-voicing application. The original structure-driven transcoding studies allowed participants to use their own equipment. However, due to limited support of ARIA, the behaviour-driven transcoding evaluation had to be performed using Firevox. This change introduced additional variables into the results as not only was the transcoding altered but also the equipment. This would not have allowed for a fair comparison of results. Therefore the previous structure-driven transcoding study was repeated, using Firevox. The results of the repeated study mirrored the results of the original study and shown SADIe transcoding to reduce the time it took users to access information on Web pages. The behaviour-driven transcoding evaluation has also shown reduced task completion times that were less than the structure-driven transcoding task completion times. This result supported Research Question 4 by demonstrating that the use of behaviour-driven transcoding can improve access to content for visually impaired users.

The evaluations also supported the use of SADIe as a transcoding platform. All the evaluation pages that were transcoded were done so in a manner that was consistent and predictable and was applied to a diverse range of Websites in a systematic fashion. While some users had difficulty completing tasks due to difficulty reaching specific parts of the content, this was an issue with the manual annotations and not the SADIe transcoding method. A further refinement of the annotations used would help alleviate this problem. This further supports

Research Question 1 by showing that the use of CSS annotations allows for a consistent and predictable transcoding platform to be constructed that can be applied to a diverse range of Websites in a systematic fashion while being used by participants in an evaluation.

6.2 Outstanding Practical Issues

Although this work has made significant contributions to the field of Web accessibility, there are a number of practicalities that should be redressed.

Migrate to the Accessibility Tools Framework:
The IBM Accessibility Tools Framework (ACTF) is an open source infrastructure that allows developers to access and contribute to components used in accessibility applications. Examples of applications that are based on this framework include aDesigner [Takagi *et al.* 2004], an evaluation platform that allows developers to assess the accessibility of their Websites; and Home Page Reader [Asakawa & Itoh 1998] and HearSay [Raman 2008], two self-voice Web browsers widely used by visually impaired users. The advantage of migrating SADIe towards ACTF is that the transcoding code could be incorporated into mature applications, such as Home Page Reader that have a wide user base. This would eliminate the need for a proxy server approach and provide wider access to the transcoding functionality investigated in this research.

Utilise Accessibility Commons Metadata:
A number of transcoding tools and Web applications make use of annotations and metadata to drive the adaptation process. These tools often have independent metadata repositories that are not compatible with each other. Accessibility Commons is an attempt to create a common standard and repository of metadata that can be shared across tools and applications [Kawanaka *et al.* 2008]. By contributing to this project, not only will SADIe's annotations be available to a larger number of tools, but SADIe may be able to exploit some of the existing annotations in the common repository and provide transcoding to a larger number of Websites than is currently possible.

Standardise the CSS Annotation Specification:

The World Wide Web Consortium has a number of Working Groups that investigate adding semantic information to Web content. These include the Protocol for Web Description Resources Working Group[1] that is investigating a protocol for publishing descriptions and metadata about Web resources; and the Gleaning Resource Descriptions from Dialects of Languages Working Group[2] that investigates methods of adding and relating domain specific languages to XML, such as through the use of microformats. With such an interest in adding metadata to Web pages and describing resources, there is scope for the CSS-based annotations described in this work to be formalised and become part of a W3C recommendation.

Create a Pool of Participants:

One of the difficulties faced by researchers conducting work in Web accessibility is recruiting visually impaired users. Participants required for user studies typically need to be severely visually impaired, have experience at using the Web, be competent using a screen reader, live in the local area and be willing to come into a usability lab to conduct the evaluation. This reduces the number of potential recruits and limits the studies that can be conducted. One practical solution to this could be to develop a network of organisations that work with visually impaired users and form more formal collaborations. For example, people from the University could assist organisations in promoting computing and IT skills and in return, organisations could help with the recruitment of suitable participants. Such joint ventures could be mutually beneficial for both parties and increase the number of people willing to participate in future studies.

Further User Evaluations:

While the results of the user studies discussed in Section 5 are encouraging, a number of factors should be addressed in further evaluations. The first factor is the annotations of the pages as some of the users had difficulty completing the tasks due to not being able to easily reach specific parts of the content. This was an issue with the manual annotations that were added to the evaluation pages and not the SADIe transcoding method. Furthermore, the number of participants was small so that any variance in the

[1] URL: `http://www.w3.org/2007/powder/`. Last accessed: 31st October 2008

[2] URL: `http://www.w3.org/2001/sw/grddl-wg/`. Last accessed: 31st October 2008

results would be greatly magnified. Running additional studies with more participants from a wider demographic would further support the results provided in Section 5. However, the results, comments and observations from the evaluations already conducted were positive and demonstrated that the behaviour-driven transcoding that was developed can assist users when accessing Web content.

6.3 Future Research

The demonstration of the SADIe transcoding method and the behaviour-driven levels of transcoding presented in this work act as a proof of concept that can now be used as a foundation for future research.

Location of the CSS Annotations:

In this work, two approaches to annotating the structure of Web elements through the Cascading Style Sheet were investigated. The first involved a two-part ontology approach with a lower-ontology containing the elements found within the CSS and importing the abstract terms from an upper ontology. The second involved extending the keywords used within the Cascading Style Sheet and using the abstract terms from an ontology to provide suitable values to the keywords. While both annotation approaches provided a mechanism of describing the roles of Web elements, they both had weaknesses that require further investigation.

The two-part ontology blurred the application functionality and structural semantics of the Website. In addition it was taxonomic that made no use of inference engines to discover new facts and relationships between the elements of the Website. This took considerable time to construct and so from a pragmatic perspective was a large amount of effort for little gain.

In contrast, the CSS extension provided the same level of annotations as the ontology but with less effort. However, there are issues that still surround the approach that need to be addressed. Since CSS was first proposed, there has been an effort to promote the separation of style and content due to the benefits that it provides [Regan 2004]. These benefits include the ability to easily redesign a Website and the opportunity for users to apply different rendering to the page in order to better suit their needs, for example a high contrast style sheet for those with low vision. Our proposal to extend CSS with the `-uom-structural-role` property has the potential to conflate the rendering of a Website with the semantics of the content, which could result in similar problems that the merger of style and content created. Consider a paragraph defined as `<p class="a">` and an associated CSS definition of `.a{font-size: 90%; margin: 4em;}`. Due to the paragraph's smaller font size and wider margins, one can assume that the designer intended for all paragraphs of class `a` to be the abstract of an online document and the CSS rendering reflects this. Therefore explicitly stating this intended role by adding `-uom-structural-role: abstract;` to the CSS definition can have benefits for tools such as SADIe that need to process and adapt the content of the page. However, consider part of the abstract paragraph defined as `<span class="org">The University of Manchester</span>` with an associated CSS definition of `.org{-uom-structural-role: organisation;}`. No rendering information has been applied to the element and `organisation` is the semantic meaning of the phrase `The University of Manchester`, not a role that it plays on the Website. For such definitions, mechanisms such as RDFa and Microformats are required. The CSS syntax is valid and will not cause any validation errors, but as more annotated data becomes available online, it may be necessary to have a clear separation of the role elements play on the page due to their rendering and the semantic meaning of the content that is present within the page. Investigating which approach is more suitable to CSS annotation will need to consider a number of issues. The first is the practicality of annotating the CSS. To gain traction with developers, a balance should be considered between the benefits of annotating a Website and the effort required to apply those annotations. Secondly, investigations into the combining of semantics and rendering is necessary. It is not clear if mixing rendering and semantics will cause issues with future tools and technologies or if it is something that is not to be concerned about. One suspects that a clean model that separates annotation and rendering is required but further work is necessary to draw concrete conclusions.

Investigate Further Coping Strategies:

The strategies presented in this work have been a thin slice of coping strategies to act as a proof of concept. Three data sets have been used to define coping strategies, validate the definition and derive transcoding functionality. In order to identify additional strategies, patterns and transcoding techniques, more studies with a broader set of users will be required. This would further enhance our work on behaviour-driven transcoding and add additional support to the coping strategies framework as it is applied to more data sets with a variety of users coping with a diverse range of Websites.

Identify When To Apply Transcoding:

The transcoding user evaluation was applied with a range of participants, both beginners and experts, and a range of Websites, some of which were familiar to the participants and some were not. During the evaluation, observations were made that suggest the transcoding may have more benefit in specific scenarios. For example, during the task on Xanga, which involved a blog, P1 became very frustrated because they became lost in the page and were unable to identify any elements or information. Indeed, P1 described Xanga as *"rubbish on the screen"*. When the same page was transcoded using SADIe, the participants who performed the task did not have any difficulty with the page. By restructuring the content, SADIe can assist users in reaching the main information on the page, even when users are not familiar with the content. SADIe ensures that content is consistently near the top and the menu is consistently near the bottom, which can be of benefit to users when they are not familiar with the page layout before transcoding has taken place.

However, if users are familiar with the page the SADIe transcoding can be problematic as the user's mental model of the content is broken. For instance, when accessing the transcoded version of the BBC News page, P2 actually reached the main headline story within three seconds, but as they were not expecting the heading to appear so quickly, rapidly pressed `tab` to move down the page. On the BBC News Website, the main news is followed by news grouped by world regions. P2 was aware of this and it was not until they heard the phrase "Africa" that they started to `tab` backwards as they realised that they had gone too far.

While these were observations, they suggest that a possible benefit of SADIe transcoding is to target pages that users are not familiar with or to target users who are new to the Web and need assistance. One possibility is to allow users to control when the transcodes are applied. In the evaluation, participants had no choice over the transcoding. However, giving users the ability to apply adaptations only when they feel the need for assistance may empower the user and also improve access to content as familiar pages will not be "broken" and unfamiliar ones will be adapted so that users can interact with them in a consistent manner.

Investigate Alternative User Group Transcoding:

The SADIe architecture consists of two aspects. The first is the annotation of the Cascading Style Sheets. These annotations provide a model of the structural semantics of the Web page's content. They define what roles elements play within the page and are independent of the second aspect of the architecture which is the transformations that are applied to the content. In this work, the transcoding has focussed on visually impaired users, but there is scope for the annotations to be used to aid other user groups. People with cognitive disabilities, senior citizens and mobile Web users all have different access needs. Identifying the coping strategies that these user groups employ and adding those to the transcoding engine would further enhance access to the Web for all users.

Incorporate Web 2.0 Transcoding:

The Web is evolving from a static resource where users consume data to one where users interact with, and create, content contained within the page. These Websites, often described as Web 2.0, are more akin to applications than static documents, with the underlying technology that drives the interaction becoming more common [Chen 2008]. The transcoding investigated as part of this research has focused primarily on static Web pages. However, future transcodes will be required that not only adapt static documents, but also Web 2.0 Websites in order for all users to be able to interact and create content in an accessible manner.

6.4 Summary

The work presented in this thesis has demonstrated that the SADIe platform provides a transcoding solution that is both accurate and scalable. The principle idea behind the approach is that the rendering of Web page elements is closely associated with a role. Rather than annotate individual Web pages, as previous accurate transcoders have proposed, SADIe annotates the rendering information. On most Websites this is typically a Cascading Style Sheet that all pages refer to, providing a single point of annotation that propagates through to every page. Technical evaluations and user studies with established structure-driven transformations demonstrated that SADIe is both beneficial to users and technically feasible. This provided a stable platform from which to investigate behaviour-driven transcoding. Based upon an understanding of coping strategies users employ, summative experimental studies allowed for the creation of a framework that identified a number of strategies. Further analysis of these strategies led to the development of six abstract coping patterns. Transcoding solutions based on four of these patterns were incorporated into the SADIe platform. Quantitative and qualitative user studies of the enhanced SADIe prototype demonstrated that behaviour-driven transcoding improved access to information on Web pages beyond that of structure-driven transcoding.

The work presented in this thesis has contributed to the field of Web accessibility by demonstrating the feasibility of using behaviour-driven transcoding as a means of enhancing access to Web content. The behaviour-driven levels of transcoding presented in this work act as a proof of concept that can now be used as a foundation for further research in the field. This work has presented a small number of coping strategies that visually impaired users employ, yet further investigations into additional strategies and the subsequent transcoding are required. Furthermore, the research conducted in this thesis has predominantly focused on static Web content where the user consumes information. As the Web evolves, users are faced with interactive Web 2.0 pages, whereby users not only consume information but also interact with, and create, content. An understanding of the strategies users employ on these Websites and the transformations required to allow access to them is essential if the Web is to be accessible to all users.

Bibliography

Abe, Mari, & Hori, Masahiro. 2003. Robust Pointing by XPath Language: Authoring Support and Empirical Evaluation. *Page 156 of: SAINT '03: Proceedings of the 2003 Symposium on Applications and the Internet*. IEEE Computer Society.

Abiteboul, Serge, Buneman, Peter, & Suciu, Dan. 1999. *Data On The Web: From Relations to Semistructural Data and XML*. 1st edn. Morgan Kaufmann. ISBN: 1-558-60622-X.

Adida, Ben. 2008. hGRDDL: Bridging Microformats and RDFa. *Web Semantics: Science, Services and Agents on the World Wide Web*, **6**(1), 54–60.

Adida, Ben, Birbeck, Mark, McCarron, Shane, & Pemberton, Steven. 2008 (September). *RDFa in XHTML: Syntax and Processing*. W3C Proposed Recommendation. World Wide Web Consortium (W3C). `http://www.w3.org/TR/rdfa-syntax/`. Last Accessed 16th September 2008.

Adler, Sharon, Berglund, Anders, Caruso, Jeff, Deach, Stephen, Graham, Tony, Grosso, Paul, Gutentag, Eduardo, Milowski, Alex, Parnell, Scott, Richman, Jeremy, & Zilles, Steve. 2001 (October). *Extensible Stylesheet Language (XSL) Version 1.0*. W3C Recommendation. World Wide Web Consortium (W3C). `http://www.w3.org/TR/xsl/`. Last Accessed 17th September 2008.

Alliance for Technology Access. 2000. *Computer and Web Resources for People with Disabilities: A Guide to Exploring Today's Assistive Technologies*. 3rd edn. Hunter House. ISBN: 978-089-79330-01.

Amitay, Einat, Carmel, David, Darlow, Adam, Lempel, Ronny, & Soffer, Aya. 2003. The Connectivity Sonar: Detecting Site Functionality By Structural Patterns. *Pages 38–47 of: HYPERTEXT '03: Proceedings of the 14th ACM Conference on Hypertext and Hypermedia*. ACM Press.

Asakawa, Chieko. 2005. What's The Web Like If You Can't See It? *Pages 1–8 of: W4A '05: Proceedings of the 2005 International Cross-Disciplinary Workshop on Web Accessibility (W4A).* ACM Press.

Asakawa, Chieko, & Itoh, Takashi. 1998. User Interface of a Home Page Reader. *Pages 149–156 of: ASSETS '98: Proceedings of the 3rd International ACM SIGACCESS Conference on Computers and Accessibility.* ACM Press.

Asakawa, Chieko, & Takagi, Hironobu. 2000. Annotation-Based Transcoding For Nonvisual Web Access. *Pages 172–179 of: ASSETS '00: Proceedings of the 4th International ACM SIGACCESS Conference on Computers and Accessibility.* ACM Press.

Baader, Franz, Horrocks, Ian, & Sattler, Ulrike. 2004. *Handbook On Ontologies.* Springer. ISBN: 3-540-40834-7. Chap. 1: Description Logics, pages 3–28.

Baljko, Melanie, & Tam, Andrew. 2006. Indirect Text Entry Using One or Two Keys. *Pages 18–25 of: ASSETS '06: Proceedings of the 8th International ACM SIGACCESS Conference on Computers and Accessibility.* ACM.

Barrett, Beatrice H., Johnston, James M., & Pennypacker, H. S. 1986. *Conceptual Foundations of Behavioral Assessment.* The Guilford Behavioral Assessment Series. Guilford Press. ISBN: 0-898-62142-9. Chap. 6: Behavior: Its Units, Dimensions and Measurement, pages 156–200.

Bates, Marcia J. 1989. The Design of Browsing and Berrypicking Techniques for the Online Search Interface. *Online Information Review*, **13**(5), 407–424.

BBC. 1999 (April). *Betsie.* British Broadcasting Corporation, PO Box 1922, Glasgow, G2 3WT, UK. `http://betsie.sourceforge.net/`. Last Accessed 17th September 2008.

Bechhofer, Sean, Horrocks, Ian, Goble, Carole, & Stevens., Robert. 2001. OilEd: a Reason-able Ontology Editor for the Semantic Web. *Pages 396–408 of: KI2001: Proceedings of the Joint German/Austrian Conference on Artificial Intelligence.* Springer Berlin / Heidelberg.

Berelson, Bernard, & Steiner, Gary A. 1964. *Human Behavior: An Inventory of Scientific Findings.* 1st edn. Harcourt, Brace & World.

Bergman, Eric, & Johnson, Earl. 1995. Toward Accessible Human-Computer Interaction. *Advances In Human-Computer Interaction*, **5**, 87–113.

Berners-Lee, Tim. 1992. *HTML Tags*. Tech. rept. World Wide Web Consortium (W3C). `http://www.w3.org/History/19921103-hypertext/hypertext/WWW/MarkUp/Tags.html`. Last Accessed 17th September 2008.

Berners-Lee, Tim. 1999. *Weaving the Web: The Original Design and Ultimate Destiny of the World Wide Web by Its Inventor*. 1st edn. Harper Collins Publishing. ISBN: 0-06-251586-1.

Berners-Lee, Tim. 2002 (April). *RDF in HTML*. Tech. rept. World Wide Web Consortium (W3C). `http://www.w3.org/2002/04/htmlrdf`. Last Accessed 17th September 2008.

Bickmore, Timothy, Girgensohn, Andreas, & Sullivan, Joseph W. 1999. Web Page Filtering And Re-Authoring For Mobile Users. *The Computer Journal*, **42**(6), 34–46.

Bigham, Jeffrey P., Cavender, Anna C., Brudvik, Jeremy T., Wobbrock, Jacob O., & Lander, Richard E. 2007. WebinSitu: a Comparative Analysis of Blind and Sighted Browsing Behavior. *Pages 51–58 of: ASSETS '07: Proceedings of the 9th International ACM SIGACCESS Conference on Computers and Accessibility*. ACM Press.

Blenkhorn, Paul, Evans, David Gareth, & Baude, Alex. 2002. Full-Screen Magnification for Windows Using DirectX Overlays. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, **10**(4), 225–231.

Boag, Scott, Chamberlin, Don, Fernández, Mary F., Florescu, Daniela, Robie, Jonathan, & Siméon, Jérôme. 2007 (January). *XQuery 1.0: An XML Query Language*. W3C Recommendation. World Wide Web Consortium (W3C). `http://www.w3.org/TR/xquery/`. Last Accessed 17th September 2008.

Boldt, Axel. 2000 (July). *Web Filter*. Metropolitan State University, Department of Mathematics, St John's Hall, Room 311, 700 E Seventh Street, St. Paul, MN, 55106, U.S.A. `http://math-www.uni-paderborn.de/~axel/NoShit/`. Last Accessed 17th September 2008.

Bos, Bert, Çelik, Tantek, Hickson, Ian, & Lie, Håkon Wium. 2007 (July). *Cascading Style Sheets Level 2 Revision 1 (CSS 2.1) Specification*. Candidate Recommendation 2.1. World Wide Web consortium (W3C). `http://www.w3.org/TR/CSS21/`. Last Accessed 17th September 2008.

Braun, Virginia, & Clarke, Victoria. 2006. Using Thematic Analysis in Psychology. *Qualitative Research in Psychology*, **3**(2), 77–101.

Bray, Tim, Paoli, Jean, Sperberg-McQueen, C. M., Maler, Eve, & Yergeau, François. 2006 (September). *Extensible Markup Language (XML) 1.0*. W3C Recommendation. World Wide Web Consortium (W3C). `http://www.w3.org/TR/xml/`. Last Accessed 17th September 2008.

Brown, Andy, & Jay, Caroline. 2008 (September). *A Review of Assistive Technologies: Can Users Access Dynamically Updating Information?* SASWAT Technical Report 2. The University of Manchester, School of Computer Science, Kilburn Building, Oxford Road, Manchester, M13 9PL, UK. `http://hcw-eprints.cs.manchester.ac.uk/70/`. Last Accessed 26th November 2009.

Brown, Silas S., & Robinson, Peter. 2001. A World Wide Web Mediator for Users with Low Vision. *In: CHI '01: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems Workshop No. 14: Universal Design: Towards Universal Access in the Information Society.*

Burks, Michael R., Lauke, Patrick H., Thatcher, Jim, Rutter, Richard, & Waddell, Cynthia. 2006. *Web Accessibility: Web Standards and Regulatory Compliance*. Friends Of Ed. ISBN: 9-785-9059638-2.

Buyukkokten, Orkut, Garcia-Molina, Hector, & Paepcke, Andreas. 2001. Accordion Summarization for End-Game Browsing on Pdas and Cellular Phones. *Pages 213–220 of: CHI '01: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems.* ACM Press. ISBN: 1-58113-327-8.

Caldwell, Ben, Cooper, Michael, Reid, Loretta Guarino, & Vanderheiden, Gregg. 2008 (December). *Web Content Accessibility Guidelines (WCAG) 2.0.* W3C Recommendation 2.0. World Wide Web Consortium (W3C). `http://www.w3.org/TR/WCAG20/`. Last Accessed 27th January 2009.

Card, Stuart K., Moran, Thomas P., & Newell., Allen. 1983. *The Psychology of Human-Computer Interaction*. Laurence Erlbaum Associates. ISBN: 0-898-59243-7.

Carr, L.A., Roure, D. De, Hall, W., & Hill, G. 1998. Implementing an Open Link Service for the World Wide Web. *World Wide Web*, **1**(2), 61–71.

Carver, C.S., Scheier, M.F., & Weintraub, J.K. 1989. Assessing Coping Strategies: A Theoretically Based Approach. *Journal of Personality and Social Psychology*, **56**(2), 267–283.

Çelik, Tantek, & Suda, Brian. 2008 (June). *hCalendar*. Specification. Microformats. `http://microformats.org/wiki/hcalendar`. Last Accessed 17$^{\text{th}}$ September 2008.

Chein, Isidor. 1972. *The Science of Behavior and the Image of Man*. Tavistock Publications. ISBN: 0-422-74130-2.

Chen, Alex Qiang. 2008 (September). *Web Evolution*. Master of Science in Advanced Computer Science, The University of Manchester, School of Computer Science.

Chen, Charles L., & Ramen, T. V. 2008. AxsJAX: A Talking Translation Bot Using Google IM: Bringing Web-2.0 Applications to Life. *Pages 54–56 of: W4A '08: Proceedings of the 2008 International Cross-Disciplinary Workshop on Web Accessibility (W4A)*. ACM Press.

Chen, Jinlin, Zhou, Baoyao, Shi, Jin, Zhang, Hongjiang, & Fengwu, Qiu. 2001. Function-Based Object Model Towards Website Adaptation. *Pages 587–596 of: WWW '01: Proceedings of the 10th International Conference on the World Wide Web*. ACM Press.

Chen, Yu, Ma, Wei-Ying, & Zhang, Hong-Jiang. 2003. Detecting Web Page Structure For Adaptive Viewing On Small Form Factor Devices. *Pages 225–233 of: WWW '03: Proceedings of the 12th International Conference on the World Wide Web*. ACM Press.

Chisholm, Wendy, Vanderheiden, Gregg, & Jacobs, Ian. 1999 (May). *Web Content Accessibility Guidelines 1.0*. W3C Recommendation 1.0. World Wide Web

Consortium (W3C). `http://www.w3.org/TR/WCAG10/`. Last Accessed 3rd October 2008.

Chisholm, Wendy, Vanderheiden, Gregg, & Jacobs, Ian. 2000 (November). *Core Techniques for Web Content Accessibility Guidelines 1.0*. W3C Note 1.0. World Wide Web Consortium (W3C). `http://www.w3.org/TR/WCAG10-CORE-TECHS/`. Last Accessed 17th September 2008.

Clark, James, & DeRose, Steve. 1999 (November). *XML Path Language (XPath) Version 1.0*. W3C Recommendation. World Wide Web Consortium (W3C). `http://www.w3.org/TR/xpath`. Last Accessed 16th September 2008.

Cooper, Michael, Schwerdtfeger, Rich, Seeman, Lisa, & Pappas, Lisa. 2008 (August). *Accessible Rich Internet Applications (WAI-ARIA) Version 1.0*. W3C Working Draft. World Wide Web Consortium (W3C). `http://www.w3.org/TR/wai-aria/`. Last Accessed 3rd October 2008.

Cranford-Teague, Jason. 2004. *DHTML and CSS For The World Wide Web*. 3rd edn. Visual Quickstart Guide. Peachpit Press. ISBN: 0-321-19958-8.

Craven, Jenny, & Brophy, Peter. 2003 (January). *Non-Visual Access to the Digital Library: The Use of Digital Library Interfaces by Blind and Visually Impaired People*. Library and Information Commission Research Report 145. Manchester Metropolitan University, Centre for Research in Library and Information Management.

Dalal, Nikunj P., Quible, Zane, & Wyatt, Katherine. 2000. Cognitive Design Of Home Pages: An Experimental Study Of Comprehension On The World Wide Web. *Information Processing and Management*, **36**(4), 607–621.

DeRose, Steve, Maler, Eve, & Orchard, David. 2001 (January). *XML Linking Language (XLink) Version 1.0*. W3C Recommendation. World Wide Web Consortium (W3C). `http://www.w3.org/TR/xlink/`. Last Accessed 17th September 2008.

Dix, Alan, Finlay, Janet, Abowd, Gregory, & Beale, Russel. 1993. *Human Computer Interaction*. 1st edn. Prentice Hall. ISBN: 0-134-58266-7.

DRC. 2004. *The Web. Access and Inclusion for Disabled People*. Formal Investigation. Disability Rights Commission. ISBN: 0-117-03287-5.

Dumas, Joseph S., & Redish, Janice C. 1999. *A Practical Guide To Usability Testing*. Intellect. ISBN: 1-841-50020-8.

Edwards, Alistair D. N. 2008. *Web Accessibility: A Foundation for Research*. 1st edn. Human-computer Interaction Series. Springer-Verlag. ISBN: 978-1-84800-049-0. Chap. 10: Assistive Technologies, pages 141–162.

Felzer, Torsten, & Nordman, Rainer. 2006. Alternative Text Entry Using Different Input Methods. *Pages 10–17 of: ASSETS '06: Proceedings of the 8th International ACM SIGACCESS Conference on Computers and Accessibility*. ACM.

Flores, Francesc Campoy, Quint, Vincent, & Vatton, Irène. 2006. Templates, Microformats and Structured Editing. *Pages 188–197 of: DocEng '06: Proceedings of the 2006 ACM Symposium on Document Engineering*. ACM Press.

Folkman, Susan, & Lazarus, Richard S. 1985. If It Changes It Must Be a Process: Study of Emotion and Coping During Three Stages of a College Examination. *Journal of Personality and Social Psychology*, **48**(1).

Freedom Scientific. 2007 (November). *JAWS for Windows*. 9 edn. Freedom Scientific BLV Group LLC, 11800 31st Court North, St. Petersburg, Florida 33716-1805, USA. `http://www.freedomscientific.com/jaws/`. Last Accessed 17th September 2008.

Goldfarb, Charles F. 1990. *The SGML Handbook*. Oxford University Press. ISBN: 0-198-53737-9.

Gupta, Suhit, Kaiser, Gail, Neistadt, David, & Grimm, Peter. 2003. DOM-based Content Extraction of HTML Documents. *Pages 207–214 of: WWW '03: Proceedings of the 12th International Conference on the World Wide Web*. ACM Press.

Gupta, Suhit, Kaiser, Gail E., Grimm, Peter, Chiang, Michael F., & Starren, Justin. 2005. Automating Content Extraction of HTML Documents. *World Wide Web*, **8**(2), 179–224.

Gutwin, Carl, & Fedak, Chris. 2004. Interacting With Big Interfaces On Small Screens: A Comparison Of Fisheye, Zoom, And Panning Techniques. *Pages*

145–152 of: GI '04: Proceedings of the 2004 Conference on Graphics Interface. Canadian Human-Computer Communications Society.

GW Micro. 2007. *Window-Eyes 6.1 Manual*. GW Micro, Inc., 725 Airport North Office Park, Fort Wayne, IN 46825, USA. `http://www.gwmicro.com/Window-Eyes/`. Last Accessed 17$^{\text{th}}$ September 2008.

Harada, Susumu, Wobbrock, Jacob O., & Landay, James A. 2007. VoiceDraw: A Hands-Free Voice-Driven Drawing Application for People With Motor Impairments. *Pages 27–34 of: ASSETS '07: Proceedings of the 9th International ACM SIGACCESS Conference on Computers and Accessibility*. ACM.

Harper, Simon, & Bechhofer, Sean. 2004. Lightweight Mark-Up for Low Cost Semantics. *In: Proceedings of the ISWC 2004 Workshop on Knowledge Markup and Semantic Annotation (Semannot'2004)*. CEUR Workshop Proceedings (CEUR-WS.org).

Harper, Simon, & Bechhofer, Sean. 2005. Semantic Triage for Increased Web Accessibility. *IBM Systems Journal*, **44**(3), 637–648.

Harper, Simon, & Yesilada, Yeliz. 2007. Web Authoring for Accessibility (WAfA). *Web Semantics: Science, Services and Agents on the World Wide Web*, **5**(3), 175–179.

Harper, Simon, Goble, Carole, Stevens, Robert, & Yesilada, Yeliz. 2003. Towel: Experiences Of Augmenting The Web With Journey Knowledge. *In: WWW '03: Proceedings of the 12th International Conference on the World Wide Web*. ACM Press.

Harper, Simon, Yesilada, Yeliz, & Goble, Carole. 2004. Special Issue on Accessible Hypermedia and Multimedia. *Chap. Introduction, pages 115–126 of: New Review of Hypermedia and Multimedia*, vol. 10. Taylor and Francis.

Harper, Simon, Bechhofer, Sean, & Lunn, Darren. 2006a. SADIe: Transcoding Based On CSS. *Pages 259–260 of: ASSETS '06: Proceedings of the 8th International ACM SIGACCESS Conference on Computers and Accessibility*. ACM Press.

Harper, Simon, Bechhofer, Sean, & Lunn, Darren. 2006b. Taming The Inaccessible Web. *Pages 64–69 of: SIGDOC '06: Proceedings of the 24th ACM International Conference on Design of Communication*. ACM Press.

Hickson, Ian, & Hyatt, David. 2009 (August). *HTML 5: A vocabulary and associated APIs for HTML and XHTML*. W3C Working Draft. World Wide Web Consortium (W3C). `http://www.w3.org/TR/html5`. Last Accessed 26th November 2009.

Holman, Ken. 2000 (August). *What is XSLT?* O'Reilly & Associates, `http://www.xml.com`. Online Publication `http://www.xml.com/pub/a/2000/08/holman/`.

Hori, Masahiro, Kondoh, Goh, Ono, Kouichi, Hirose, Shinichi, & Singhal, Sandeep. 2000. Annotation-Based Web Content Transcoding. *Pages 197–211 of: Proceedings of the 9th International Conference on the World Wide Web On Computer Networks: The International Journal Of Computer And Telecommunications Netowrking*. North-Holland Publishing Co.

Hori, Masahiro, Abe, Mari, & Ono, Kouichi. 2003 (March). Extensible Framework of Authoring Tools for Web Document Annotation. *Pages 1–8 of: Proceedings of the International Workshop on Semantic Web Foundations and Application Technologies (SWFAT)*.

Horridge, Matthew, & Patel-Schneider, Peter F. 2009 (October). *OWL 2 Web Ontology Language Manchester Syntax*. W3C Working Group Note. World Wide Web Consortium (W3C). `http://www.w3.org/TR/owl2-manchester-syntax/`. Last Accessed 26th November 2009.

Huang, Anita W., & Sundaresan, Neel. 2000. A Semantic Transcoding System To Adapt Web Services For Users With Disabilities. *Pages 156–163 of: ASSETS '00: Proceedings of the 4th International ACM SIGACCESS Conference on Computers and Accessibility*. ACM Press.

Hutt, Max L, Isaacson, Robert L, & Blum, Milton L. 1966. *The Science of Interpersonal Behavior*. Harper & Row.

Huynh, David, Mazzocchi, Stefano, & Karger, David. 2005. Piggy Bank: Experience the Semantic Web Inside Your Web Browser. *Pages 413–430 of: ISWC*

2005: Proceedings of the 4th International Semantic Web Conference. Springer-Verlag Heidelberg.

Ihde, Steven C., Maglio, Paul, Meyer, Jörg, & Barrett, Rob. 2001. Intermediary-Based Transcoding Framework. *IBM Systems Journal*, **40**(1), 179–192.

Ivory, Melody Y., & Hearst, Marti A. 2001. The State of the Art in Automating Usability Evaluation of User Interfaces. *ACM Computing Surveys*, **33**(4), 470–516.

Ivory, Melody Y., & Megraw, Rodrick. 2005. Evolution Of Web Site Design Patterns. *ACM Transactions on Information Systems (TOIS)*, **23**(4), 463–497.

James, Frankie. 1998 (May). *Presenting HTML Structure in Audio: User Satisfaction with Audio Hypertext*. Tech. rept. CS-TN-98-68. Stanford University, Department of Computer Science. `ftp://reports.stanford.edu/pub/cstr/reports/cs/tn/98/68/`. Last Accessed 17th September 2008.

Jang, Minsu, Kim, Jaehong, & Sohn, Joo-Chan. 2003. Web Content Adaptation and Transcoding Based on CC/PP and Semantic Templates. *Page 195 of: WWW '03: Proceedings of the 12th International Conference on the World Wide Web*. ACM Press.

Jay, Caroline, Stevens, Robert, Glencross, Mashhuda, & Chalmers, Alan. 2006. How People Use Presentation To Search For A Link: Expanding The Understanding Of Accessibility On The Web. *Pages 113–120 of: W4A: Proceedings of the 2006 International Cross-Disciplinary Workshop on Web Accessibility (W4A)*. ACM Press.

Jay, Caroline, Lunn, Darren, & Michailidou, Eleni. 2008. *Web Accessibility: A Foundation for Research*. 1st edn. Human-computer Interaction Series. Springer-Verlag. ISBN: 9-781-84800049-0. Chap. 4 part II: End User Evaluations, pages 107–126.

John, Bonnie E., & Kieras, David E. 1996. The GOMS Family of User Interface Analysis Techniques: Comparison and Contrast. *ACM Transactions on Computer-Human Interaction (TOCHI)*, **3**(4), 320–351.

Kahan, José, & Koivunen, Marja-Ritta. 2001. Annotea: An Open RDF Infrastructure For Shared Web Annotations. *Pages 623–632 of: Proceedings of the 10th International Conference on the World Wide Web.*

Kawanaka, Shinya, Borodin, Yevgen, Bigham, Jeffrey P., Lunn, Darren, Takagi, Hironobu, & Asakawa, Chieko. 2008. Accessibility Commons: A Metadata Infrastructure for Web Accessibility. *Pages 153–160 of: ASSETS '08: Proceedings of the 10th International ACM SIGACCESS Conference on Computers and Accessibility.* ACM Press.

Kay, Michael. 2004. *XSLT 2.0 Programmer's Reference.* 3rd edn. Programmer To Programmer. Wrox Press. ISBN: 0-764-56909-0. Chap. 1: XSLT In Context, pages 1–42.

Kay, Michael. 2007. *XSL Transformations (XSLT) Version 2.0.* W3C Recommendation. World Wide Web Consortium (W3C). `http://www.w3.org/TR/xslt20/`. Last Accessed 17th September 2008.

Kelly, Brian, Sloan, David, Phipps, Lawrie, Petrie, Helen, & Hamilton, Fraser. 2005. Forcing Standardization Or Accommodating Diversity?: A Framework For Applying The WCAG In The Real World. *Pages 46–54 of: W4A '05: Proceedings of the 2005 International Cross-Disciplinary Workshop on Web Accessibility (W4A).* ACM Press.

Kew, Nick. 2001 (September). *Why Validate?* W3C Markup Validation Service Mailing List Archive `http://lists.w3.org/Archives/Public/www-validator/2001JulSep/0597.html`. Last Accessed 17th September 2008.

Khare, Rohit, & Çelik, Tantek. 2006. Microformats: A Pragmatic Path To The Semantic Web. *Pages 865–866 of: Proceedings of the 15th International Conference on the World Wide Web.* ACM Press.

King, A., Evans, G., & Blenkhorn, P. 2004 (March). Webbie, A Web Browser For Visually Impaired People. *In: CWUAAT '04: Proceedings of the 2nd Cambridge Workshop on Universal Access and Assistive Technology.*

King, Alasdair Robin. 2006 (September). *Re-presenting Visual Content For Blind People.* phd, The University of Manchester, School of Informatics.

King, Ryan. 2008 (June). *hResume*. Specification. Microformats. `http://microformats.org/wiki/hResume`. Last Accessed 17th September 2008.

Lazarus, Richard S. 1966. *Psychological Stress and the Coping Process*. McGraw-Hill series in psychology. McGraw-Hill.

Lazarus, Richard S. 1993. Coping Theory And Research: Past, Present, And Future. *Psychosomatic Medicine*, **55**(3), 234–247.

Lazarus, Richard S., & Folkman, Susan. 1984. *Stress, Appraisal, and Coping*. Springer. ISBN: 0-826-14190-0.

Lie, Håkon Wium. 2005 (March). *Cascading Style Sheets*. Ph.D. thesis, University of Oslo, Faculty of Mathematics and Natural Sciences.

Lie, Håkon Wium, & Bos, Bert. 1998 (September). *Using XSL and CSS Together*. 2nd edn. World Wide Web Consortium (W3C). `http://www.w3.org/TR/NOTE-XSL-and-CSS`. Last Accessed 17th September 2008.

Maanen, John Van (ed). 1979. *Qualitative Methods*. SAGE Publications Ltd. ISBN: 0-803-92117-9.

Maeda, Junji, Fukuda, Kentarou, Takagi, Hironobu, & Asakawa, Chieko. 2003. WebDigest: Layout-Preserving Visually Enhanced Web Pages. *Page 418 of: SAINT '03: Proceedings of the 2003 Symposium on Applications and the Internet*. Washington, DC, USA: IEEE Computer Society.

Maglio, Paul, & Barrett, Rob. 2000. Intermediaries Personalize Information Streams. *Communications of the ACM*, **43**(8), 96–101.

Mahmud, Jalal U., Borodin, Yevgen, & Ramakrishnan, I. V. 2007. Csurf: A Context-Driven Non-Visual Web-Browser. *Pages 31–40 of: WWW '07: Proceedings of the 16th International Conference on The World Wide Web*. ACM Press.

Manola, Frank, & Miller, Eric. 2004 (February). *RDF Primer Version 1.0*. W3C Recommendation. World Wide Web Consortium (W3C). `http://www.w3.org/TR/rdf-primer/`. Last Accessed 17th September 2008.

McGuinness, Deborah L., & van Harmelen, Frank. 2004 (February). *OWL Web Ontology Language Overview*. W3C Recommendation. World Wide Web Consortium (W3C). `http://www.w3.org/TR/owl-features/`. Last Accessed 17th September 2008.

McKellar, Peter. 1989. *Abnormal Psychology: It's Experience and Behaviour*. Routledge. ISBN: 0-415-02812-4.

Meyer, Eric A. 2005. Is Accessible Design A Myth? *Pages 60–62 of: W4A '05: Proceedings of the 2005 International Cross-Disciplinary Workshop on Web Accessibility (W4A)*. ACM Press.

Miles, Matthew B., & Huberman, Michael. 1994. *Qualitative Data Analysis: An Expanded Sourcebook*. 2nd edn. SAGE Publications Ltd. ISBN: 0-803-95540-5.

Millard, David E., & Ross, Martin. 2006. Web 2.0: Hypertext By Any Other Name? *Pages 27–30 of: HYPERTEXT '06: Proceedings of the 17th Conference on Hypertext and Hypermedia*. ACM Press.

Miller, Eric. 1998. An Introduction to the Resource Description Framework. *D-Lib magazine*, **4**(5), 2000. Online Publication.

Mitchell, Tom M. 1997. *Machine Learning*. 1st edn. McGraw-Hill Series in Computer Science. McGraw-Hill. ISBN: 0-071-15467-1.

Mukherjee, Saikat, Yang, Guizhen, & Ramakrishnan, I.V. 2003. Automatic Annotation of Content-Rich HTML Documents: Structural and Semantic Analysis. *Pages 533–549 of: ISWC 2003: Proceedings of the 2nd International Semantic Web Conference*. Springer-Verlag.

Musciano, Chuck, & Kennedy, Bill. 2000. *HTML and XHTML: The Definitive Guide*. 4th edn. O'Reilly. ISBN: 0-596-00026-X.

Nagao, Katashi. 2000 (August). Semantic Transcoding: Making The World Wide Web More Understandable and Usable with External Annotations. *In: SSGRR 2000: Proceedings of the International Conference on Advances in Infrastructure for E-Business, Science, and Education on the Internet*.

Nelsson, Rosemery O., & Hayes, Steven C. 1986. *Conceptual Foundations of Behavioral Assessment*. The Guilford Behavioral Assessment Series. Guilford

Press. ISBN: 0 898 621 429. Chap. 1: The Nature of Behavioral Assessment, pages 3–41.

Nguyen, Kevin K., & Petty, Linda S. 1996. *Web Browsing through Adaptive Technology: A Consumer Information Resource.* Tech. rept. Adaptive Technology Resource Centre (ATRC), University of Toronto, Ontario, Canada.

Nielsen, Jakob. 2004a. Risks of Quantitative Studies. *Alertbox: Current Issues in Web Usability*, March. `http://www.useit.com/alertbox/20040301.html`. Last Accessed 17th September 2008.

Nielsen, Jakob. 2004b. *Usability Engineering.* 1st edn. The Morgan Kaufmann Series in Interactive Technologies. Academic Press Inc. ISBN: 0-125-18406-9.

Olson, Judith Reitman, & Olson, Gary M. 1990. The Growth of Cognitive Modeling in Human-Computer Interaction Since GOMS. *Human-Computer Interaction*, **5**(2), 221–265.

Palmer, Sean. 2002. RDF in HTML: Approaches. *Infomesh*, June, 1–12. Online Publication. `http://infomesh.net/2002/rdfinhtml/`. Last Accessed 17th September 2008.

Parmanto, Bambang, Ferrydiansyah, Reza, Saptono, Andi, Song, Lijing, Sugiantara, I Wayan, & Hackett, Stephanie. 2005. AcceSS: Accessibility Through Simplification & Summarization. *Pages 18–25 of: W4A '05: Proceedings of the 2005 International Cross-Disciplinary Workshop on Web Accessibility (W4A).* ACM Press.

Pavlov, Ivan Petrovich. 1927. *Conditioned Reflexes: an Investigation of the Physiological Activity of the Cerebral Cortex.* Oxford University Press.

Pemberton, Steven, Austin, Daniel, Axelsson, Jonny, Çelik, Tantek, Dominiak, Doug, Elenbaas, Herman, Epperson, Beth, Ishikawa, Masayasu, Matsui, Shin'ichi, McCarron, Shane, Navarro, Ann, Peruvemba, Subramanian, Relyea, Rob, Schnitzenbaumer, Sebastian, & Stark, Peter. 2002 (August). *XHTML 1.0 The Extensible HyperText Markup Language.* W3C Recommendation. World Wide Web Consortium (W3C). `http://www.w3.org/TR/xhtml1/`. Last Accessed 17th September 2008.

Petrie, Helen, & Kheir, Omar. 2007. The Relationship Between Accessibility and Usability of Websites. *Pages 397–406 of: CHI '07: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems.* ACM Press.

Petrie, Helen, Hamilton, Fraser, & King, Neil. 2004. Tension, What Tension?: Website Accessibility And Visual Design. *Pages 13–18 of: W4A '04: Proceedings Of The 2004 International Cross-Disciplinary Workshop on Web Accessibility (W4A).* ACM Press.

Pilgrim, Mark. 2002. *Dive Into Accessibility.* Online Publication. `http://diveintoaccessibility.org/`. Last Accessed 17th September 2008.

Pole, Christopher, & Morrison, Marlene. 2003. *Ethnography For Education.* Doing Qualitative Research In Educational Settings. Open University Press. ISBN: 0-335-20600-X.

Raggett, Dave, Hors, Arnaud Le, & Jacobs, Ian. 1999 (December). *HTML 4.01 Specification.* W3C Recommendation. World Wide Web Consortium (W3C). `http://www.w3.org/TR/html401/`. Last Accessed 17th September 2008.

Raman, T. V. 2008. Cloud Computing And Equal Access For All. *Pages 1–4 of: W4A '08: Proceedings of the 2008 International Cross-Disciplinary Workshop on Web Accessibility (W4A).* ACM Press.

Ramen, T. V. 2008. *Web Accessibility: A Foundation for Research.* 1st edn. Human-computer Interaction Series. Springer-Verlag. ISBN: 978-1-84800-049-0. Chap. 12: Specialized Browsers, pages 195–213.

Regan, Bob. 2004. Accessibility And Design: A Failure Of The Imagination. *Pages 29–37 of: W4A '04: Proceedings Of The 2004 International Cross-Disciplinary Workshop On Web Accessibility (W4A).* ACM Press.

Reid, Loretta Guarino, & Snow-Weaver, Andi. 2008. WCAG 2.0: A Web Accessibility Standard for the Evolving Web. *Pages 109–115 of: W4A '08: Proceedings of the 2008 International Cross-Disciplinary Workshop on Web Accessibility (W4A).* ACM Press.

Rice, Pranee Liamputtong, & Ezzy, Douglas. 1999. *Qualitative Research Methods.* A Health Focus. Oxford University Press. ISBN: 0-195-50610-3.

RNIB. 2004 (July). *See It Right Accessible Website Audits.* Royal National Institute of Blind People, 105 Judd Street, London, WC1H 9NE, UK.

Seeman, Lisa. 2004. The Semantic Web, Web Accessibility, And Device Independence. *Pages 67–73 of: W4A '04: Proceedings Of The 2004 International Cross-Disciplinary Workshop on Web Accessibility (W4A).* ACM Press.

Shinohara, Kristen, & Tenenberg, Josh. 2007. Observing Sara: A Case Study of a Blind Person's Interactions With Technology. *Pages 171–178 of: ASSETS '07: Proceedings of the 9th International ACM SIGACCESS Conference on Computers and Accessibility.* ACM Press.

Takagi, Hironobu, & Asakawa, Chieko. 2000. Transcoding Proxy For Nonvisual Web Access. *Pages 164–171 of: ASSETS '00: Proceedings of the 4th International ACM SIGACCESS Conference on Computers and Accessibility.* ACM Press.

Takagi, Hironobu, Asakawa, Chieko, Fukuda, Kentarou, & Maeda, Junji. 2002. Site-Wide Annotation: Reconstructing Existing Pages To Be Accessible. *Pages 81–88 of: ASSETS '02: Proceedings of the 5th International ACM SIGACCESS Conference on Computers and Accessibility.* ACM Press.

Takagi, Hironobu, Asakawa, Chieko, Fukuda, Kentarou, & Maeda, Junji. 2004. Accessibility Designer: Visualizing Usability For The Blind. *Pages 177–184 of: ASSETS '04: Proceedings of the 6th International ACM SIGACCESS Conference on Computers and Accessibility.* ACM Press.

Takagi, Hironobu, Saito, Shin, Fukuda, Kentarou, & Asakawa, Chieko. 2007. Analysis of Navigability of Web Applications for Improving Blind Usability. *ACM Transactions on Computer-Human Interaction*, **14**(3), 13. ISSN: 1073-0516.

Thiessen, Peter, & Chen, Charles. 2007. Ajax Live Regions: Chat As a Case Example. *Pages 7–14 of: W4A '07: Proceedings of the 2007 International Cross-Disciplinary Workshop on Web Accessibility (W4A).* ACM Press.

Thompson, Henry S., Beech, David, Maloney, Murray, & Mendelsohn, Noah. 2004 (October). *XML Schema Part 1: Structures.* W3C Recommendation. World Wide Web Consortium (W3C). `http://www.w3.org/TR/xmlschema-1/`. Last Accessed 3rd October 2008.

Trewin, Shari, Keates, Simeon, & Moffatt, Karyn. 2006. Developing Steady Clicks: A Method of Cursor Assistance for People With Motor Impairments. *Pages 26–33 of: ASSETS '06: Proceedings of the 8th International ACM SIGACCESS Conference on Computers and Accessibility.* ACM.

Tsarkov, Dmitry, & Horrocks, Ian. 2006. FaCT++ Description Logic Reasoner: System Description. *Pages 292–297 of: IJCAR 2006: Proceedings of the 3rd International Joint Conference on Automated Reasoning.* Springer Berlin / Heidelberg.

UK Government. 1995 (April). *Disability Discrimination Act 1995.* `http://www.opsi.gov.uk/acts/acts1995/1995050.htm`. Last Accessed 17th September 2008.

von Ahn, L., Blum, M., Hopper, N.J., & Langford, J. 2003. CAPTCHA: Using Hard AI Problems for Security. *Pages 294–311 of: EUROCRYPT 2003: Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques.* Springer Berlin / Heidelberg.

Wandmacher, Tonio, Antoine, Jean-Yves, & Poirier, Franck. 2007. SIBYLLE: A System for Alternative Communication Adapting to the Context and Its User. *Pages 203–210 of: ASSETS '07: Proceedings of the 9th International ACM SIGACCESS Conference on Computers and Accessibility.* ACM.

White, Ryen W., Kules, Bill, Drucker, Steven M., & m.c. schraefel. 2006. Supporting Exploratory Search. *Communications of the ACM*, **49**(4), 37–39.

Winer, Dave. 2007 (April). *RSS 2.0 Specification.* Specification. Berkman Center for Internet & Society at Harvard Law School. `http://cyber.law.harvard.edu/rss/`. Last Accessed 16th February 2009.

Yesilada, Yeliz, Harper, Simon, Goble, Carole, & Stevens, Robert. 2004. DANTE Annotation and Transformation of Web Pages for Visually Impaired Users. *Pages 490–491 of: WWW Alt. '04: Proceedings Of The 13th International World Wide Web Conference On Alternate Track Papers & Posters.* ACM Press. ISBN - 1-58113-912-8.

Yesilada, Yeliz, Bechhofer, Sean, & Horan, Bernard. 2007a (August). *COHSE: Dynamic Linking of Web Resources.* Technical Reports and Perspectives Essay Series TR-2007-167. Sun Microsystems Laboratories.

Yesilada, Yeliz, Stevens, Robert, Harper, Simon, & Goble, Carole. 2007b. Evaluating DANTE: Semantic Transcoding for Visually Disabled Users. *ACM Transactions on Computer-Human Interaction (TOCHI)*, **14**(3), 14.

Yin, Xinyi, & Lee, Wee Sun. 2004. Using Link Analysis To Improve Layout On Mobile Devices. *Pages 338–344 of: WWW '04: Proceedings of the 13th International Conference on the World Wide Web*. ACM Press.

Appendix A

SADIe Upper Ontology

```
Namespace:  <http://hcw.cs.manchester.ac.uk/experiments/sadie/ontologies/SADIe.owl#>
Namespace:  rdfs <http://www.w3.org/2000/01/rdf-schema#>
Namespace:  swrl <http://www.w3.org/2003/11/swrl#>
Namespace:  protege <http://protege.stanford.edu/plugins/owl/protege#>
Namespace:  owl2xml <http://www.w3.org/2006/12/owl2-xml#>
Namespace:  xsp <http://www.owl-ontologies.com/2005/08/07/xsp.owl#>
Namespace:  owl <http://www.w3.org/2002/07/owl#>
Namespace:  xsd <http://www.w3.org/2001/XMLSchema#>
Namespace:  swrlb <http://www.w3.org/2003/11/swrlb#>
Namespace:  rdf <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
Namespace:  SADIe <http://hcw.cs.manchester.ac.uk/experiments/sadie/ontologies/SADIe.owl#>

Ontology:  <http://hcw.cs.manchester.ac.uk/experiments/sadie/ontologies/SADIe.owl>

ObjectProperty:  isRemovable
    Characteristics:  Functional
    Range:  Boolean

ObjectProperty:  isConcertinaFullDetails
    Characteristics:  Functional
    Range:  Boolean

ObjectProperty:  isConcertinaOverview
    Characteristics:  Functional
    Range:  Boolean

ObjectProperty:  isMainMenu
    Characteristics:  Functional
    Range:  Boolean

ObjectProperty:  isSubMenu
    Characteristics:  Functional
    Range:  Boolean

ObjectProperty:  hasPriorityValue
    Characteristics:  Functional
    Range:  Integer
```

```
Class:  ConcertinaOverview
    EquivalentTo:  isConcertinaOverview some True
    SubClassOf:  ConcertinaMenu
    DisjointWith:  ConcertinaFullDetails

Class:  Fifteen
    SubClassOf:  Integer
    DisjointWith:  Five, Ten, Twelve, Eight, Eleven, One, Three, Four, Seven, Two,
        Thirteen, Nine, Fourteen, Six

Class:  LowPriority
    EquivalentTo:  hasPriorityValue some (Five or Four or One or Three or Two)
    SubClassOf:  Priority
    DisjointWith:  HighPriority, MediumPriority

Class:  MainMenu
    EquivalentTo:  isMainMenu some True
    SubClassOf:  Menu
    DisjointWith:  SubMenu

Class:  NonRemovable
    SubClassOf:  WebElement
    DisjointWith:  Removable

Class:  Priority
    SubClassOf:  NonRemovable

Class:  Eight
    SubClassOf:  Integer
    DisjointWith:  Five, Ten, Fifteen, Twelve, One, Eleven, Three, Four, Seven, Two,
        Thirteen, Nine, Fourteen, Six

Class:  ConcertinaFullDetails
    EquivalentTo:  isConcertinaFullDetails some True
    SubClassOf:  ConcertinaMenu
    DisjointWith:  ConcertinaOverview

Class:  Integer
    EquivalentTo:  Eight or Eleven or Fifteen or Five or Four or Fourteen or Nine or One
        or Seven or Six or Ten or Thirteen or Three or Twelve or Two
    SubClassOf:  DataTypes

Class:  Eleven
    SubClassOf:  Integer
    DisjointWith:  Five, Ten, Fifteen, Twelve, Eight, One, Three, Four, Seven, Two,
        Thirteen, Nine, Fourteen, Six

Class:  Boolean
    EquivalentTo:  False or True
    SubClassOf:  DataTypes

Class:  Three
    SubClassOf:  Integer
```

```
    DisjointWith:  Five, Ten, Fifteen, Twelve, Eight, Eleven, One, Four, Seven, Two,
        Thirteen, Nine, Fourteen, Six

Class:  True
    SubClassOf:  Boolean
    DisjointWith:  False

Class:  Four
    SubClassOf:  Integer
    DisjointWith:  Five, Ten, Fifteen, Twelve, Eight, One, Eleven, Three, Seven, Two,
        Thirteen, Nine, Fourteen, Six

Class:  Seven
    SubClassOf:  Integer
    DisjointWith:  Five, Ten, Fifteen, Twelve, Eight, One, Eleven, Three, Four, Two,
        Thirteen, Nine, Fourteen, Six

Class:  Two
    SubClassOf:  Integer
    DisjointWith:  Five, Ten, Fifteen, Twelve, Eight, Eleven, One, Three, Four, Seven,
        Thirteen, Nine, Fourteen, Six

Class:  SubMenu
    EquivalentTo:  isSubMenu some True
    SubClassOf:  Menu
    DisjointWith:  MainMenu, ConcertinaMenu

Class:  Thirteen
    SubClassOf:  Integer
    DisjointWith:  Five, Ten, Fifteen, Twelve, Eight, Eleven, One, Three, Four, Seven,
        Two, Nine, Fourteen, Six

Class:  Nine
    SubClassOf:  Integer
    DisjointWith:  Five, Ten, Fifteen, Twelve, Eight, One, Eleven, Three, Four, Seven,
        Two, Thirteen, Fourteen, Six

Class:  False
    SubClassOf:  Boolean
    DisjointWith:  True

Class:  Fourteen
    SubClassOf:  Integer
    DisjointWith:  Five, Ten, Fifteen, Twelve, Eight, One, Eleven, Three, Four, Seven, Two,
        Thirteen, Nine, Six

Class:  Six
    SubClassOf:  Integer
    DisjointWith:  Five, Ten, Fifteen, Twelve, Eight, Eleven, One, Three, Four, Seven, Two,
        Thirteen, Nine, Fourteen

Class:  Five
    SubClassOf:  Integer
```

```
    DisjointWithTen, Fifteen, Twelve, Eight, Eleven, One, Three, Four, Seven, Two,
        Thirteen, Nine, Fourteen, Six

Class:  Ten
    SubClassOf:  Integer
    DisjointWith:  Five, Fifteen, Twelve, Eight, One, Eleven, Three, Four, Seven, Two,
        Thirteen, Nine, Fourteen, Six

Class:  DataTypes
    SubClassOf:  SADIe

Class:  Twelve
    SubClassOf:  Integer
    DisjointWith:  Five, Ten, Fifteen, Eight, Eleven, One, Three, Four, Seven, Two,
        Thirteen, Nine, Fourteen, Six

Class:  SADIe

Class:  MediumPriority
    EquivalentTo:  hasPriorityValue some (Eight or Nine or Seven or Six or Ten)
    SubClassOf:  Priority
    DisjointWith:  HighPriority, LowPriority

Class:  WebElement
    SubClassOf:  SADIe

Class:  One
    SubClassOf:  Integer
    DisjointWith:  Five, Ten, Fifteen, Twelve, Eight, Eleven, Three, Four, Seven, Two,
        Thirteen, Nine, Fourteen, Six

Class:  HighPriority
    EquivalentTo:  hasPriorityValue some (Eleven or Fifteen or Fourteen or Thirteen or Twelve)
    SubClassOf:  Priority
    DisjointWith:  LowPriority, MediumPriority

Class:  ConcertinaMenu
    SubClassOf:  Menu
    DisjointWith:  SubMenu

Class:  Removable
    EquivalentTo:  isRemovable some True
    SubClassOf:  WebElement
    DisjointWith:  NonRemovable

Class:  Menu
    SubClassOf:  NonRemovable
```

Appendix B

CNN Ontology

```
Namespace:  <http://hcw.cs.manchester.ac.uk/experiments/sadie/ontologies/cnn.owl#>
Namespace:  protege <http://protege.stanford.edu/plugins/owl/protege#>
Namespace:  xsp <http://www.owl-ontologies.com/2005/08/07/xsp.owl#>
Namespace:  SADIe <http://hcw.cs.manchester.ac.uk/experiments/sadie/ontologies/SADIe.owl#>
Namespace:  cnn <http://hcw.cs.manchester.ac.uk/experiments/sadie/ontologies/cnn.owl#>
Namespace:  rdfs <http://www.w3.org/2000/01/rdf-schema#>
Namespace:  swrl <http://www.w3.org/2003/11/swrl#>
Namespace:  owl2xml <http://www.w3.org/2006/12/owl2-xml#>
Namespace:  owl <http://www.w3.org/2002/07/owl#>
Namespace:  xsd <http://www.w3.org/2001/XMLSchema#>
Namespace:  swrlb <http://www.w3.org/2003/11/swrlb#>
Namespace:  rdf <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
Ontology:  <http://hcw.cs.manchester.ac.uk/experiments/sadie/ontologies/cnn.owl>
Import:  <http://hcw.cs.manchester.ac.uk/experiments/sadie/ontologies/SADIe.owl>

ObjectProperty:  SADIe:isRemovable
ObjectProperty:  SADIe:isMainMenu
ObjectProperty:  SADIe:hasPriorityValue

Class:  cnnTopNewsModule
    SubClassOf:  CNN, SADIe:isRemovable some SADIe:True

Class:  cnnT1
    SubClassOf:  CNN, SADIe:hasPriorityValue some SADIe:Thirteen

Class:  cnnMpPartners
    SubClassOf:  CNN, SADIe:isRemovable some SADIe:True

Class:  csimanagerdiv
    SubClassOf:  CNN, SADIe:isRemovable some SADIe:True

Class:  SADIe:Eight

Class:  cnnTopics
    SubClassOf:  CNN, SADIe:isRemovable some SADIe:True

Class:  cnnPadMpMoreNews
```

```
    SubClassOf:  CNN, SADIe:hasPriorityValue some SADIe:Eight

Class:  cnnSuperBox
    SubClassOf:  CNN, SADIe:isRemovable some SADIe:True

Class:  cnnPartnerMod
    SubClassOf:  CNN, SADIe:isRemovable some SADIe:True

Class:  cnnSR3ItemPhoto
    SubClassOf:  CNN, SADIe:isRemovable some SADIe:True

Class:  cnnContentHeader
    SubClassOf:  CNN, SADIe:isRemovable some SADIe:True

Class:  cnnStoryPhotoBox
    SubClassOf:  CNN, SADIe:isRemovable some SADIe:True

Class:  cnnSnapShotHeader
    SubClassOf:  CNN, SADIe:isRemovable some SADIe:True

Class:  cnnSectTopStories
    SubClassOf:  CNN, SADIe:hasPriorityValue some SADIe:Eight

Class:  cnnHeaderContent
    SubClassOf:  CNN, SADIe:isRemovable some SADIe:True

Class:  cnnWCBoxHeader
    SubClassOf:  CNN, SADIe:isRemovable some SADIe:True

Class:  cnnSectionNews
    SubClassOf:  CNN, SADIe:isRemovable some SADIe:True

Class:  cnnSCFontButtons
    SubClassOf:  CNN, SADIe:isRemovable some SADIe:True

Class:  cnnT1cCol2
    SubClassOf:  CNN, SADIe:hasPriorityValue some SADIe:Thirteen

Class:  cnnSR3Sponsor
    SubClassOf:  CNN, SADIe:isRemovable some SADIe:True

Class:  cnnIrptBox
    SubClassOf:  CNN, SADIe:isRemovable some SADIe:True

Class:  cnnFooter
    SubClassOf:  CNN, SADIe:isRemovable some SADIe:True

Class:  CNN

Class:  cnnStoryToolsFooter
    SubClassOf:  CNN, SADIe:isRemovable some SADIe:True

Class:  TargetImageDE
    SubClassOf:  CNN, SADIe:isRemovable some SADIe:True
```

```
Class:  cnnT1cCol1
    SubClassOf:  CNN, SADIe:isRemovable some SADIe:True

Class:  cnnVideoCmpnt
    SubClassOf:  CNN, SADIe:isRemovable some SADIe:True

Class:  cnnHnT1
    SubClassOf:  CNN, SADIe:hasPriorityValue some SADIe:Thirteen

Class:  cnnRightCol
    SubClassOf:  CNN, SADIe:isRemovable some SADIe:True

Class:  cnn728Container
    SubClassOf:  CNN, SADIe:isRemovable some SADIe:True

Class:  cnnTabNav
    SubClassOf:  CNN, SADIe:isRemovable some SADIe:True

Class:  cnnHnAdTxt
    SubClassOf:  CNN, SADIe:isRemovable some SADIe:True

Class:  SADIe:True

Class:  cnnT1Img
    SubClassOf:  CNN, SADIe:isRemovable some SADIe:True

Class:  cnnNavigation
    SubClassOf:  CNN, SADIe:isMainMenu some SADIe:True

Class:  cnnUtilityNavigation
    SubClassOf:  CNN, SADIe:isRemovable some SADIe:True

Class:  SADIe:Thirteen

Class:  cnnPL
    SubClassOf:  CNN, SADIe:isRemovable some SADIe:True

Class:  cnn234Container
    SubClassOf:  CNN, SADIe:isRemovable some SADIe:True

Class:  ad-134608
    SubClassOf:  CNN, SADIe:isRemovable some SADIe:True

Class:  cnnPopNewsImg
    SubClassOf:  CNN, SADIe:isRemovable some SADIe:True

Class:  cnnPad12Top
    SubClassOf CNN

Class:  cnnTopStories
    SubClassOf:  CNN, SADIe:hasPriorityValue some SADIe:Eight
```

```
Class:  cnnUGCBox
    SubClassOf:  CNN, SADIe:isRemovable some SADIe:True

Class:  csimanagerdivdelayed
    SubClassOf:  CNN, SADIe:isRemovable some SADIe:True

Class:  cnnT2s
    SubClassOf:  CNN, SADIe:hasPriorityValue some SADIe:Thirteen

Class:  csiIframe
    SubClassOf:  CNN, SADIe:isRemovable some SADIe:True

Class:  cnnPad18Top
    SubClassOf:  CNN, SADIe:isRemovable some SADIe:True

Class:  cnnStoryElementBox
    SubClassOf:  CNN, SADIe:isRemovable some SADIe:True

Class:  cnnTopVideo
    SubClassOf:  CNN, SADIe:isRemovable some SADIe:True

Class:  cnnBannerContainer
    SubClassOf:  CNN, SADIe:isRemovable some SADIe:True

Class:  cnnTxtCmpnt
    SubClassOf:  CNN, SADIe:hasPriorityValue some SADIe:Thirteen

Class:  cnnPartHeader
    SubClassOf:  CNN, SADIe:hasPriorityValue some SADIe:Eight

Class:  TargetImage
    SubClassOf:  CNN, SADIe:isRemovable some SADIe:True

Class:  cnnSnapShot
    SubClassOf:  CNN, SADIe:hasPriorityValue some SADIe:Thirteen

Class:  cnnLinkBin
    SubClassOf:  CNN, SADIe:hasPriorityValue some SADIe:Eight

Class:  cnnPad18TRL
    SubClassOf:  CNN, SADIe:isRemovable some SADIe:True
```

Appendix C

Extended Cascading Style Sheet for CNN

```
.cnnCeilnav
   { -uom-structural-role: LinkList; }

ul.cnnNavigation
   { -uom-structural-role: LinkList; }

div.CNNhomeBox
   { -uom-structural-role: Chunk; }

ol.cnnMostPopular
   { -uom-structural-role: Chunk; }

div#cnnTopStoriesModule ul
   { -uom-structural-role: Chunk; }

div#cnnHeaderLeftCol h1
   { -uom-structural-role: Chunk; }

div#cnnTxtCmpnt p
   { -uom-structural-role: Chunk; }

div#cnnHeaderRightCol ul
   { -uom-structural-role: PageSummary; }
```

Appendix D

AxsJAX JavaScript Code for CNN

```
/**
* @title AxsJAX Enabled SADIeCNN
* @author Darren Lunn
* @version 1.0
* @date-created Tue Jun 09 15:13:51 BST 2008
* @last-modified Fri Aug 29 12:46:27 BST 2008
* @file-overview Code that will insert ARIA into the CNN pages using the AxsJAX framework
* developed by Google.  This script adds the lens functionality to highlight focussed areas.
*/

// create namespace
var axsSADIeCNN = {};

// The AxsJAX objects that will do the navigation and speaking.
axsSADIeCNN.axsJAXObj = null;
axsSADIeCNN.axsNavObj = null;
axsSADIeCNN.axsLensObj = null;

// The AxsLens object that will magnify content.
axsSADIeCNN.MAGNIFICATION_SIZE = 2.5;

// These are strings to be spoken to the user when they require help
axsSADIeCNN.NO_MENU_STRING = 'There is no Menu on this page';
axsSADIeCNN.NO_OVERVIEW_STRING = 'There is no overview information for this page';
axsSADIeCNN.HELP_STRING = 'The following shortcut keys are available.  ' +
      'M, main menu.  ' +
      'N, next paragraph of main text.  ' +
      'P, previous paragraph of main text.  ' +
      'O, overview of the page content.  ';



/**
* Initialises the object by creating the navigation and speech objects.
* We also define the main content elements so that users can easily
* navigate forwards and backwards between the chunks of the page.
*/
axsSADIeCNN.init = function()
      {
       // Defeine the AxsJAX speech object and the
       // navigation objects used within the page
       axsSADIeCNN.axsJAXObj = new AxsJAX(true);
       axsSADIeCNN.axsNavObj = new AxsNav(axsSADIeCNN.axsJAXObj);

       //Add event listeners to monitor the users key presses
       document.addEventListener('keypress', axsSADIeCNN.extraKeyboardNavHandler, true);
```

```
      // Create the XML that defines which elements are the main chunks of the page.
      // Chunks are identified using XPATH notation
      var cnrString = "<cnr>" +
            "<list title='Main Article' next='RIGHT j' prev='LEFT k' fwd='n' back='p'>" +
            "<item>" +
                  "//div[@id='cnnTxtCmpnt']//p | " +
                  "//div[@id='cnnTopStoriesModule']//ul | " +
                  "//div[@id='cnnHeaderLeftCol']//h1 | " +
                  "//div[@class='CNNhomeBox'] | " +
                  "//ol[@class='cnnMostPopular']" +
            "</item>" +
            "<target title='Open result' hotkey='ENTER'>" +
                  "./" +
            "</target>" +
            "</list>" +
            "</cnr>";

      // use the XML string we have just created to initialise the navigation
      axsSADIeCNN.axsNavObj.navInit(cnrString, null);

      // add a magnification lens to the page to highlight where the user is moving to.
      axsSADIeCNN.axsLensObj = new AxsLens(axsSADIeCNN.axsJAXObj);
      axsSADIeCNN.axsNavObj.setLens(axsSADIeCNN.axsLensObj);
      axsSADIeCNN.axsLensObj.setMagnification(axsSADIeCNN.MAGNIFICATION_SIZE);

      // Read the first thing on the page that we have defined as important.
      // Use a set time out just in case the browser is not entirely ready yet.
      window.setTimeout(axsSADIeCNN.readTheFirstThing,100);
      };


/**
* Reads the first thing on the page that we have deemed important.
*/
axsSADIeCNN.readTheFirstThing = function(evt)
      {
      // get the first element from the list of important chunks
      var firstElem = axsSADIeCNN.axsNavObj.nextItem().elem;

      // set the focus to the first element
      axsSADIeCNN.axsLensObj.view(firstElem);
      axsSADIeCNN.axsJAXObj.goTo(firstElem);
      };



/**
* Adds additional keyboard commands to allow users to jump between
* areas of the page.  This does not interfere with our list of important
* chunks we have defined above.  The AxsJAX framework remembers the location
* that the user was in before they left, allowing them to easily jump back
* to the content later.
*/
axsSADIeCNN.extraKeyboardNavHandler = function(evt)
      {
      // is the user is in an input field then pressing the esc key will
      // leave the current focus element and refocus on the main elements
      if (evt.keyCode == 27)
            {
            axsSADIeCNN.axsJAXObj.lastFocusedNode.blur();
            return false;
            }

      // is the user is in an input field then ignore any key press
      // except for ESC, which is caught above
      if (axsSADIeCNN.axsJAXObj.inputFocused)
            {return true;}

      // If Ctrl is held, it must be for some Assistive Technology function
      // so we ignore it.
      if (evt.ctrlKey)
            {return true;}
```

```
// if the user presses m then we set the focus to the main menu of
// the page.  There is only one but it is defined as a class.  We therefore get
// an array of all the list elements with a class of cnnNavigation and set the
// focus to the first (and only one) found.
if (evt.charCode == 109)
    {
     var menu;

     // get the node we want to start searching from and the class we're looking for
     var rootNode = document.getElementsByTagName('body')[0];
     var xpath = "//*[@class='cnnCeilnav']";

     // get the xpath value of the nodes and set the focus to the first one
     var menuArray = axsSADIeCNN.axsJAXObj.evalXPath(xpath, rootNode);
     menu = menuArray[0];

     // If we haven't found a menu node, find alternatives
     if (menu == null)
         {
          // get the node we want to start searching from and the class we're looking for
          var rootNode = document.getElementsByTagName('body')[0];
          var xpath = "//div[@class='cnnHeaderNav']";

          // get the xpath value of the nodes and set the focus to the first one
          var menuArray = axsSADIeCNN.axsJAXObj.evalXPath(xpath, rootNode);
          menu = menuArray[0];
         }

     // if a node has been found then change the focus to that node
     if (menu != null)
         {axsSADIeCNN.axsJAXObj.goTo(menu);}

     // if no node can be found then there is no menu so we inform the user
     // leaving the focus at the current element
     else
         {axsSADIeCNN.axsJAXObj.speakTextViaNode(axsSADIeCNN.NO_MENU_STRING);}

     return false;
    }

// if the user presses s then we set the focus on the overview of the
// page.  There is only one and it has an ID value so we use that to
// get the element.
if (evt.charCode == 115)
    {
     var overview;

     // get the node we want to start searching from and the class we're looking for
     var rootNode = document.getElementsByTagName('body')[0];
     var xpath = "//div[@id='cnnHeaderRightCol']//ul";

     // get the xpath value of the nodes and set the focus to the first one
     var overviewArray = axsSADIeCNN.axsJAXObj.evalXPath(xpath, rootNode);
     overview = overviewArray[0];

     // if a node has been found then change the focus to that node
     if (overview != null)
         {axsSADIeCNN.axsJAXObj.goTo(overview);}

     // if there was no node on the page that was designated a summary,
     // then see if there is a meta tag that we can use
     else
         {
          var pageSummary;

          // get the node we want to start searching from and the class we're looking for
          var rootNode = document.getElementsByTagName('head')[0];
          var xpath = "//meta";

          // get the XPath value of the nodes and set the focus to the first one
          var metaTagArray = axsSADIeCNN.axsJAXObj.evalXPath(xpath, rootNode);
```

```
                  // Deacription may be in lower case so if we haven't found anything, try again.
                  if (metaTagArray != null)
                       {
                        // create a variable in which to add a meta tag if we find one
                        var theParagraph;

                        // loop around each tag
                        for (var i = 0; i < metaTagArray.length; i++)
                             {
                              theParagraph = metaTagArray[i];

                              // for all the attributes of the meta tag node, look for the content
                              for (var j = 0; j < theParagraph.attributes.length; j++)
                                   {
                                    if (theParagraph.attributes[j].nodeValue.toLowerCase() === 'description')
                                         {
                                          for (var k = 0; k < theParagraph.attributes.length; k++)
                                               {
                                                if (theParagraph.attributes[k].nodeName.toLowerCase() === 'content')
                                                     {pageSummary = " " + theParagraph.attributes[k].nodeValue;}
                                               }
                                         }
                                   }
                             }
                       }

                  // if a node has been found then change the focus to that node
                  if (pageSummary != null)
                       {axsSADIeCNN.axsJAXObj.speakTextViaNode(pageSummary);}

                  // if no node can be found then there is no overview so we inform the user
                  // leaving the focus at the current element
                  else
                       {axsSADIeCNN.axsJAXObj.speakTextViaNode(axsSADIeCNN.NO_OVERVIEW_STRING);}
                 }

            return false;
           }

      // if the user presses ?  (question mark) then we read out the help
      // string that is associated with this class and page
      if (evt.charCode == 63)
           {
            axsSADIeCNN.axsJAXObj.speakTextViaNode(axsSADIeCNN.HELP_STRING);
            return false;
           }

      // if there is no keys identified then we return true and
      // keep the focus on the currently main element
      return true;
     };


/**
* Initialises the class and sets up the AxsJAX framework so that the
* user can start interacting with the page
*/
axsSADIeCNN.init();
```

Appendix E

Technical Evaluation Websites

Table E.1: A list of the forty Websites used in the SADIe prototype technical evaluation and the result of transcoding the index page. The CSS Usage indicates if the Website used pure CSS for style, a mixture of tables and CSS or no CSS styling. The categories are those defined by [Amitay *et al.* 2003].

Category	***Website***	***URL***	***CSS***	***Result***
Corporate Sites	Microsoft Corporation	`http://www.microsoft.com/`	Mix	Success
	Digital Designs	`http://www.digitaldesign.us`	Pure	Success
	Stagecoach Buses	`http://www.stagecoachbus.com/`	Pure	Success
	British Nuclear Fuels	`http://www.bnfl.com/`	Pure	Success
	Epson Corporation	`http://www.epson.co.jp/e/`	Mix	Success
Content & Media	Blogger	`http://www.blogger.com/`	Pure	Success
	The Mac Observer	`http://www.macobserver.com/`	Pure	Success
	New Musical Express	`http://www.nme.com/`	Mix	Failure
	BBC News	`http://news.bbc.co.uk/`	Mix	Failure
	CNN International	`http://edition.cnn.com/`	Mix	Failure
Search Engines	Google	`http://www.google.co.uk/`	None	Failure
	Yahoo	`http://uk.yahoo.com/`	Mix	Success
	Ask Jeeves	`http://www.askjeeves.co.uk/`	Mix	Success
	MSN Search	`http://search.msn.com/`	Pure	Success
	HotBot	`http://www.hotbot.co.uk/`	Pure	Success
Directories	Google Directory	`http://directory.google.co.uk/`	None	Failure
	Yahoo Directory	`http://uk.dir.yahoo.com/`	None	Failure
	This Is Our Year	`http://www.thisisouryear.com/`	Mix	Success
	HotSheet	`http://www.hotsheet.com/`	Pure	Success
	HaaBaa Web Directory	`http://www.haabaa.com/`	Mix	Success
Portals	AOL UK	`http://www.aol.co.uk/`	Mix	Success
	MSN UK	`http://www.msn.co.uk/`	Mix	Success

Continued on next page

Table E.1 – continued from previous page

Category	*Website*	*URL*	*CSS*	*Result*
	Wanadoo	http://www.wanadoo.co.uk/	Mix	Success
	Virgin Net	http://www.virgin.net/	Mix	Success
	Tiscali UK	http://www.tiscali.co.uk/	Pure	Success
E-stores	Play	http://www.play.com/	Mix	Success
	Amazon UK	http://www.amazon.co.uk/	None	Failure
	Tiny	http://www.tiny.com/	Mix	Success
	Tesco	http://www.tesco.com/	Mix	Success
	Red Letter Days	http://www.redletterdays.co.uk/	Mix	Success
Virtual Hosting	Bravenet	http://www.bravenet.com/	Mix	Success
	InMotion Hosting	http://www.inmotionhosting.com/	None	Failure
	Path Host	http://www.pathhost.net/	Mix	Success
	Honest Web Host	http://www.honestwebhost.com/	Mix	Failure
	Netwalker Internet	http://www.netwalker.biz/	Mix	Success
Universities	Manchester University	http://www.manchester.ac.uk/	Mix	Success
	York University	http://www.york.ac.uk/	Mix	Success
	Sheffield University	http://www.shef.ac.uk/	Mix	Success
	Oxford University	http://www.ox.ac.uk/	Mix	Success
	Southampton University	http://www.soton.ac.uk/	Mix	Success

Appendix F

Unclassified Instances of Coping Strategies

Table F.1: The initial coding performed on the Henshaws' data. Unlike the example in Table 4.1 the data Excerpt is not shown.

Difficulty	***Strategy***
Can not proceed due to the need to select a language and a location on the front page.	Ask the observer for help.
Can not proceed due to difficulty locating the 'Continue' Button.	Move around the page until a button is found. This is pressed in the hope that it performs the correct action.
Unsure if the correct page is being displayed due to a mismatch between expected content and actual content.	Move around the page trying to identify cues (logo, title) to confirm that the correct Website is being displayed.
Difficulty reading a news story as it switches between three different articles.	Wait for the story to roll around again.
Difficulty navigating through the content and the menu of a page.	Persistently move around the page until information of interest is found. This involves reading lots of irrelevant information.
Accessing the main content of a search result.	Press the 'tab' key several times to get to the first search result.
Finding the search button located behind the input form to instigate the search.	Use the 'tab' button to go forward several times then go backwards several times.
Finding and image with no alternative description.	Give up.

Continued on next page

Table F.1 – continued from previous page

Difficulty	***Strategy***
Failure to enter text in a search form.	Turn Jaws into ‘Form Mode’
Finding search results	Keep pressing the ‘tab’ key in the hope of finding a link that sounds like a search result.
Lost in the navigation area of the page.	Close the browser window to start again at home page (Google).
Frustration of listening to a menu repeated for every page.	Turn Jaws off and struggle with ZoomText.
Understanding the difference between a link and text written as a link.	Try and click links until something happens.
Lost within large chunks of text.	Try to use the intra-page menu.
Confusion caused by the inter-page menu and the intra-page menu being close together.	Give up and scroll down the page line by line to find information.
Finding search results.	Rapidly scroll down to approximately where the results may be.
Finding search results.	Rapidly scroll down to approximately where the results may be.
Failure to enter text in a search form.	Turn Jaws into ‘Form Mode’.
Lost within the page.	Use back button to return to Google and follow the same link.
Confusion caused by the inter-page menu and the intra-page menu being close together.	Navigate between the two several times before deciding which one to use.
Lost within a large amount of text.	Scroll rapidly, pause to re-orientate and then scroll some more.
Unable to find information because the page is rendered wider than the browser window.	Scroll vertically until the horizontal scroll bar and scroll the window horizontally.
Listening to the entire menu before reaching the main content.	Sit and listen to the links, occasionally taking the headphones off for respite.
Listening to all the information on a page.	Sit and listen to everything.
Unsure if the correct page is being displayed due to a mismatch between expected content and actual content.	Return to Google and follow the same link.
Confused when the page content doesn’t change when a link is selected (the link being clicked was the link to the current page).	Assume the link is broken.

Continued on next page

Table F.1 – continued from previous page

Difficulty	***Strategy***
Unsure if the correct page is being displayed due to a mismatch between expected content and actual content.	Search around the page looking for a suitable cue ('Enter Website' link).
Unable to find a desired link.	Use prior knowledge of menu location to try and identify.
Mismatch between link title and discovered information.	Click several links that may lead to the desired information.
Unable to locate the button that allows a form to be submitted due to the page size being large than the browser window.	Ask the observer to increase the window size.
Unsure if the correct page is being displayed due to a mismatch between expected content and actual content.	Return to Google and follow the same link.
Unable to locate the menu.	Ask the observer to point it out.
Filling in forms that span multiple pages.	Perseverance.
Unable to find the main content.	Scroll down the page and listen to everything to find information.
Unable to find required information.	Leave the Website and go back to Google.
Loss of data in a form.	Physically abuse the keyboard and start again.

Appendix G

Henshaws' Study Two Coping Strategies

Table G.1: Validating the coping definition defined in Section 4.2 through a classification of the coping strategies observed at the second Henshaws' study.

	Strategy	*Response*	*Used By*	*Difficulty*	*Cause*	*Experience*	*Application*	*Client*	*Context*
1	Ask the observer for help.	Mixed	Single User	Content inaccessible beyond the skill of the user	Page Design	Beginner	Multiple Pages	Human	Browsing
2	Turn Jaws into 'Form Mode'	Problem-Focused	Single User	Failure to enter text into a form	Screen Reader	Beginner	Multiple pages	Screen Reader	Information Search
3	Return to Google and follow the same link.	Problem-Focused	Single User	Lost in the Web page.	Page Design	Beginner	Multiple Pages	Screen Reader	Browsing
4	Give up.	Emotion-Focused	Single User	Assistive Technology mixing JavaScript with Content.	Screen Reader	Beginner	Single Page	Screen Reader	Browsing
5	Give up.	Emotion-Focused	Single User	Content inaccessible beyond the skill of the user.	Page Design	Beginner	Single Page	Screen Reader	Browsing
6	Use Link List Dialogue Box	Problem-Focused	Single User	A large number of links on a page.	Page Design	Beginner	Multiple Pages	Screen Reader	Browsing
7	Use Link List Dialogue Box	Problem-Focused	Single User	Reaching the search results.	Page Design	Beginner	Multiple Pages	Screen Reader	Browsing
8	Use Heading List Dialogue Box	Problem-Focused	Single User	Significant amount of content on a page.	Page Design	Beginner	Multiple Pages	Screen Reader	Browsing

Appendix H

NoVA Coping Strategies

Table H.1: Validating the coping definition defined in Section 4.2 through a classification of the coping strategies observed during the NoVA study.

	Strategy	*Response*	*Used By*	*Difficulty*	*Cause*	*Experience*	*Application*	*Client*	*Context*
1	Click on the link in the hope that it goes to useful information	Problem-Focused	Single User	Bitmap Images used as links but with no labels to inform the user what they are	Page Design	NULL	Single Page	Screen Reader	Information Search
2	Ask the observer for help	Mixed	Community	Content inaccessible beyond the skill of the user	Page Design	NULL	Multiple Pages	Human	Information Search
3	Give up	Emotion-Focused	Community	Can not find desired information	Page Design	NULL	Multiple Pages	All Clients	Information Search
4	Give up	Emotion-Focused	Community	Source code read alongside page text	Screen Reader	NULL	Multiple Pages	Screen Reader	Information Search
5	Shout at the computer	Emotion-Focused	Single User	Screen Reader taking too long to read aloud the page	Screen Reader	NULL	Single page	Human	Information Search
6	Press ‘Insert + F4’ to force Jaws to reformat the page	Problem-Focused	Community	Trying to find information within the page	Page Design	NULL	Multiple Pages	Screen Reader	Information Search
7	Return to Alta Vista home page and start again	Problem-Focused	Community	Lost in the Web page.	Page Design	NULL	Multiple Pages	All Clients	Information Search
8	Return to MMU OPAC home page and start again	Problem-Focused	Single User	Lost in the Web page	Page Design	NULL	Single Page	Screen Reader	Information Search

Continued on next page

Table H.1 – continued from previous page

	Strategy	*Response*	*Used By*	*Difficulty*	*Cause*	*Experience*	*Application*	*Client*	*Context*
9	Jump to the bottom of the page then tab backwards to avoid the banner	Problem-Focused	Community	Finding Information	Page Design	NULL	Multiple Pages	Screen Reader	Information Search
10	Quickly tab down the page and memorise links that may be useful so they can be returned to and investigated thoroughly later	Problem-Focused	Community	Finding Information	Page Design	NULL	Single Page	Screen Reader	Information Search
11	Press "Insert + F7" to bring up the Links List Dialogue Box	Problem-Focused	Single User	Finding Information	Page Design	NULL	Multiple Pages	Screen Reader	Information Search
12	Listen to the computer and wait for it to stop making a noise	Problem-Focused	Single User	Discovering when a page has finished loading	Users' Client	NULL	Single Page	Human	Information Search
13	Move down to the status bar to see if it has finished moving	Problem-Focused	Community	Discovering when a page has finished loading	Users' Client	NULL	Single Page	All clients	Information Search
14	Try and arrow down the page. This will only happen when the page has finished loading	Problem-Focused	Single User	Discovering when a page has finished loading	Users' Client	NULL	Single Page	Screen Reader	Information Search
15	Try and click links until something happens.	Problem-Focused	Community	Confusion between links and text that looks like a link	All Clients	NULL	Multiple pages	Screen Reader	Information Search
16	Sit and listen to everything to find the desired information	Problem-Focused	Single User	Finding information within a page	Page design	NULL	Single page	Screen Reader	Information Search
17	Click several links that may lead to the desired information.	Problem-Focused	Community	Mismatch between link title and discovered information.	Page Design	NULL	Multiple Pages	Screen Reader	Information Search
18	Quickly tab down the page listening to partial words as the user knows what to expect	Problem-Focused	Single User	Finding information on a familiar page	Page Design	NULL	Single Page	Screen Reader	Information Search
19	Press "tab" to skip past two tables so access the correct information	Problem-Focused	Single User	Finding information on a page	Page Design	NULL	Single Page	Screen Reader	Information Search
20	Listen to everything on the page until the words "Terms and Conditions" are read aloud. This inductees the end of the search results	Problem-Focused	Single User	Finding information on a page	Page Design	NULL	Single Page	Screen Reader	Information Search
21	Press "Insert + T" to read aloud the page title	Problem-Focused	Single User	Confused as to what page is being accessed	Page Design	NULL	Single Page	Screen Reader	Information Search
22	Press "Alt + Enter" to remove unwanted links	Problem-Focused	Single User	Overwhelmed by the mount of information	Page Design	NULL	Single Page	Screen Reader	Information Search

Appendix I

Coping Strategies Grouped By Patterns

The following tables group the coping strategies previously identified in Table 4.4 on page 103, Table G.1 on page 190, and Table H.1 on page 191 that have similar goals. The ID of the patterns is of the format `data.number` where `data` represents the data set that the strategy came from with H1 representing the Henshaws' Period One data set; H2 representing the Henshaws' Period Two data set and N representing the NoVA evaluation data set. `Number` is the ID of the strategy from the given data set. For example H1.2 is coping strategy two from the Henshaws' Period One data set, as found in Table 4.4.

I.1 Candidate Chunk Discovery Strategy

Table I.1: Coping strategies belonging to the Candidate Chunk Discovery Strategy, discussed further in Section 4.4.1.

	Strategy	*Response*	*Used By*	*Difficulty*	*Cause*	*Experience*	*Application*	*Client*	*Context*
H1.11	Rapidly scroll to approximate location of the results, re-orientate and then scroll again.	Problem-Focused	Community	Finding information within a page.	Page Design	Intermediate	Multiple Pages	Screen Magnifier	Multiple Contexts
H1.14	Try to use the intra- page menu.	Problem-Focused	Single User	Lost within large chunks of text.	Page Design	Beginner	Single page	All Clients	Information Search
H1.15	Repeatedly navigate between two menus before choosing.	Problem-Focused	Single User	Confusion caused by inter- and intra-page menu being together.	Page Design	Beginner	Single page	All Clients	Information Search
H2.8	Use Heading List Dialogue Box	Problem-Focused	Single User	Significant amount of content on a page.	Page Design	Beginner	Multiple Pages	Screen Reader	Browsing
N.10	Quickly tab down the page and memorise links that may be useful so they can be returned to and investigated thoroughly later	Problem-Focused	Community	Finding Information	Page Design	NULL	Single Page	Screen Reader	Information Search
N.18	Quickly tab down the page listening to partial words as the user knows what to expect	Problem-Focused	Single User	Finding information on a familiar page	Page Design	NULL	Single Page	Screen Reader	Information Search
N.22	Press "Alt + Enter" to skip links and move to next block of text	Problem-Focused	Single User	Overwhelmed by the mount of information	Page Design	NULL	Single Page	Screen Reader	Information Search

I.2 Masthead Avoidance Strategy

Table I.2: Coping strategies belonging to the Masthead Avoidance Strategy, discussed further in Section 4.4.2.

	Strategy	*Response*	*Used By*	*Difficulty*	*Cause*	*Experience*	*Application*	*Client*	*Context*
H1.5	Press the 'tab' key several times to get to the first search result.	Problem-Focused	Community	Difficulty finding search results.	Page Design	Expert	Single Page	Screen Reader	Information Search
H2.6	Use Link List Dialogue Box	Problem-Focused	Single User	Reaching the search results.	Page Design	Beginner	Multiple Pages	Screen Reader	Browsing
N.9	Jump to the bottom of the page then tab backwards to avoid the banner	Problem-Focused	Community	Finding Information	Page Design	NULL	Multiple Pages	Screen Reader	Information Search

Continued on next page

Table I.2 – continued from previous page

	Strategy	*Response*	*Used By*	*Difficulty*	*Cause*	*Experience*	*Application*	*Client*	*Context*
N.19	Press "tab" to skip past two tables so access the correct information	Problem-Focused	Single User	Finding information on a page	Page Design	NULL	Single Page	Screen Reader	Information Search

I.3 Clustered Element Strategy

Table I.3: Coping strategies belonging to the Clustered Element Strategy, discussed further in Section 4.4.3.

	Strategy	*Response*	*Used By*	*Difficulty*	*Cause*	*Experience*	*Application*	*Client*	*Context*
H1.7	Use the 'tab' button to go forward several times then go backwards several times.	Problem-Focused	Single User	Finding the search button located behind the input form to instigate the search.	Page Design	Expert	Single Page	Screen Reader	Information Search
H1.17	Use prior knowledge of menu location to try and identify a link.	Problem-Focused	Single User	Unable to find a desired link.	Page Design	Intermediate	Single page	All Clients	Information Search
N.20	Listen to everything on the page until the words "Terms and Conditions" are read aloud. This inductees the end of the search results	Problem-Focused	Single User	Finding information on a page	Page Design	NULL	Single Page	Screen Reader	Information Search

I.4 Probing Strategy

Table I.4: Coping strategies belonging to the Probing Strategy, discussed further in Section 4.4.4.

	Strategy	*Response*	*Used By*	*Difficulty*	*Cause*	*Experience*	*Application*	*Client*	*Context*
H1.13	Click several links that may lead to the desired information.	Problem-Focused	Single User	Mismatch between link title and discovered information.	Page Design	Intermediate	Multiple Pages	All Clients	Multiple Contexts
H2.7	Use Link List Dialogue Box	Problem-Focused	Single User	A large number of links on a page.	Page Design	Beginner	Multiple Pages	Screen Reader	Browsing

Continued on next page

Table I.4 – continued from previous page

	Strategy	*Response*	*Used By*	*Difficulty*	*Cause*	*Experience*	*Application*	*Client*	*Context*
N.1	Click on the link in the hope that it goes to useful information	Problem-Focused	Single User	Bitmap Images used as links but with no labels to inform the user what they are	Page Design	NULL	Single Page	Screen Reader	Information Search
N.11	Press "Insert + F7" to bring up the Links List Dialogue Box	Problem-Focused	Single User	Finding Information	Page Design	NULL	Multiple Pages	Screen Reader	Information Search
N.17	Click several links that may lead to the desired information.	Problem-Focused	Community	Mismatch between link title and discovered information.	Page Design	NULL	Multiple Pages	Screen Reader	Information Search

I.5 Backtracking Strategy

Table I.5: Coping strategies belonging to the Backtracking Strategy, discussed further in Section 4.4.5.

	Strategy	*Response*	*Used By*	*Difficulty*	*Cause*	*Experience*	*Application*	*Client*	*Context*
H1.9	Return to Google and follow the same link.	Problem-Focused	Community	Lost in the Web page.	Page Design	Intermediate	Multiple Pages	All Clients	Multiple Contexts
H2.3	Return to Google and follow the same link.	Problem-Focused	Single User	Lost in the Web page.	Page Design	Beginner	Multiple Pages	Screen Reader	Browsing
N.7	Return to Alta Vista home page and start again	Problem-Focused	Community	Lost in the Web page.	Page Design	NULL	Multiple Pages	All Clients	Information Search
N.8	Return to MMU OPAC home page and start again	Problem-Focused	Single User	Lost in the Web page	Page Design	NULL	Single Page	Screen Reader	Information Search

I.6 Withdrawal Strategy

Table I.6: Coping strategies belonging to the Withdrawal Strategy, discussed further in Section 4.4.6.

	Strategy	*Response*	*Used By*	*Difficulty*	*Cause*	*Experience*	*Application*	*Client*	*Context*
H1.1	Ask the observer for help.	Mixed	Community	Content inaccessible beyond the skill of the user	Page Design	Beginner	Multiple Pages	Human	Multiple Contexts
H1.6	Physically abuse the keyboard and start again.	Emotion-Focused	Single User	Loss of data in a form.	Page Design	Beginner	Single Page	Computer	Information Search
H1.8	Give up.	Emotion-Focused	Single User	Finding and image with no alternative description.	Page Design	Expert	Single Page	Screen Reader	Information Search
H2.1	Ask the observer for help.	Mixed	Single User	Content inaccessible beyond the skill of the user	Page Design	Beginner	Multiple Pages	Human	Browsing
H2.4	Give up.	Emotion-Focused	Single User	Assistive Technology mixing JavaScript with Page Content.	Page Design	Beginner	Single Page	Screen Reader	Browsing
H2.5	Give up.	Emotion-Focused	Single User	Content inaccessible beyond the skill of the user.	Page Design	Beginner	Single Page	Screen Reader	Browsing
N.2	Ask the observer for help	Mixed	Community	Content inaccessible beyond the skill of the user	Page Design	NULL	Multiple Pages	Human	Information Search
N.3	Give up	Emotion-Focused	Community	Can not find desired information	Page Design	NULL	Multiple Pages	All Clients	Information Search
N.4	Give up	Emotion-Focused	Community	Source code read alongside page text	Screen Reader	NULL	Multiple Pages	Screen Reader	Information Search
N.5	Shout at the computer	Emotion-Focused	Single User	Screen Reader taking too long to read aloud the page	Screen Reader	NULL	Single page	Human	Information Search

I.7 Unclassified Strategies

Table I.7: Coping strategies that have yet to be classified to any abstract pattern.

	Strategy	*Response*	*Used By*	*Difficulty*	*Cause*	*Experience*	*Application*	*Client*	*Context*
H1.2	Turn Jaws into 'Form Mode'	Problem-Focused	Community	Failure to enter text into a form	Screen Reader	Beginner	Single page	Screen Reader	Information Search
H1.3	Move around the page trying to identify cues to confirm that the correct Website is being displayed.	Problem-Focused	Single User	Unsure if the correct page is being displayed	Page Design	Intermediate	Multiple Pages	All Clients	Multiple Contexts

Continued on next page

Table I.7 – continued from previous page

	Strategy	*Response*	*Used By*	*Difficulty*	*Cause*	*Experience*	*Application*	*Client*	*Context*
H1.4	Wait for the story to roll around again.	Emotion-Focused	Single User	Difficulty reading a news story as it switches between three different articles.	Page Design	Beginner	Single Page	Screen Magnifier	Browsing
H1.10	Turn Jaws off and struggle with ZoomText.	Problem-Focused	Single User	Frustration of listening to a menu repeated for every page.	Page Design	Beginner	Single Page	Screen Reader	Browsing
H1.12	Sit and listen to everything to Find Desired Information.	Emotion-Focused	Community	Finding information within a page.	Page Design	Intermediate	Multiple Pages	Screen Reader	Multiple Contexts
H1.16	Try and click links until something happens.	Problem-Focused	Single User	Confusion between links and text that looks like a link	Screen Magnifier	Beginner	Multiple pages	Screen Magnifier	Information Search
H1.18	Scroll vertically until the horizontal scroll bar then scroll horizontally.	Problem-Focused	Single User	Unable to find information because the page is rendered wider than the browser window.	Page Design	Beginner	Single page	Screen Magnifier	Browsing
H2.2	Turn Jaws into 'Form Mode'	Problem-Focused	Single User	Failure to enter text into a form	Page Design	Beginner	Multiple pages	Screen Reader	Information Search
N.6	Press 'Insert + F4' to force Jaws to reformat the page	Problem-Focused	Community	Trying to find information within the page	Page Design	NULL	Multiple Pages	Screen Reader	Information Search
N.12	Listen to the computer and wait for it to stop making a noise	Problem-Focused	Single User	Discovering when a page has finished loading	Users' Client	NULL	Single Page	Human	Information Search
N.13	Move down to the status bar to see if it has finished moving	Problem-Focused	Community	Discovering when a page has finished loading	Users' Client	NULL	Single Page	All clients	Information Search
N.14	Try and arrow down the page. This will only happen when the page has finished loading	Problem-Focused	Single User	Discovering when a page has finished loading	Users' Client	NULL	Single Page	Screen Reader	Information Search
N.15	Try and click links until something happens.	Problem-Focused	Community	Confusion between links and text that looks like a link	All Clients	NULL	Multiple pages	Screen Reader	Information Search
N.16	Sit and listen to everything to find the desired information	Problem-Focused	Single User	Finding information within a page	Page design	NULL	Single page	Screen Reader	Information Search
N.21	Press "Insert + T" to read aloud the page title	Problem-Focused	Single User	Confused as to what page is being accessed	Page Design	NULL	Single Page	Screen Reader	Information Search

Appendix J

User Evaluation Websites

Table J.1: Web pages used as stimuli in the user evaluations and the tasks participants were asked to complete.

Web Page	***URL***	***Question***
About	`http://www.about.com/`	What is the title of the main article?
AOL	`http://www.aol.com/`	What is the title of the first article?
BBC News	`http://news.bbc.co.uk/`	What is the title of the main news story?
Blogger	`http://matildaintherock.blogspot.com/`	What is the title of the first blog entry?
CNET	`http://www.cnet.com/`	What is the title of the first article?
CNN	`http://www.cnn.com/`	What is the title of the main news story?
Digg	`http://www.digg.com/`	What is the title of the first news story?
Go	`http://www.go.com/`	What is the title of the first news story?
Google	`http://www.google.com/`	What is the title of the first search result?
Information	`http://www.information.com/`	What is the title of the first search result?
Live Search	`http://www.live.com/`	What is the title of the first search result?
		Continued on next page

Table J.1 – continued from previous page

Web Page	***URL***	***Question***
Microsoft	`http://www.microsoft.com/`	What is the title of the first highlight article?
MSN	`http://www.msn.com/`	What is the title of the first entertainment story?
New York Times	`http://www.nytimes.com/`	What is the title of the first news story?
Rediff	`http://www.rediff.com/`	What is the title of the first headline story?
Reference	`http://www.reference.com/`	What is the title of the first search result?
Slashdot	`http://slashdot.live.com/`	What is the title of the first news story?
Wordpress	`http://aculturedleftfoot.wordpress.com/`	What is the title of the first blog entry?
Xanga	`http://www.xanga.com/LadyValkyrie37/`	What is the title of the first blog entry?
Yahoo! Search	`http://search.yahoo.com/`	What is the title of the first search result?

Appendix K

Technical Reports

The results of the research described in thesis have led to the following technical reports, shown in chronological order.

1. **A Review of Piggy Bank and How it Compares to SADIe** *Darren Lunn*. The University of Manchester, School of Computer Science, November 2005. `http://hcw-eprints.cs.manchester.ac.uk/58/`

   Piggy Bank is a tool that attempts to extract data from a Web page in order to provide users with some of the information retrieval power of the Semantic Web without fully adopting the tools and technologies that will be required for the Semantic Web to come into existence. SADIe is an approach that uses the structure of the Web page in order to apply transcoding that better suits the accessibility needs of the user. This report looked at what Piggy Bank could achieve with the data extracted from the page and how this compared with SADIe.

2. **SADIe: Structural-Semantics for Accessibility and Device Independence** *Darren Lunn*. The University of Manchester, School of Computer Science, September 2006. `http://hcw-eprints.cs.manchester.ac.uk/59/`

   This report described the underlying principles of the SADIe modelling approach which involves annotating the rendering information contained within the Cascading Style Sheet of a Website. Such an approach allows an accurate description of the elements to be provided, yet is highly scalable, due to the association between the pages contained within Website and the

Website's CSS. A single annotation propagates from the rendering information through to all the pages contained within the Website. This report was submitted to the University of Manchester for the transferral from the Master of Philosophy to the Doctor of Philosophy degree program in the Faculty of Engineering and Physical Sciences.

3. **Randomization Testing And Why It Is Appropriate For SADIe** *Darren Lunn*. The University of Manchester, School of Computer Science, February 2007. `http://hcw-eprints.cs.manchester.ac.uk/60/`

   The evaluations conducted using the SADIe Transcoding Proxy were not suitable for the conditions needed to satisfy classical parametric testing requirements. These requirements include the use of large data sets chosen at random from the candidate population and the need for theoretical distribution models of the evaluation tests that are being conducted. This report looked at why SADIe failed to meet those requirements and presented a case for the use of randomization testing. These tests make no assumptions about the data obtained during the evaluation and can be applied to small data sets.

4. **Observational Notes Acquired from Henshaws' Skillstep to Success Class: Observation Period 1** *Darren Lunn and Eleni Michailidou*. The University of Manchester, School of Computer Science, July 2007. `http://hcw-eprints.cs.manchester.ac.uk/61/`

   To model coping strategies, seven two-hour sessions were spent at Henshaws' Society for Blind People in order to conduct summative experimental studies. This report provides the anonymised data obtained from the first set of sessions, during which time the authors discussed accessibility issues with students and staff of the class in addition to studying students accessing the Web.

5. **CASTA: Coping Strategy Analysis to Support Transcoding Algorithms: Analysis of Henshaws' Observational Data** *Darren Lunn*. The University of Manchester, School of Computer Science, December 2007. `http://hcw-eprints.cs.manchester.ac.uk/62/`

   As a first step to understanding coping strategies, several sessions of an IT class organised by Henshaws' Society for Blind People were observed.

This report provides an analysis of the raw anecdotal and quantitative data that was obtained during those observational sessions. By looking at coping strategies used from a number of dimensions, an initial model and classification of coping strategies was presented.

6. **Building Ontologies For The SADIe Transcoder** *Darren Lunn.* The University of Manchester, School of Computer Science, January 2008. `http://hcw-eprints.cs.manchester.ac.uk/63/`

   The prototype SADIe transcoding proxy creates a sequential audio rendering of the page that is better suited to screen readers employed by visually impaired users. The transcoding that achieves this is driven by an ontology that explicates implicit information held within the page. The ontology consists of two parts. The first is an upper level ontology containing high level abstract concepts representing the potential roles of Web page elements. The second part of the ontology is a Website specific extension to the upper ontology. This paper provides an overview of the SADIe ontology structure and details how Website developers can build ontologies for their Websites that will operate correctly with SADIe.

7. **Observational Notes Acquired from Henshaws' Skillstep to Success Class: Observation Period 2** *Darren Lunn and Eleni Michailidou.* The University of Manchester, School of Computer Science, February 2008. `http://hcw-eprints.cs.manchester.ac.uk/64/`

   To model coping strategies, four two-hour sessions were spent at Henshaws' Society for Blind People in order to conduct summative experimental studies. This report provides the anonymised data obtained from the second set of sessions, during which time the authors discussed accessibility issues with students and staff of the class in addition to studying students accessing the Web.

8. **A Quantitative Evaluation of The SADIe Transcoder: Measuring Time To Task Completion** *Darren Lunn.* The University of Manchester, School of Computer Science, July 2008. `http://hcw-eprints.cs.manchester.ac.uk/54/`

   This report presents a user evaluation that was performed using the SADIe transcoding prototype. Users were presented with a series of Web pages, some having been adapted using SADIe's shallow transcoding functionality and others retaining their original state. Participants were asked to provide answers to fact-based questions, allowing a quantitative comparison between access to transcoded and non-transcoded Web pages to be made. Statistical analyses using both randomization testing and t-tests indicate that SADIe can improve the time users can reach information within a Web page.

9. **Verification of The Coping Strategy Framework Through An Analysis of the NoVA Evaluation Data** *Darren Lunn.* The University of Manchester, School of Computer Science, July 2008. `http://hcw-eprints.cs.manchester.ac.uk/67/`

   In previous reports coping strategies were analysed and an initial model produced that can be used to identify and categorise coping strategies. This report describes a validation of the model whereby independent data taken from an evaluation of the NoVA Project was taken and successfully applied to the coping strategy model. The results show that the model is flexible and accurate enough to be applied to third party data and validates the approach taken.

10. **Coping Strategy Pattern Identification: An Analysis of the Henshaws' and NoVA Data** *Darren Lunn.* The University of Manchester, School of Computer Science, August 2008. `http://hcw-eprints.cs.manchester.ac.uk/69/`

    In previous reports a model of coping strategies was defined and verified with independent data. This report discusses the application of the coping model to three data sets. The strategies that were captured were then analysed and six recurring patterns were identified from the data. These recurring abstract patterns can now be used to form the basis of algorithms designed to reduce the need for coping strategies and move transcoders towards the deeper levels of transcoding that we desire.

11. **A Quantitative Evaluation of The SADIe Transcoder With Sighted Users: Measuring Time To Task Completion** *Darren Lunn.* The University of Manchester, School of Computer Science, November 2008. `http://hcw-eprints.cs.manchester.ac.uk/78/`

This report presents a user evaluation that was performed using the SADIe prototype transcoding system with sighted users. Users were presented with a series of Web pages that retained their original state. Participants were asked to provide answers to fact-based questions, allowing a quantitative comparison between access to content by sighted user and access to Web pages by visually impaired users to be made.

12. **A Quantitative Evaluation of The SADIe Transcoder Using Firevox: Measuring Time To Task Completion** *Darren Lunn.* The University of Manchester, School of Computer Science, November 2008. `http://hcw-eprints.cs.manchester.ac.uk/77/`

    This report presents a user evaluation that was performed using the SADIe transcoding prototype woth participants using the Firevox self-voicing Web browser. Users were presented with a series of Web pages, some having been adapted using SADIe's shallow transcoding functionality and others retaining their original state. Participants were asked to provide answers to fact-based questions, allowing a quantitative comparison between access to transcoded and non-transcoded Web pages to be made. Statistical analyses using t-tests indicate that SADIe can significantly improve the time users can reach information within a Web page.

13. **A Quantitative Evaluation of Deep SADIe Transcoding: Measuring Time To Task Completion** *Darren Lunn.* The University of Manchester, School of Computer Science, December 2008. `http://hcw-eprints.cs.manchester.ac.uk/78/`

    This report presents a user evaluation that was performed using the SADIe prototype transcoding system. Users were presented with a series of Web pages, some having been adapted using SADIe's deep transcoding functionality and others retaining their original state. Participants were asked to provide answers to fact-based questions, allowing a quantitative comparison between access to transcoded and non-transcoded Web pages to be made. Statistical analyses using t-tests indicate that SADIe's deep transcoding can improve the time users can reach information within a Web page.