Requirements for web developers and web commissioners in ubiquitous Web 2.0 design and development

Deliverable 3.2 :: Public

Keywords: web design and development, Web 2.0, accessibility, disabled web users, older web users
## Table of Contents

Glossary of abbreviations ............................................................................................................. 6  
Executive Summary ......................................................................................................................... 7  
1 Introduction ................................................................................................................................ 12  
  1.1 Terminology ............................................................................................................................. 13  
2 Requirements for Web commissioners .......................................................................................... 15  
  2.1 Introduction ............................................................................................................................... 15  
  2.2 Previous work ............................................................................................................................ 15  
  2.3 Method ..................................................................................................................................... 17  
    2.3.1 Participants .......................................................................................................................... 17  
    2.3.2 Interview schedule .............................................................................................................. 18  
    2.3.3 Procedure .......................................................................................................................... 18  
  2.4 Results ..................................................................................................................................... 18  
    2.4.1 Range of organizations ....................................................................................................... 18  
    2.4.2 The websites and web applications .................................................................................... 19  
    2.4.3 Organization of web design and development .................................................................... 19  
    2.4.4 Website audiences .............................................................................................................. 20  
    2.4.5 Policy on website accessibility ............................................................................................. 20  
    2.4.6 Knowledge of and training in web accessibility and related concepts ............................... 20  
    2.4.7 Quality processes for web accessibility .............................................................................. 21  
    2.4.8 Quality processes for web usability ..................................................................................... 21  
  2.5 Discussion and Conclusions ...................................................................................................... 22  
3 Requirements for web developers ............................................................................................... 24  
  3.1 Introduction ................................................................................................................................ 24  
  3.2 Previous work ............................................................................................................................ 24  
  3.3 Survey of Web Developer Practices .......................................................................................... 27  
    3.3.1 Method ................................................................................................................................ 27  
    3.3.2 Results ................................................................................................................................ 29  
  3.4 Contextual Inquiries with Web Developers ............................................................................ 35  
    3.4.1 Method ................................................................................................................................ 35  
    3.4.2 Results ................................................................................................................................ 39  
    3.4.3 Discussion and Conclusions ............................................................................................... 50  
4 Requirements for Web accessibility experts .............................................................................. 54  
  4.1 Introduction ................................................................................................................................ 54  
  4.2 Method ..................................................................................................................................... 54
4.2.1 Participants ........................................................................................................... 54
4.2.2 Interview schedule ............................................................................................... 55
4.2.3 Procedure .............................................................................................................. 56
4.2.4 Data Analysis ....................................................................................................... 56
4.3 Results ...................................................................................................................... 56
4.3.1 Training and Experience ....................................................................................... 56
4.3.2 Participation in Guidelines/Standards Bodies ....................................................... 57
4.3.3 Tool Functionality ................................................................................................. 57
4.3.4 Completeness of accessibility tests ....................................................................... 60
4.3.5 Sampling Pages for Accessibility Testing ............................................................. 60
4.3.6 Tracking of Accessibility Tests and Problems ..................................................... 61
4.4 Common Accessibility Problems ............................................................................ 62
4.4.1 Reporting Accessibility Problems ........................................................................ 63
4.4.2 Maintenance of Accessibility ............................................................................... 64
4.5 Discussion and Conclusions .................................................................................... 64
5 References .................................................................................................................. 66
Appendix A: Interview schedule for website commissioners ........................................ 68
Appendix B: Survey of Web Developer Practices ........................................................ 73
Appendix C: Contextual inquiry schedule on “Accessibility support for web developers”  81
Appendix D: Web Accessibility Evaluator Interview Schedule ..................................... 90
### List of Tables

Table 2.1: Frequency of website features ................................................................. 19  
Table 2.2: Knowledge/Training in web technologies, accessibility and usability ........ 21  
Table 3.1: Age distribution of respondents .............................................................. 28  
Table 3.2: Experience distribution of respondents .................................................. 29  
Table 3.3: Features for which the largest proportion of respondents prefer to use an IDE ... 32  
Table 3.4: Features for which the largest proportion of respondents prefer to use some other application ................................................................. 33  
Table 3.5: Features for which the largest proportion of respondents do not use .......... 33  
Table 3.6: Technologies for which the largest proportion of respondents prefer to use an IDE ................................................................. 33  
Table 3.7: Technologies for which the largest proportion of respondents prefer to use a text editor ................................................................. 33  
Table 3.8: Technologies for which the largest proportion of respondents prefer to use some other application ................................................................. 34  
Table 3.9: Technologies for which the largest proportion of respondents do not use ....... 34  
Table 4.1: Age breakdown of interviewees .............................................................. 54  
Table 4.2: Functionality used in WATTs and AATTs ................................................ 58
Requirements for web developers and web commissioners in ubiquitous Web 2.0 design and development
I2Web project (Grant no.: 257623)

| Requirements for web developers and web commissioners in ubiquitous Web 2.0 design and development |
| Project: I2Web (Grant no.: 257623) |
| Version 2.0 |
| Page 5 of 92 |
| © I2Web Consortium 2011 |

| Contractual Date of Delivery to the EC: | 30th April 2011 + 60 days |
| Actual Date of Delivery to the EC: | 6th June 2011, 9th January 2012 (revised) |
| Editors: | Helen Petrie, Chris Power, David Swallow (YORK) |
| Contributors: | I2Web Consortium |

**DOCUMENT HISTORY**

<table>
<thead>
<tr>
<th>Version</th>
<th>Version date</th>
<th>Responsible</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>23 March 2011</td>
<td>York</td>
<td>Table of Contents</td>
</tr>
<tr>
<td>B</td>
<td>29 April 2011</td>
<td>York</td>
<td>First draft</td>
</tr>
<tr>
<td>C</td>
<td>10 May 2011</td>
<td>York</td>
<td>Second draft</td>
</tr>
<tr>
<td>D</td>
<td>6 June 2011</td>
<td>York</td>
<td>Final Version</td>
</tr>
<tr>
<td>2.0</td>
<td>9 January 2012</td>
<td>York</td>
<td>New version in response to 6M Review request.</td>
</tr>
</tbody>
</table>

The information and views set out in this publication are those of the author(s) and do not necessarily reflect the official opinion of the European Communities. Neither the European Union institutions and bodies nor any person acting on their behalf may be held responsible for the use which may be made of the information contained therein.

Copyright © 2011 I2Web Consortium.

Permission is granted to copy, distribute and/or modify this document text and its contained artworks under the Creative Commons Attribution License 3.0 as described at [http://creativecommons.org/licenses/by/3.0](http://creativecommons.org/licenses/by/3.0).
Glossary of abbreviations

**AATT:** automatic accessibility testing tool

**CFIT:** Centre for Inclusive Technology, part of NCBI

**CMS:** Content Management System

**CSS:** Cascading Style Sheet

**DOM:** Document Object Model

**IDE:** Integrated Development Environment

**nWATT:** non-web accessibility testing tool

**WAI:** Web Accessibility Initiative of the W3C (q.v.)

**WATT:** Web accessibility testing tool

**WCAG1** and **WCAG2:** Web Content Accessibility Guidelines, Version 1 and Version 2

**W3C:** World Wide Web Consortium
Executive Summary

At the moment it is clear that much Web content and many Web 2.0 applications are not accessible to disabled and older users (Kane, Shulman, Shockley, and Ladner, 2007; Lazar et al, 2011). If this situation is to change, there is a chain of stakeholders who need to be both aware of accessibility issues and able to play their appropriate role in creating accessible Web content and Web 2.0 applications. This chain consists of:

- Individuals who commission, own or manage websites (for simplicity called “website commissioners” in this deliverable);
- Individuals who design and develop websites and applications, usually commissioned by individuals in the first group;
- Individuals who assess the accessibility of websites and applications, either commissioned by either of the previous two groups, or by an external organization, such as a government or independent organisation that is benchmarking the accessibility of websites.

This deliverable presents three studies which conducted in-depth interviews with members of each of these stakeholder groups, in order to investigate their understanding of web accessibility in the Web2.0 context and their requirements for support in playing their appropriate roles in creating accessible Web2.0 content and applications.

Chapter 2 presents the in-depth interviews with 26 website commissioners, drawn from across the private, public and third sectors, and responsible for a range of websites and applications from the modest to the very large. The websites include a range of complex features that are challenges for accessibility, particularly the use of media players (found in 61.5% of the websites) and link sharing for social network sites (found in 57.7% of the websites). The use of CMSs was very common, with 88.5% of the websites using them. There were a number of comments from participants indicating that they rely on the CMS to ensure accessibility of the website. This strategy is only as good as the CMS used and its ability to ensure accessibility correctly. Web commissioners need to be able to evaluate accessibility independently of their CMS, and in the case of several participants, this was clearly not happening.

Participants’ organizations often out-source part or all of the website work (34.6% total outsourcing, 46.2% partial outsourcing). Out-sourcing of web design and development work lead to a number of issues around accessibility. Several participants remarked that companies claimed to have expertise in accessibility, but they found this hard to judge independently. In addition, out-sourcing could create accessibility-related conflicts, as one participant found to their cost, when their design company proposed a design for the website that the implementation company argued would be inaccessible (unfortunately, design won over accessibility).

In terms of disabled and older people as audiences for websites, only 11.5% of participants spontaneously mentioned these groups as potential audiences for their website. When prompted, 69.2% agreed that these groups might be potential audiences for the website. These results show that website commissioners are not really thinking about the potential audiences of their websites, as disabled and older people are almost always possible
audiences for websites. This means that accessibility will often be omitted from the agenda in the commissioning of websites.

In contrast, when asked whether their organization has a policy on accessibility of the website for disabled and older people, 61.5% said they did. This difference between the question about audiences and the question about web accessibility policy probably shows the subject bias effect one gets in asking questions about web accessibility. It may also show the lack of realism in the minds of most people about people with disabilities and older people – they fail to realize the range of things that disabled and older people may wish to do. An example was a school for children with disabilities who did not think their website needed to be accessible to disabled or older people – although parents of the pupils were a key target audience. Could not the parent of a disabled child be disabled themselves or could not the guardian of a disabled child be their grandparent?

Most website commissioners (88.5%) correctly understood the meaning of web accessibility with somewhat less showing a correct understanding of web usability (69.2%). About half the participants had some training or knowledge of web technologies, web accessibility and web usability. However, only a third had any training in web accessibility and usability and had picked up information on the job. In retrospective, it would have been better to ask separate questions about training and knowledge, to tease out this difference. Even the figure of one third of web commissioners having knowledge of web accessibility may over estimate their knowledge, as a number of the participants had worked with the CFIT, part of NCBI and had training or knowledge in web accessibility from this collaboration.

Finally, participants were asked about their web quality processes for accessibility and usability. For web accessibility, more than half (61.5%) of participants said that they (or the companies they out-source to) assess the accessibility of their website. However, it was worrying that only a quarter of these participants could say what kind of testing had been done. This does not indicate a good understanding of the process. And only 7.7% of participants named a WCAG level conformance that they were aiming for/were tested against. Few participants had a formal process for ensuring that accessibility problems identified in testing were addressed and that accessibility of the website was maintained going forward from the testing. This would indicate a clear gap in the support provided to website commissioners in tracking accessibility of their site.

Results for web usability quality control were very similar and participants appeared to deal with accessibility and usability together. This may be a “hook” that those concerned with web accessibility can use to convince website commissioners to implement quality control for accessibility – by bundling it with usability and showing that commissioners can achieve two goals with one system or tool.

Chapter 3 presents the methods used to investigate the working practices of web developers along with initial results of the analysis of the data collected. This aspect of the investigation began with an online survey completed by 60 web developers that provided an initial understanding of the technologies, tools and techniques used. The full survey used in the investigation can be found in Appendix B: Survey of Web Developer Practices This was followed up by a series of in-depth contextual inquiries with 13 web developers, which allowed the complex work practices of this cohort to be explored. The full contextual inquiry schedule used in the investigation can be found in Appendix C: Contextual inquiry schedule on “Accessibility support for web developers”
The points of particular interest for the i2Web Project that emerged from these contextual inquiries with web developers are:

- Despite acknowledging that some accessibility problems simply cannot be “programmatically determined”, almost all of the developers want a clear, quantitative checklist of accessibility problems.

- Developers want accessibility tools to provide not only solutions and suggestions to improve the code but also the reasoning behind them. This could be in the form of a bullet-point headline list of problems in clear “human” language along with the more technical code suggestions behind each point. Simple videos of disabled users of websites are also considered to be a very effective educational tool.

The integration of accessibility testing into existing unit testing is popular with developers. Citing the Selenium web application testing system as an example, developers suggest that accessibility unit tests could be written and performed automatically numerous times at various stages of development.

- Few developers want accessibility tools to be integrated into their text editor. The stripped down, simple environment of such tools is what often attracts the developer to them in the first place. Developers who use IDEs are more open to the possibility of accessibility tool integration.

- Many developers expressed a desire for accessibility tools that are incorporated into their browser. This is where they already preview and test their websites and is the stage of the web development process during which the developers would appreciate accessibility information.

- Should accessibility tools be incorporated into their IDE or text editor, developers are keen to retain as much control as possible. Developers feel that inline active accessibility tools in the code editor - working similar to a spell-checker - could prove very annoying. Developers prefer to ‘action’ an accessibility check themselves and review the suggestions, at a stage in the development process that they feel appropriate. Furthermore, developers are suspicious about any form of “wizard” interface or avatar assistance (such as ‘Clippy’ in Microsoft Office), assuming that any code generated by such tools would be incorrect and have to be manually refined.

- Developers are open to the idea of simulating the experiences of web users with disabilities as a tool in developing accessible websites and applications. However, they are cautious about the amount of time it would take to run and, in particular, to interpret the results from such simulations. As well as simulating disabilities and impairments (such as partial sight or colour blindness), developers are keen to reduce costs by simulating popular (and expensive) assistive technologies (e.g. the output from a screenreader).

Overall these in-depth contextual inquiries produced much detailed information about the work practices and accessibility requirements of web developers that had not been elicited in previous surveys and interviews.

Chapter 4 presents the in-depth interviews with 14 web accessibility evaluators. The results indicate that there is currently a gap in how tools support accessibility evaluators and the reality of practice of accessibility evaluators. In this section, the requirements and implications for design are presented for new tools in supporting accessibility evaluation.
The statements from evaluators regarding AATTs are particularly important. The current tools do not provide adequate information to the evaluator about what the tools test or what the problems are on a webpage. These statements are supported by the fact that so few evaluators report using AATTs. If new AATTs are to be widely adopted they must provide more information about what is being tested. It is not sufficient for a tool to simply “dump out” a set of problems to an evaluator. It must be possible for the evaluator to query what tests are being performed, and engage in a dialogue with the tool regarding how decisions are made by a tool. All of this information must be presented in the language of the evaluators and frame the results in terms of how the problems are likely to impact the user, and in terms of repair of the issue.

Further to this, most evaluators feel that they bring more knowledge and experience to an evaluation than can be captured in an AATT. When new automated tools are developed, they must be created in such a way that the evaluator can contribute their knowledge into the system in order to eliminate false positives or false negatives. For example, assume that an AATT returns a list of alternative text warnings, such as having an empty (alt=””) text for an image. Many will be decorative images not needing an alternative text, thus creating a false positive. However, in the first instance of such a test, the evaluator must manually check all warnings. In current AATTs, each time the automated test is run, all warnings are produced, and as a result each error may need to be checked by an evaluator each time. In a future AATT one could image a case where the evaluator could annotate the information about which warnings are real and which are false positives for propagation to the next time the tests are run. In this way, such manual checking would only need to be done the first time after elements on web pages change.

In terms of manual testing, there are a large number of evaluators who are relying on multiple tools in order to get the complete set of features needed to undertake evaluations. The use of different toolbars, browsers and development tools (e.g. Firebug) in an ad hoc way makes it very difficult to integrate different tests and related reports together. A more unified approach, or at least a unified view of these different tools through a common interface, is something that would likely be welcomed by evaluators.

Web accessibility evaluators are in need of tools that will help them manage the pages they have been asked to test and what tests have been undertaken on those pages. These features must be flexible enough to accommodate different strategies that evaluators have when undertaking an evaluation. For example, in some cases evaluators will take the traditional approach of performing all tests on one page, and that is one strategy that should continue to be supported. However, the tools should also support the approach of applying one test (e.g. checking for alternative texts on images) to all pages in sequence. Tools that help in these seemingly mundane and tedious tasks will allow evaluators who are working in opportunistic and/or scrambled styles of evaluation to move their practices towards more strategic approaches. Hopefully, this will reduce the overall potential for missing problems in a given set of pages and increase the reliability of an individual evaluator.

The comments regarding the lack of training that many evaluators have, in conjunction with the rapidly changing technology environment, show a need to provide more structured support and help during evaluation sessions through tools. If it is the case that evaluators are reluctant to engage in formal training, and yet there are still issues that evaluators do not understand, then tools that providing comprehensive guidance and structured dialogues for repair would be of value.
It is absolutely essential that any future tools that are produced be able to generate a variety of different formats of reports. Reports that only contain lists of violations found in tests are not of interest to the majority of evaluators or their clients. While it is important to still provide this feature in situations where an organization must have a complete audit of their web application/websites against a set of guidelines, it is equally important that tools support annotation of problems by evaluators. Further, having the ability of collecting together similar problems between pages and then annotate those problems with examples or solutions, is a feature that would be received well by the evaluator community.
1 Introduction

At the moment it is clear that much Web content and many Web 2.0 applications are not accessible to disabled and older users (Kane, Shulman, Shockley, and Ladner, 2007; Lazar et al, 2011). If this situation is to change, there is a chain of stakeholders who need to be both aware of accessibility issues and able to play their appropriate role in creating accessible Web content and Web 2.0 application and services.

The first link in this chain is the individuals who commission, own and manage websites and Web 2.0 applications. In this deliverable we will refer to these individuals as “website commissioners” for brevity. These individuals may not understand a great deal about the technicality of the websites and applications, and in particular, they may not understand the technicalities of how these are created to be accessible. But they do need to be able to understand the general issues of accessibility and to monitor whether the websites and applications they are responsible for are indeed accessible. This might involve running or understanding the output from accessibility testing of websites and applications, and discussions with web developers and web accessibility evaluators.

The second group in the chain is individuals who design and develop websites and Web 2.0 applications (we will refer to this group as web developers for brevity). Some of these individuals may do only the design of a website or application, such as the layout and colour schemes. They should know about aspects of accessibility such as easy to read fonts and good colour contrast for partially sighted people, colour combinations not suitable for people with colour vision deficiencies and line length and spacing requirements for people with dyslexia. They need tools to help them assess whether their designs are going to be accessible. Developers are those individuals who actually code a website, application or service. They need to understand all the “nitty gritty” of how to code for accessibility, from marking up tables correctly to ensuring that Flash animations are coded appropriately. They need tools to help them produce accessible code most effectively and efficiently.

The third group in the chain are the accessibility experts, who may well advise the first two groups in the chain, and who test websites, applications and services for accessibility. They may be retained by commissioners or developers or they may be working for external services that are benchmarking websites and applications, such as government agencies or representatives of disabled and older people. They need to understand all the complexities of technical accessibility. Nonetheless, they need tools to support them in assessing the accessibility of websites and applications effectively and efficiently, and also to be able to communicate the results of these assessments to commissioners and developers in ways that will be understandable to these individuals who may have different levels of understanding of accessibility.

This deliverable will present the results of in-depth interviews with samples of individuals from each of these three groups in the chain to achieve accessible websites and applications. These results will provide the i2Web project with a greater understanding of the needs of each of these groups and requirements for the development of tools to support each group.
1.1 Terminology

For purposes of this deliverable, a webpage is a collection of content that can be presented through a web browser to a user. A webpage may consist of several different pieces of content, such as text, images or multimedia, and can be heterogeneous in the technology used to integrate the content into the webpage.

A web application is a collection of webpages that are designed to allow users to achieve a particular goal. This could be a set of webpages that are static in nature, through which users obtain information, or a more complex dynamic interaction where the user needs to contribute information to the system. This second type of application could include simple or complex forms or more complex web content such as games or interactive portals.

A website is a collection of web applications under one domain name or belonging to one company.

A web accessibility evaluator is someone who undertakes testing for accessibility on websites and web applications.

“Manual accessibility testing” (sometimes shortened to “manual testing”) is any process where the human web accessibility evaluator (sometimes shortened to “accessibility evaluator”) makes the decision about whether something is an accessibility issue or not. To do this, they may use web accessibility testing tools (WATTs) to provide information to help them make this decision. These tools are often plug-ins to web browsers and include:

- WAVE Toolbar
- Web Accessibility Toolbar from the Paciello Group
- AIS Web Accessibility Toolbar

These tools allow a human web accessibility evaluator to work with the source code to assess accessibility issues. For example, a tool can find all the `<img>` elements on a web page and indicate whether they have an alt text or not. However, a human accessibility evaluator decides whether the alt text, if provided, is appropriate for the `<img>`. Accessibility evaluators also often use testing tools not specifically developed for accessibility testing but for other web development purposes (non-web accessibility testing tools, nWATTs), such as:

- Firebug
- Web Developer Toolbar for Mozilla
- Opera Dragonfly
- Code inspector for Safari

---

1 [http://wave.webaim.org/toolbar](http://wave.webaim.org/toolbar) (retrieved April 25 2011)
For example, many WATTs allow the evaluator to isolate specific lines of code; however, many nWATTs provide more useful functionality, and allow the accessibility evaluator to dynamically move from a line of code to view and access the elements that encapsulate it, or allow the accessibility evaluator to view the Document Object Model (DOM) directly.

“Automatic accessibility testing” (sometimes shortened to “automatic testing”) is any process where an algorithm makes the decision about whether something is an accessibility problem or not. For example, an algorithm can decide whether an <img> element has an alt text or not. The first and still most famous automatic accessibility testing tool (AATT) was Bobby⁹, now no longer available. An extensive list of AATTs is available at the WAI website¹⁰. AATTs inevitably only assess a subset of web accessibility problems, as many problems require a human judgement, as discussed above.

¹⁰http://www.w3.org/WAI/ER/tools/complete.html (retrieved April 25 2011)
2 Requirements for Web commissioners

2.1 Introduction

Web commissioners\(^{11}\) are the individuals who are responsible for the accessibility of a website or application or set of websites. In many countries in the European Union and beyond this is now a legal responsibility\(^{12}\). However, web commissioners may have little or no knowledge about the complex area of web accessibility and how to assess it in relation to their websites and applications.

This chapter presents a study that involved in-depth interviews with a range of individuals who commission, own or manage websites to investigate their understanding of web accessibility issues and their requirements for tools and information to support them in playing their role appropriately in ensuring the accessibility of the websites and/or applications for which they are responsible.

2.2 Previous work

Little research has focussed on the understanding and needs of website commissioners. The Disability Rights Commission (2004) undertook a survey of website commissioners and web developers, to assess their knowledge of and interest in accessibility. 712 website commissioners were approached by letter to complete a short questionnaire, but there was only a 9% response rate (64 responses). The questionnaire was therefore followed up with face-to-face interviews with 21 website commissioners. The questionnaire responses showed considerable differences between those employed by larger organizations (defined as more than 250 employees) and smaller organizations. Those responding from large organizations appeared to be well aware of web accessibility issues:

- 97% reported that their organization thought that accessibility was an important issue;
- 88% claimed to be aware of their responsibilities under the Disability Discrimination Act\(^{13}\);
- 76% stated that their organization had policies on website accessibility;
- 68% reported that they took accessibility into account when developing a website;
- 71% reported that they had conducted some form of accessibility testing of their website; and
- 88% said they had plans to make improvements to the accessibility of the website.

The responses from those in smaller organizations were less encouraging:

---

\(^{11}\) As mentioned in the Introduction, we will use the term “web commissioners” to refer to all individuals who commission, own or manager a website or web application.

\(^{12}\) For information on legislation in different countries, see [www.w3.org/WAI/Policy](http://www.w3.org/WAI/Policy) (retrieved 25 April 2011)

\(^{13}\) Relevant UK legislation at that time to ensure the rights of people with disabilities.
69% reported that their organization was aware of website accessibility as an issue;
48% claimed to be aware of their responsibilities under the Disability Discrimination Act;
34% reported that the organization had policies on website accessibility;
29% reported that they took accessibility into account when developing a site;
17% reported that they had conducted some form of accessibility testing of their website; and
58% reported that they had plans to make improvements to the accessibility of the site.

The interview responses reinforced the impression of differences between large and smaller organizations: 9 of the 21 website commissioners interviewed were from smaller organizations, 7 of these (78%) were completely unaware of web accessibility as an issue. One thought that “web accessibility” referred to the number of Internet cafes and similar public facilities for people to use the Web.

These results were at odds to the findings on the assessment of the accessibility of 1000 websites in Great Britain undertaken by the DRC at the same time, which covered those commissioned by small, medium and large organizations. This assessment found that only 19% of websites met even the most basic accessibility requirements (WCAG1 Level A). If 68% of large organizations and 29% of medium and small organizations take accessibility into account when developing their site, one would have expected far better results from the websites.

The BenToWeb Project (2006) conducted a European-wide online survey of web commissioners with a follow-up sample of respondents being interviewed by phone to gather more in-depth information. 139 people responded to the online survey, from 11 different European countries. Some of the key results were:

50.4% of respondents reported that their organization was “very interested” in web accessibility, with no differences reported due to sector or organizational size

22.3% of respondents reported that their organization was “very knowledgeable” about web accessibility, with public sector organizations being significantly more knowledgeable than private sector organizations, and large organizations being significantly more knowledgeable than small organization

33.3% of respondents reported that their organization was “very knowledgeable” about understanding the needs of disabled web users, with large organization being significantly more knowledgeable than small organization

45.6% of respondents reported that their organization had a policy regarding the accessibility of their website

7.5% of respondents reported that their organization’s website was “very accessible”, with respondents from large organizations reporting significantly higher levels than respondents from small organizations
No other research has been carried out into the understanding of web accessibility of website commissioners and their needs in relation to understanding, assessing and maintaining web accessibility of their websites and applications.

2.3 Method

2.3.1 Participants

An opportunistic sample of participants was used. FAST, NCBI and York all recruited participants for the study. Potential participants were told that the interview was about “web quality processes”, so that they were not cued to the fact that it was about web accessibility. This would have created “subject bias” (Rosnow and Rosenthal, 1997), that is they would answer in ways that they think the interviewer wants them to answer. This is probably inevitable to a certain extent in this study, as all three organizations that recruited participants and conducted the interviews are well known for their commitments to people with disabilities.

Participants were not offered any recompense for participating in the study, apart from a copy of the report on the results of the study if they were interested.

Methods of recruitment were as follows:

- At FAST, ten organizations were approached via email to participate in the study, and to promote it through their relevant networks. These were selected from FAST's networks and contacts as having particular responsibility for website commissioning. A total of ten web commissioners agreed to participate in the study. (Six of whom were previously known to FAST).

- At NCBI, 28 organizations in both the public (state and semi state) and private sector were contacted in relation to this study and invited to participate. All contacts were made via the Centre For Inclusive Technology database at NCBI. An email containing information about the project, this study and what was expected from them was sent to potential participants. If they consented to participate they were asked to reply by email and an NCBI staff member would make contact and a time would be arranged to conduct the interview. Fourteen companies agreed to participate within the time frame (some companies consented too late for inclusion).

- At York, 10 organizations in the public, private and third sector were contacted by email or phone and invited to participate in the study. These were organizations that York has some relationship with (not necessarily in relation to accessibility work).

26 website commissioners participated in this study. 3 (11.5%) work in the private sector, 13 (50%) in the public sector and 10 (38.5%) in the third sector (for charities, NGOs etc.). 10 (38.5%) work for very small organizations (49 employees or less), 7 (26.9%) for small organizations (50 to 250 employees) and 9 (34.6%) for large organizations (more than 250 employees).
2.3.2 Interview schedule

A schedule of interview questions was developed in an iterative process with contributions from all partners. A number of versions of the interview schedule were produced to cover all the relevant questions, without being excessively lengthy for participants.

The schedule covered the following topics:

- The organization
- The website, and web applications if relevant
- The interviewee, particularly their knowledge/training in web technologies, accessibility and usability
- Web quality processes

As the interviewer did not want to cue the interviewee that the interview was about web accessibility in particular, the recruitment information and interview schedule was presented as being about web quality processes. This was reasonable, as web accessibility is one aspect of web quality. Therefore questions about web usability as well as web accessibility were asked.

A pilot with several participants also led to refinements in the questions and the wording of particular questions. The final interview schedule is provided in Appendix A: Interview schedule for website commissioners.

2.3.3 Procedure

Participants were interviewed either face-to-face or by phone. Participants were given an introduction to the i2Web Project and its aims. They were assured of the confidentiality of the information they would be providing. Where possible, interviews were recorded for later transcription, but in some cases it was only possible to take notes during the interview. The interviewer worked through the interview schedule, asking all appropriate questions and allowing the participant to elaborate as much as possible on their answers. Interviews lasted between 30 and 45 minutes. When the interview schedule was complete, participants were asked whether they had any further comments they would like to make about their experience of web accessibility. Finally they were thanked for their time and asked whether they would like a copy of the report on the results of the study.

2.4 Results

The following sections outline the results of the interview data collected.

2.4.1 Range of organizations

As mentioned above, 3 of the participants (11.5%) work in the private sector, 13 (50%) in the public sector and 10 (38.5%) in the third sector (for charities, NGOs etc.). 10 (38.5%) work for very small organizations (49 employees or less), 7 (26.9%) for small organizations (50 to 250 employees) and 9 (34.6%) for large organizations (more than 250 employees). Thus a range of organization types and sizes is represented in the sample.
2.4.2 The websites and web applications

In most instances (92.3%, 24), the website which the participant is responsible for is the primary website for the organization. In some cases, participants were responsible for a number of websites for their organization.

Websites ranged in size from 25 to approximately 750,000 pages, with an average of 48,530 (standard deviation = 187,122) pages. The number of visitors per month ranged from approximately 1500 to approximately 430,000 with an average of 74,600 (standard deviation = 107,131).

In terms of the features offered by the websites and applications, Table 2.1, below shows the percentage and number of participants who reported having these features. This table shows that the most commonly used website features are media players (with associated multimedia content) (reported by 61.5% of participants) and link sharing for social media websites (reported by 57.7% of participants).

Table 2.1: Frequency of website features

<table>
<thead>
<tr>
<th>Web feature</th>
<th>% (Number) of participants reporting this feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content that refreshes automatically</td>
<td>34.6 (9)</td>
</tr>
<tr>
<td>Media player and multimedia content</td>
<td>61.5 (16)</td>
</tr>
<tr>
<td>Interactive menus</td>
<td>50.0 (13)</td>
</tr>
<tr>
<td>Comments from visitors</td>
<td>23.1 (6)</td>
</tr>
<tr>
<td>Flash content</td>
<td>19.2 (5)</td>
</tr>
<tr>
<td>Social network site feeds</td>
<td>42.3 (11)</td>
</tr>
<tr>
<td>Link sharing for social network sites</td>
<td>57.7 (15)</td>
</tr>
<tr>
<td>Blog or wiki</td>
<td>38.5 (10)</td>
</tr>
<tr>
<td>Financial transactions</td>
<td>50.0 (13)</td>
</tr>
</tbody>
</table>

The use of content management systems (CMSs) was very common. 88.5% (23) participants reported that their website used a CMS. A wide variety of CMSs were mentioned, with Terminal 4\(^{14}\) being the only one mentioned more than once (4 times in total). Interestingly, Terminal 4 does highlight the accessibility of their CMS.

2.4.3 Organization of web design and development

Participants were asked whether the website they are responsible for is developed in-house or out-sourced or a combination of these two. 19.2% (5) participants reported total in-house development, 34.6% (9) reported totally out-sourcing, and 46.2% (12) reported a combination of the two.

\(^{14}\) [http://www.terminalfour.com/](http://www.terminalfour.com/)
Often participants’ organizations out-sourced the initial design of the website and the CMS. Sometimes one company was commissioned to do all this work, sometimes more than one (usually two, but in one case three companies). With out-sourcing, organizations were sometimes left unsure about what work on accessibility or accessibility evaluation was being done. Interviewees sometimes spoke of the fact that organizations claimed to know about accessibility, but this had to be taken on trust.

In one case out-sourcing to multiple companies lead to a particularly interesting situation with regard to accessibility. One company was commissioned to come up with a design for the website, another company was commissioned to implement the design. The second company said that the design would not be accessible, particularly with respect to colour contrasts. This was debated between the design company, the development company, the participant and the director of the company. Design won over accessibility. And the participant concluded that they now have a website that is not “particularly easy to read for anyone”. So the participant has learnt a good lesson in inclusive design, but it is not clear that the design company or the director of the participant’s company have learnt this lesson.

2.4.4 Website audiences

Participants were asked about the intended audiences for their website. Only 11.5% (3) mentioned disabled or older people unprompted. However, when asked explicitly, 69.2% (18) said that disabled or older people might be part of the audience.

2.4.5 Policy on website accessibility

Participants were asked whether their organization has a policy on the accessibility of the website for disabled and older people. 61.5% (16) said they did, although sometimes this was not a formal policy, but a general understanding.

2.4.6 Knowledge of and training in web accessibility and related concepts

Participants were asked a number of questions to assess their knowledge of web accessibility and related concepts.

Participants were asked what “web accessibility” means to them. 88.5% (23) of participants had an appropriate understanding of this concept. Participants were also asked what “web usability” means to them. Interestingly, only 69.2% (18) had an appropriate concept of this concept.

Participants were asked what knowledge and/or training they have of web technologies, web accessibility and web usability\(^\text{15}\). Table 2.2, below, summarizes the results for these questions. About half the participants had training and/or knowledge in all three areas. However, these similar percentages hide the fact participants were much more likely to have formal training and qualifications in web technologies (73.3%, 11 of the 15 who said training or knowledge of web technologies) than in either web accessibility (33.3%, 5 of the 15 who said they had knowledge of web accessibility) or usability (33.3%, 5 of the 15 who

\(^\text{15}\) Web usability is in fact beyond the scope of the project, but questions about this topic were included to follow through the idea that that interview was about web quality processes in a broader sense than web accessibility. We report the results on these questions, as they may be of interest.
said they had knowledge of web usability). A number of participants had worked with the Centre for Inclusive Technology (CFIT) at the NCBI and had either attended short courses on accessibility with them or learnt about accessibility through activities such as watching user testing of their website. These are excellent practices, but may mean that the level of knowledge about web accessibility is higher in this sample than amongst web commissioners in general.

Table 2.2: Knowledge/Training in web technologies, accessibility and usability

<table>
<thead>
<tr>
<th>Topic</th>
<th>Percentage (Number) of participants with training/knowledge</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web technologies</td>
<td>57.7% (15)</td>
</tr>
<tr>
<td>Web accessibility</td>
<td>57.7% (15)</td>
</tr>
<tr>
<td>Web usability</td>
<td>50.0% (13)</td>
</tr>
</tbody>
</table>

2.4.7 Quality processes for web accessibility

Participants were asked about the quality processes they have in place to establish and maintain web accessibility and usability.

In relation to web accessibility, participants were asked whether they (or the organizations to whom they outsource the web development) assess the website or application for accessibility for disabled and older people. 61.5% (16) participants said that they did. Most interestingly, participants were often very unclear how accessibility had been assessed. Only 4 participants (25.0% of the 16 participants who have accessibility assessment) could name the type of assessment, for example, WCAG1, WCAG2 conformance testing or user testing. Only 7.7% (2) participants mentioned that they had a specific WCAG1 or WCAG2 target, for example that the website should be WCAG2 AA conformant.

When asked how the findings from such assessments were carried through to the website, 42.3% (11) of participants said that it was their responsibility, or the responsibility of their team. But few participants discussed a formal process for ensuring accessibility, only 15.4% (4) referred to using the checklist from the accessibility audit. Several other participants were concerned that they had no process for ensuring accessibility was carried through to the website.

42.3% (11) of participants also had a process for maintaining accessibility, although for the majority this was through manual checking. 2 of these participants rely on the CMS to ensure accessibility.

2.4.8 Quality processes for web usability

In relation to web usability, participants were asked whether they (or the organizations to whom they outsource the web development) assess the website or application for usability. 61.5% (16) participants said that they did, exactly the same percentage as those who assess web accessibility. Interestingly, this was not the same 16 participants who assessed for web accessibility. So this is not a question of general web quality processes.
When asked how the findings from such assessments were carried through to the website, 50.0% (13) of participants said that it was their responsibility, or the responsibility of their team. As with web accessibility, few participants discussed a formal process, in this case 19.2% (5) mentioned using a checklist.

2.5 Discussion and Conclusions

The chapter presented the study of in-depth interviews with 26 website commissioners, drawn from across the private, public and third sectors, and responsible for a range of websites and applications from the modest to the very large.

The websites for which the participants are responsible include a range of complex features that are challenges for accessibility. The most common are media players (found in 61.5% of the websites) and link sharing for social network sites (found in 57.7% of the websites). The use of CMSs was very common, with 88.5% of the websites using them. There were a number of comments from participants indicating that they rely on the CMS to ensure accessibility of the website. This strategy is only as good as the CMS used and its ability to ensure accessibility correctly. Web commissioners need to be able to evaluate accessibility independently of their CMS, and in the case of several participants, this was clearly not happening.

Participants’ organizations often out-source part or all of the website work (34.6% total outsourcing, 46.2% partial outsourcing). Out-sourcing of web design and development work lead to a number of issues around accessibility. Several participants remarked that companies claimed to have expertise in accessibility, but they found this hard to judge independently. In addition, out-sourcing could create accessibility-related conflicts, as one participant found to their cost, when their design company proposed a design for the website that the implementation company argued would be inaccessible (unfortunately, design won over accessibility).

In terms of disabled and older people as audiences for websites, only 11.5% of participants spontaneously mentioned these groups as potential audiences for their website. When prompted, 69.2% agreed that these groups might be potential audiences for the website. These results show that website commissioners are not really thinking about the potential audiences of their websites, as disabled and older people are almost always possible audiences for websites. This means that accessibility will often be omitted from the agenda in the commissioning of websites.

The results on audience contrast strangely with the results on policies on web accessibility. When asked whether their organization has a policy on accessibility of the website for disabled and older people, 61.5% said they did. This is a total contrast to only 11.5% saying that disabled and older people are part of the potential audience for the website. This contrast probably shows the subject bias effect one gets in asking questions about web accessibility. It may also show the lack of realism in the minds of most people about people with disabilities and older people – they fail to realize the range of things that disabled and older people may wish to do. An example was a school for children with disabilities who did not think their website needed to be accessible to disabled or older people – although parents of the pupils were a key target audience. Could not the parent of a disabled child be disabled themselves or could not the guardian of a disabled child be their grandparent?
Interestingly, the results on policies on web accessibility are very similar to those found by the Disability Rights Commission (2004) survey (55% of website commissioners said their organization had a policy on accessibility) and the BenToWeb Project survey (45.6% of website commissioners said their organization had a policy on web accessibility).

Most website commissioners (88.5%) correctly understood the meaning of web accessibility with somewhat less showing a correct understanding of web usability (69.2%). About half the participants had some training or knowledge of web technologies, web accessibility and web usability. However, only a third had any training in web accessibility and usability and had picked up information on the job. In retrospective, it would have been better to ask separate questions about training and knowledge, to tease out this difference. Even the figure of one third of web commissioners having knowledge of web accessibility may over estimate their knowledge, as a number of the participants had worked with the CFIT, part of NCBI and had training or knowledge in web accessibility from this collaboration.

Finally, participants were asked about their web quality processes for accessibility and usability. For web accessibility, more than half (61.5%) of participants said that they (or the companies they out-source to) assess the accessibility of their website. However, it was worrying that only a quarter of these participants could say what kind of testing had been done. This does not indicate a good understanding of the process. And only 7.7% of participants named a WCAG level conformance that they were aiming for/were tested against. Few participants had a formal process for ensuring that accessibility problems identified in testing were addressed and that accessibility of the website was maintained going forward from the testing. This would indicate a clear gap in the support provided to website commissioners in tracking accessibility of their site. As one participant said:

“we’d like to get in a system that automatically kicks off once a week and draws attention/highlights issues that need fixing. We know it would not catch everything, but it would catch a lot …”

Results for web usability quality control were very similar and participants appeared to deal with accessibility and usability together. This may be a “hook” that those concerned with web accessibility can use to convince website commissioners to implement quality control for accessibility – by bundling it with usability and showing that commissioners can achieve two goals with one system or tool.
3  Requirements for web developers

3.1  Introduction

The second group of stakeholders in the chain to produce accessible websites and applications is web developers. In comparison to website commissioners, developers need to know not only the principles of web accessibility, but also the “nitty gritty” of how to produce accessible code. This includes not only knowledge of the WCAG, but also an understanding of how users access websites and applications and how their assistive technologies interact with web code.

The chapter presents a number of methods used to investigate the working practices of web developers along with initial results of the analysis of the data collected. The particular aim of this study was to investigate web developers’ understanding of web accessibility issues as well as their requirements for tools and information to support them in developing accessible websites and applications. This aspect of the investigation began with an online survey completed by 60 web developers that provided an initial understanding of the technologies, tools and techniques used. This was followed up by a series of in-depth contextual inquiries with 13 web developers, which allowed the complex work practices of this cohort to be explored.

3.2  Previous work

A surprisingly small number of studies have investigated the needs of web developers in understanding and implementing accessible web content and applications. Many studies have shown that most aspects of the Web are not accessible and that the situation is not improving (Kane, Shulman, Shockley, and Ladner, 2007; Lazar et al, 2011). Therefore we need information from both web developers and website “responsibles” (people who commission, own or are in any way responsible for a website, without being the developer) as to what the barriers to accessible web development are. This chapter presents a study of web developers, to explore their understanding of web accessibility issues and the problems they have in addressing web accessibility. The following chapter presents a study of website responsibles, to explore their understanding of the same issues.

Lazar, Dudley-Sponaugle and Greenidge (2004) conducted one of the first studies to investigate web developers understanding of accessibility issues and their perceptions of barriers to accessible web design. They conducted an online survey of 175 webmasters. Results were surprisingly positive with 65.7% of respondents indicating they had developed an accessible website, 73.7% indicating they are familiar with the appropriate legal requirements for accessible websites and 56.0% indicating that their current website is accessibility to people with visual impairments. Lazar and colleagues note that there are a number of paradoxes in the results – 78.9% of respondents said they were familiar with automated software tools to help with accessibility, but only 56.0% indicated that their current website was accessible. However, they fail to comment on the far greater paradox of these results – that if web developers were so aware of the issues and creating accessible websites so much of the time, the web would not be as inaccessible as it still is. It could be that the self-selecting sample of web developers is not at all representative of web
developers in general or it could be that particularly in the anonymous environment of an online questionnaire, it is too easy for web developers to give the responses that they feel are appropriate (an example of subject bias, see Rosnow and Rosenthal, 1997).

Rosson et al (2005) also conducted of 334 web developers. They only asked one question related to accessibility, whether they tested their site for whether “users who might have disabilities will be able to use your site”. This was answered on a 1 to 5 scale, where 1 meant hardly ever and 5 meant quite often. The mean response was 2.75 (standard deviation = 1.26, 276 responses). This was a lower mean rating than for evaluating the usability of the site (mean = 4.33, standard deviation = 0.93, 276 responses) or for testing browser compatibility (mean = 3.75, standard deviation = 1.26, 276 responses). Although this is only one question, it is interesting as this sample of web developers would not have self-selected for an interest or expertise in web accessibility.

As mentioned in Chapter 2 (section 2.2) the Disability Rights Commission (2004) undertook a survey of web developers. 388 web development agencies were approached by letter to complete a short questionnaire, but only a 6% response rate ensued. The response rate was particularly disappointing, as there had been considerable publicity and interest amongst website developers about the research by the Disability Rights Commission, and one would have hoped that the issue was considered important to this group. The questionnaire was therefore followed up with face-to-face interviews with 25 website developers. Findings from the questionnaires with website developers included:

- 80% reported that they attempted to develop accessible sites at least some of the time;
- Clients were generally neither interested in nor knowledgeable about accessibility issues;
- Accessibility expertise amongst web developers was low: only 9% claimed any sort of expertise;
- Only 9% of web developers had ever included disabled people in user testing of a site;
- 21% used automated tools and 65% used guidelines (WCAG, RNIB and UK government guidelines) on accessibility.

Responses to the interviews with the web developers painted a very similar picture.

The BenToWeb Project (2006) conducted a European-wide online survey of web developers with a follow-up sample of respondents being interviewed by phone to gather more in-depth information. 249 people responded to the online survey, from 8 different European countries. Some of the key results were:

- A very wide variety of tools was used to develop websites, over 50 in total
- 39.8% developed multimedia environments such as Flash
- 45.4% used JavaScript at least occasionally
- Many developers used CMSs, with over 60 being mentioned
- 68.3% used CSS in development
- 45.4% reported that their organizations were “very interested” in web accessibility
• 33.2% of web developers said their organization employed one or more accessibility experts, significantly more likely in large organizations

• 42.2% of web developers reported they had training in web design, significantly more likely in large organizations, 59% of these reported that their training included web accessibility

• 19.7% reported that they were “very knowledgeable” about web accessibility

• 73.9% reported that they had tried to develop an accessible website at some point

• 2.8% thought it was “very easy” to develop an accessible site

• When asked what was difficult about accessible web design, the most common answers were browser incompatibility (mentioned 19 times), CSS issues (13 mentions) and avoiding tables for layout (9 mentions)

• 67.4% reported that they have conducted accessibility testing; this included automated testing (30.1%), expert testing (18.1%) and user testing (18.1%)

• 39.8% reported that they used WCAG for accessibility testing

• 63.1% reported that WCAG was “very useful” in accessibility testing

• 16 different web accessibility testing resources were listed, the most popular being the Einfach-fur-alle.de website (19 mentions), “Barrierefreies Web design“ by Jan Hellbuschs (16 mentions), www.w3.org website (14 mentions)

• Web developers were asked what they felt the greatest barriers to accessible web design were, the most common answers were lack of interest/commitment /knowledge amongst web commissioners (27 mentions), the additional time/cost for development of an accessible site (18 mentions), and problems with WCAG: outdated, inflexible, unclear, open to interpretation, not concrete enough, too difficult/time consuming to implement (17 mentions)

van Isacker et al (2010, see also Korn et al, 2009) conducted a survey of 254 web developers, this included 25 students, so not all were professionals. They found that 90% of their sample was familiar with the concept of accessibility. Of the professionals, 87% were familiar with WCAG1, but only 45% were familiar with WCAG2. 63% had participated in accessibility related projects, and these individuals were asked what tools they used to evaluate the accessibility of their websites. The results are rather strange: the most commonly used tool type is CSS validation tools (used by 72%), followed by simulation tools such as IBM aDesigner (53%). So-called “HTML assessment validators” such as WAVE\(^{16}\) and Hera\(^{17}\) are used by 45% of respondents. WAVE is actually a web accessibility testing tool (WATT) and Hera is an automatic accessibility testing tool (AATT), neither is primarily an HTML validator. It would have been interesting to distinguish between these two types of tools in the questions put to these developers. Unfortunately, the authors jump to the conclusion that because their respondents use particular tools, these are their preferred tools; but they failed to ask their opinions of the tools, or difficulties with the tools, simply concluding that those used by more than 50% of respondents are the most preferred tools.

\(^{16}\) http://wave.webaim.org/  
\(^{17}\) http://www.sidar.org/hera/index.php.en
3.3 Survey of Web Developer Practices

3.3.1 Method

To gain an initial understanding of the technologies, tools and techniques used by web developers, a survey was developed. Development of the survey was an iterative process with contributions from the personnel at York as well as the wider i2Web consortium. Several versions of the schedule were produced over a series of discussions. The survey was also piloted on a web developer who is a friend of one of the York personnel, with the questions refined in light of the feedback received.

To ensure the survey reached a potentially global audience, it was administered online using the survey software, QuestionPro. A link to the survey was distributed via web developer mailing lists, online forums, the i2Web project website and personal contacts in the industry. Respondents were encouraged to pass the survey onto colleagues and an incentive of a 10€ gift voucher was offered for the first 50 respondents to complete the survey.

3.3.1.1 Survey topics

The survey comprised 25 questions and was estimated to take approximately 10 minutes to complete. To prevent the survey from becoming too much of a burden on the time of busy professionals, the majority of questions were closed-ended. Respondents were, however, given the opportunity to elaborate upon their responses with a number of optional open-ended questions. Respondents were informed that their responses were confidential and that data would only be reported in the aggregate.

The survey covered the following topics:

- **Roles:** their job title, their organisation, and their experience in the field of web development. To establish their educational background, the respondent was asked about the educational qualifications they had completed and the subject they had studied.

- **Websites:** the nature and size of the websites that the web developer is involved in, as well as the platforms they develop for and the website features they include. Although it was not asked explicitly, these questions helped to determine whether the respondent was a developer of Web 2.0 and Rich Internet applications.

- **Development tools:** the development tools used by the web developer, such as Integrated Development Environments (IDEs) and/or text editors. A near-exhaustive list of tools was provided, although the respondent was invited to specify anything that had not been covered.

- **Features of development tools:** the typical features of IDEs and text editors that are used (or unused) by the web developer. IDEs and text editors include a range of features to aid the web development process. As well as establishing whether or not the respondent uses a particular feature, these questions also determined the environment it is used in, be it an IDE, a text editor or another application altogether. Again, the respondent was invited to specify anything that had not been covered in the near-exhaustive list of features.
• **Web development technologies:** *the technologies that are used (or unused) by the web developer*. There are numerous technologies that can be used to develop websites. As well as establishing whether or not the respondent uses a particular technology, these questions also determined the environment it is used in, be it an IDE, a text editor or another application altogether. Again, the respondent was invited to specify anything that had not been covered in the near-exhaustive list of technologies.

• **Standards-compliance:** *whether the websites that the web developer is involved in conform to published standards, such as HTML, CSS etc.* These questions established if and how the respondent checks whether a website is standards-compliant.

• **Accessibility:** *whether the websites that the web developer is involved in are usable by people of all abilities and disabilities.* These questions established if and how the respondent checks whether a website is accessible.

The full survey can be found in Appendix B: Survey of Web Developer Practices

### 3.3.1.2 Respondents

60 respondents completed the survey. 78% (47) were male, 17% (10) were female, 5% (3) did not respond. The age groups that respondents fell into were quite varied (see Table 3.1, below).

<table>
<thead>
<tr>
<th>Age Group</th>
<th>Percentage (Number)</th>
</tr>
</thead>
<tbody>
<tr>
<td>18-20</td>
<td>0 (0)</td>
</tr>
<tr>
<td>21-30</td>
<td>38 (23)</td>
</tr>
<tr>
<td>31-40</td>
<td>38 (23)</td>
</tr>
<tr>
<td>41-50</td>
<td>13 (8)</td>
</tr>
<tr>
<td>51-60</td>
<td>5 (3)</td>
</tr>
<tr>
<td>61-70</td>
<td>2 (1)</td>
</tr>
<tr>
<td>Did not respond</td>
<td>3 (2)</td>
</tr>
<tr>
<td><strong>TOTAL</strong></td>
<td><strong>100 (60)</strong></td>
</tr>
</tbody>
</table>

The survey drew respondents from a wide range of countries, including Mexico, South Africa and the USA, however the vast majority were from the United Kingdom and Ireland. 87% (52) of respondents had a Bachelor’s degree or higher qualification. Although respondents were qualified in a range of subjects, the most common subject was Computer Science.
3.3.2 Results

3.3.2.1 Roles

Respondents’ job titles were surprisingly varied and ranged from “Web Developer” to “Interaction Architect”. The most common response, however, was “Software Engineer”.

The majority of respondents either worked for a small company or organisation (43% / 26) or a large company or organisation (37% / 22). Only 10% (6) were independent contractors or freelancers. Another 10% worked for no financial gain, split between 8% (5) who worked for a charity, educational institution, library, museum or non-profit organization, and 2% (1) who were volunteers, students or hobbyists.

Respondents were relatively experienced with 25% (15) having worked in the field of web development for more than 10 years. The remaining participants were spread fairly equally between the other categories (see Table 3.2, below).

Table 3.2: Experience distribution of respondents

<table>
<thead>
<tr>
<th>Experience</th>
<th>Percentage (Number)</th>
</tr>
</thead>
<tbody>
<tr>
<td>More than 10 years</td>
<td>25 (15)</td>
</tr>
<tr>
<td>9-10 years</td>
<td>15 (9)</td>
</tr>
<tr>
<td>6-8 years</td>
<td>18 (11)</td>
</tr>
<tr>
<td>3-5 years</td>
<td>22 (13)</td>
</tr>
<tr>
<td>1-2 years</td>
<td>17 (10)</td>
</tr>
<tr>
<td>Less than 1 year</td>
<td>3 (2)</td>
</tr>
<tr>
<td>Did not respond</td>
<td>0 (0)</td>
</tr>
<tr>
<td><strong>TOTAL</strong></td>
<td><strong>100 (60)</strong></td>
</tr>
</tbody>
</table>

3.3.2.2 Websites

Respondents were involved in various types of websites and applications:

- The majority (68% / 41) were involved in creating informational websites. These are largely text-based, with the occasional picture and possibly a feedback form.
- 37% (22) were involved in creating E-commerce websites. These allow users to browse and/or buy products online.
- 47% (28) were involved in community websites. These allow users to interact with other users, via discussion boards, forums, chat, private messaging etc.
- 32% (19) were involved in productivity websites. These provide interactive end-user services, such as currency conversion, driving directions, search engines etc.
- Half of the respondents (50% / 30) were involved in internal websites / intranets. These are private websites used internally by employees of a company or organisation.
- 45% (27) were involved in Rich Internet Applications / Web 2.0 Applications. These are complex web applications that have many of the characteristics of desktop application software.

Other types of websites/applications that respondents were involved in include mobile apps, online portfolio/CV websites, educational websites, and healthcare portals.

The size of websites that respondents were involved in was varied. 40% (24) of respondents were involved in developing small websites (6 - 50 pages). 27% (16) were involved in developing medium websites (51 - 100 pages). 23% (14) were involved in developing large websites (100 - 1000 pages). 10% (6) are involved in developing very large (over 1000 pages).

It is unlikely (but not impossible) that a web developer would be responsible for maintaining hundreds or thousands of individual webpages. More likely is the use of a Content Management System (CMS) whereby multiple webpages can be generated from a small number of templates.

The vast majority of respondents (95% / 57) developed websites for the desktop or laptop. The growing use of mobile Internet devices is reflected in the 53% (32) of respondents who developed for smartphones, and the 32% (19) who developed for iPad and tablets. Only 3% (2) developed for iTV/IPTV and 2% (1) developed for self-service machines. The choice of platform was irrelevant to 4% (2) of respondents, who claimed to develop platform- and device-independent websites and applications.

Respondents include a range of features in the websites they are involved in:

- The most common features are, understandably, text and images, which are both incorporated by 85% (51) of respondents. Tables are slightly less common but still incorporated by 67% (40) of respondents. (It was not specified whether these were data or layout tables although the former is presumed).

- Almost two thirds of respondents incorporate some degree of interactivity through feedback forms (65% / 39), blogs (65% / 39) and user feedback (e.g. comments, ratings, reviews etc. (65% / 39). Surprisingly, polls and quizzes are less common (55% / 33) and community features, such as discussion boards and forms, even less so (47% / 28). Only 28% (17) incorporate wikis into the websites they are involved in.

- Connections with other websites and services are fairly common, with 62% (37) of respondents incorporating social networking (e.g. Facebook, twitter etc.) into the websites they are involved in. 58% (35) of respondents incorporate web feeds (e.g. RSS, Atom) allowing content to be shared elsewhere and 43% (26) create “mashups” with web services, such as Google Maps or Wikipedia.

- Multimedia, such as videos, animations and podcasts, is only incorporated by 43% (26) of respondents and only 32% (19) of respondents incorporate image maps.

- Though it is often noted that all applications eventually evolve to include email capabilities, only 32% (19) of respondents incorporate messaging facilities, such as private messaging, chat and email, into the websites they are involved in.

Other features that respondents mentioned included: data visualisation (interactive graphs, drag-and-drop functionality for interacting with data), interactive help systems, QR codes, self-service, e-commerce, and APIs.
### 3.3.2.3 Development tools

An Integrated Development Environment (IDE) is a software application that provides a set of features to aid the development of websites and applications. These features typically include, but are not limited to, a source code editor, a compiler/interpreter, build automation tools and a debugger. IDEs are popular with many web developers: only 18% (11) of respondents do not use an IDE at all. In order of most frequently mentioned, the IDEs used by survey respondents include:

1. Eclipse (50% / 30 of respondents)
2. Adobe Dreamweaver (33% / 20)
3. Microsoft Visual Studio (22% / 13)
4. Adobe Flash Builder (18% / 11)
5. Netbeans (15% / 9)
6. Adobe Flex (13% / 8)
7. HTML-Kit (10% / 6)
8. ActiveState Komodo (8% / 5)
9. Aptana Studio (8% / 5)
10. Code (8% / 5)
11. IBM Rational Application Developer (8% / 5)
12. Microsoft Expression Studio (8% / 5)
13. JetBrains IntelliJ (7% / 4)
14. JetBrains WebStorm (5% / 3)
15. TopStyle (2% / 1)
16. WinDev (2% / 1)

Other IDEs mentioned by respondents include: ColdFusion Builder and Wing IDE.

IDEs not used by any of the respondents include: Code Crusader, Evrsoft FirstPage, FlashDevelop, Open Dialect, Quanta Plus, RapidWeaver, and uCMS.

A text editor is an application used for editing plain text files. Though not as sophisticated as an IDE, many text editors offer additional features to facilitate the production of source code. These features typically include, but are not limited to, syntax highlighting, keyboard shortcuts, tooltip prompts and auto-completion. Text editors are also very popular with many web developers: only 8% (5) of respondents do not use a text editor at all. In order of most frequently mentioned, the text editors used by survey respondents include:

1. Notepad++ (43% / 26 of respondents)
2. Vi/Vim (23% / 14)
3. TextMate (13% / 8)
4. Bluefish (12% / 7)
5. TextPad (10% / 6)
6. BBEdit (8% / 5)
7. Emacs (8% / 5)
8. PSPad (8% / 5)
9. UltraEdit (8% / 5)
10. CoffeeCup (7% / 4)
11. BlueGriffon (5% / 3)
12. jEdit (5% / 3)
13. Crimson Editor (3% / 2)
14. Freeway (3% / 2)
15. Geany (3% / 2)
16. HTMLPad (3% / 2)
17. EmEditor (2% / 1)
18. OpenBEXI (2% / 1)
19. SciTE (2% / 1)
20. SlickEdit (2% / 1)

Other text editors mentioned by respondents include: Coda, gedit, IDLE, Kate, Komodo Edit, Microsoft Notepad, Notepad 2, SilverEdit, Smultron, Textweaver, and TextWrangler.

IDEs not used by any of the respondents include: iWeb, KompoZer, and Vrode.

3.3.2.4 Features of development tools

Both IDEs and text editors include a range of features to aid the web development process. Each type of application has its own strengths and weaknesses. To make the most of a particular feature, a web developer may prefer to use an IDE, or a text editor, or another type of application altogether.

For the features listed in Table 3.3, below, the largest proportion of respondents prefer to use an IDE.

<table>
<thead>
<tr>
<th>Feature</th>
<th>IDE</th>
<th>Text editor</th>
<th>Other application</th>
<th>Do not use</th>
</tr>
</thead>
<tbody>
<tr>
<td>Code completion</td>
<td>67% (40)</td>
<td>37% (22)</td>
<td>5% (3)</td>
<td>8% (5)</td>
</tr>
<tr>
<td>Code debugging</td>
<td>55% (33)</td>
<td>17% (10)</td>
<td>15% (9)</td>
<td>7% (4)</td>
</tr>
<tr>
<td>Code generation</td>
<td>47% (28)</td>
<td>17% (10)</td>
<td>8% (5)</td>
<td>22% (13)</td>
</tr>
<tr>
<td>Code navigation</td>
<td>55% (33)</td>
<td>25% (15)</td>
<td>2% (1)</td>
<td>13% (8)</td>
</tr>
<tr>
<td>Compiler / interpreter</td>
<td>43% (26)</td>
<td>5% (3)</td>
<td>8% (5)</td>
<td>23% (14)</td>
</tr>
<tr>
<td>CSS validation</td>
<td>37% (22)</td>
<td>23% (14)</td>
<td>30% (18)</td>
<td>15% (9)</td>
</tr>
<tr>
<td>Error reporting</td>
<td>47% (28)</td>
<td>10% (6)</td>
<td>10% (6)</td>
<td>17% (10)</td>
</tr>
<tr>
<td>File management</td>
<td>42% (25)</td>
<td>20% (12)</td>
<td>28% (17)</td>
<td>10% (6)</td>
</tr>
<tr>
<td>HTML / XML validation</td>
<td>40% (24)</td>
<td>25% (15)</td>
<td>30% (18)</td>
<td>12% (7)</td>
</tr>
<tr>
<td>Source version control</td>
<td>38% (23)</td>
<td>8% (5)</td>
<td>37% (22)</td>
<td>10% (6)</td>
</tr>
<tr>
<td>Unit testing</td>
<td>40% (24)</td>
<td>7% (4)</td>
<td>22% (13)</td>
<td>18% (11)</td>
</tr>
</tbody>
</table>

For none of the features did the largest proportion of respondents prefer to use a text editor. For the features listed in Table 3.4, below, the largest proportion of respondents prefer to use some other application.
Table 3.4: Features for which the largest proportion of respondents prefer to use some other application

<table>
<thead>
<tr>
<th>Feature</th>
<th>IDE</th>
<th>Text editor</th>
<th>Other application</th>
<th>Do not use</th>
</tr>
</thead>
<tbody>
<tr>
<td>File transfers</td>
<td>17% (10)</td>
<td>12% (7)</td>
<td>50% (30)</td>
<td>12% (7)</td>
</tr>
<tr>
<td>Web page preview</td>
<td>25% (15)</td>
<td>12% (7)</td>
<td>50% (30)</td>
<td>7% (4)</td>
</tr>
</tbody>
</table>

Though respondents were not asked to specify the other applications they prefer to use, a dedicated FTP application is most likely to be used for file transfers whereas web page previews are most likely to be carried out in a web browser.

The largest proportion of respondents do not use the features listed in Table 3.5, below.

Table 3.5: Features for which the largest proportion of respondents do not use

<table>
<thead>
<tr>
<th>Feature</th>
<th>IDE</th>
<th>Text editor</th>
<th>Other application</th>
<th>Do not use</th>
</tr>
</thead>
<tbody>
<tr>
<td>Automated accessibility tests</td>
<td>25% (15)</td>
<td>8% (5)</td>
<td>10% (6)</td>
<td>38% (23)</td>
</tr>
<tr>
<td>Model creation and editing</td>
<td>27% (16)</td>
<td>7% (4)</td>
<td>18% (11)</td>
<td>28% (17)</td>
</tr>
</tbody>
</table>

3.3.2.5 Web development technologies

There is a range of web development technologies that can be used to develop websites. When working with a particular web development technology, a web developer may prefer to use an IDE, or a text editor, or another type of application altogether.

For the web development technologies listed in Table 3.6, below, the largest proportion of respondents prefer to use an IDE.

Table 3.6: Technologies for which the largest proportion of respondents prefer to use an IDE

<table>
<thead>
<tr>
<th>Feature</th>
<th>IDE</th>
<th>Text editor</th>
<th>Other application</th>
<th>Do not use</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ajax</td>
<td>45% (27)</td>
<td>33% (20)</td>
<td>10% (6)</td>
<td>12% (7)</td>
</tr>
<tr>
<td>CSS</td>
<td>50% (30)</td>
<td>48% (29)</td>
<td>10% (6)</td>
<td>8% (5)</td>
</tr>
<tr>
<td>HTML / XHTML</td>
<td>58% (35)</td>
<td>55% (33)</td>
<td>17% (10)</td>
<td>2% (1)</td>
</tr>
<tr>
<td>Java</td>
<td>43% (26)</td>
<td>13% (8)</td>
<td>8% (5)</td>
<td>23% (14)</td>
</tr>
<tr>
<td>JavaScript</td>
<td>52% (31)</td>
<td>47% (28)</td>
<td>12% (7)</td>
<td>5% (3)</td>
</tr>
</tbody>
</table>

For the web development technologies listed in Table 3.7, below, the largest proportion of respondents prefer to use a text editor.

Table 3.7: Technologies for which the largest proportion of respondents prefer to use a text editor

<table>
<thead>
<tr>
<th>Feature</th>
<th>IDE</th>
<th>Text editor</th>
<th>Other application</th>
<th>Do not use</th>
</tr>
</thead>
<tbody>
<tr>
<td>PHP</td>
<td>37% (22)</td>
<td>41% (25)</td>
<td>7% (4)</td>
<td>18% (11)</td>
</tr>
</tbody>
</table>
For the web development technologies listed in Table 3.8, below, the largest proportion of respondents prefer to use some other application.

**Table 3.8: Technologies for which the largest proportion of respondents prefer to use some other application**

<table>
<thead>
<tr>
<th>Feature</th>
<th>IDE</th>
<th>Text editor</th>
<th>Other application</th>
<th>Do not use</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apache</td>
<td>17%</td>
<td>27%</td>
<td><strong>28%</strong></td>
<td>28%</td>
</tr>
<tr>
<td>MySQL</td>
<td>28%</td>
<td>18%</td>
<td><strong>43%</strong></td>
<td>8%</td>
</tr>
</tbody>
</table>

Though respondents were not asked to specify the other applications they prefer to use, both Apache and MySQL are likely to be managed using a dedicated software or web application.

The majority of respondents do not use the features listed in Table 3.9, below.

**Table 3.9: Technologies for which the largest proportion of respondents do not use**

<table>
<thead>
<tr>
<th>Feature</th>
<th>IDE</th>
<th>Text editor</th>
<th>Other application</th>
<th>Do not use</th>
</tr>
</thead>
<tbody>
<tr>
<td>ActiveX components</td>
<td>15%</td>
<td>7%</td>
<td>10%</td>
<td>43%</td>
</tr>
<tr>
<td>Adobe Flash</td>
<td>20%</td>
<td>8%</td>
<td><strong>17%</strong></td>
<td>45%</td>
</tr>
<tr>
<td>ASP</td>
<td>12%</td>
<td>8%</td>
<td>2%</td>
<td>45%</td>
</tr>
<tr>
<td>CGI</td>
<td>7%</td>
<td>17%</td>
<td>3%</td>
<td>40%</td>
</tr>
<tr>
<td>ColdFusion</td>
<td>10%</td>
<td>5%</td>
<td>2%</td>
<td><strong>52%</strong></td>
</tr>
<tr>
<td>CSP</td>
<td>5%</td>
<td>2%</td>
<td>2%</td>
<td><strong>53%</strong></td>
</tr>
<tr>
<td>DB2</td>
<td>5%</td>
<td>5%</td>
<td>7%</td>
<td><strong>43%</strong></td>
</tr>
<tr>
<td>Firebird (RDBMS)</td>
<td>3%</td>
<td>5%</td>
<td>5%</td>
<td><strong>50%</strong></td>
</tr>
<tr>
<td>Google Web Toolkit</td>
<td>13%</td>
<td>7%</td>
<td>12%</td>
<td><strong>43%</strong></td>
</tr>
<tr>
<td>Groovy</td>
<td>8%</td>
<td>3%</td>
<td>2%</td>
<td><strong>50%</strong></td>
</tr>
<tr>
<td>Lotus Domino</td>
<td>3%</td>
<td>3%</td>
<td>0%</td>
<td><strong>53%</strong></td>
</tr>
<tr>
<td>Microsoft Silverlight</td>
<td>10%</td>
<td>3%</td>
<td>5%</td>
<td><strong>48%</strong></td>
</tr>
<tr>
<td>Microsoft SQL Server</td>
<td>15%</td>
<td>5%</td>
<td>20%</td>
<td><strong>33%</strong></td>
</tr>
<tr>
<td>.NET</td>
<td>20%</td>
<td>7%</td>
<td>5%</td>
<td><strong>43%</strong></td>
</tr>
<tr>
<td>Oracle</td>
<td>13%</td>
<td>5%</td>
<td>3%</td>
<td><strong>50%</strong></td>
</tr>
<tr>
<td>PostgreSQL</td>
<td>8%</td>
<td>5%</td>
<td>17%</td>
<td><strong>40%</strong></td>
</tr>
<tr>
<td>SQLite</td>
<td>3%</td>
<td>3%</td>
<td>13%</td>
<td><strong>43%</strong></td>
</tr>
<tr>
<td>Sybase</td>
<td>3%</td>
<td>0%</td>
<td>2%</td>
<td><strong>55%</strong></td>
</tr>
<tr>
<td>Perl</td>
<td>5%</td>
<td>17%</td>
<td>5%</td>
<td><strong>38%</strong></td>
</tr>
<tr>
<td>Pyjamas</td>
<td>2%</td>
<td>2%</td>
<td>2%</td>
<td><strong>57%</strong></td>
</tr>
</tbody>
</table>
### 3.3.2.6 Standards-compliance

82% (49) of respondents check whether the websites they are involved in are standards-compliant (i.e. they conform to published standards, such as HTML, CSS etc.). Respondents use a range of methods to check whether the websites they are involved in are standards-compliant:

- 73% (44) use the W3C HTML validation service website
- 65% (39) use the W3C CSS validation service website
- 42% (25) use browser toolbars and extensions (e.g. Web Developer Toolbar)
- 40% (24) use validation applications (e.g. HTML Tidy, Total Validator etc.)
- 37% (22) use their own judgement
- 28% (17) use in-application support (e.g. through Adobe Dreamweaver)
- 25% (15) use unit testing (e.g. Selenium)
- 8% (5) manually compare websites against the standards documents

### 3.3.2.7 Accessibility

70% (42) of respondents check whether the websites they are involved in are accessible (i.e. they are usable by people of all abilities and disabilities). Respondents use a range of methods to check whether the websites they are involved in are accessible.

- 57% (34) conduct manual accessibility testing
- 28% (17) conduct automated accessibility testing (e.g. CynthiaSays, imergo etc.)
- 28% (17) conduct testing with assistive technologies (e.g. using a screen reader)
- 27% (16) conduct expert testing (e.g. by an accessibility expert)
- 25% (15) conduct user testing with people with disabilities
- 15% (9) conduct user testing with older adults

### 3.4 Contextual Inquiries with Web Developers

#### 3.4.1 Method

To further understand the complex work practices of web developers, a series of contextual inquiries were conducted. The contextual inquiry methodology was developed by Beyer and Holtzblatt (1997) and incorporates a one-on-one observation of work practice in its naturally
occurring context. Detailed information can be collected about work practices by observing and interviewing a participant whilst he or she actually works. At any point before, during, or after an observation, the interviewer can discuss the participant’s daily routines and work processes in order to develop a deep understanding of them. Ultimately, the purpose of contextual inquiry is to understand how and why something is done or why something is not done and how it may be improved.

Whilst contextual inquiry was considered a suitable methodology to investigate the work practices of web developers, some adjustments were necessary. Web development requires a great deal of focus and concentration over long periods of time. In addition to this, despite often being a member of a team, web developers typically work alone. As a result, any observation of work practice – however discreet – may be disruptive to an engaged web developer. Furthermore, given the nature of web development, which is largely carried out via a user interface, it may be difficult for the interviewer to observe the minutiae of the work tasks being undertaken.

Therefore, a more directed approach was carried out, during which the interviewer prompted the participant to carry out certain activities. These activities were then observed and discussed according to the methodology of contextual inquiry. Though slightly more artificial, this adaptation of contextual inquiry ensured minimal disruption to the participants and allowed the interviewer to focus on the aspects of the participants’ work that were relevant to the investigation.

### 3.4.1.1 Contextual Inquiry schedule

A contextual inquiry schedule of questions and tasks for the web developer to undertake was developed in an iterative process, with contributions from personnel at York. Several versions of the schedule were produced over a series of discussions. The interview and task schedule was also piloted on a web developer who is a friend of one of the York personnel, with the schedule refined in light of that experience.

The schedule covered the following topics:

- **The web developer:** their knowledge and skills, their organisation, and the nature of the work they are involved in. As well as asking for objective, factual information to put the participant at ease, this section also established whether the participant felt their knowledge of the field was up-to-date and what they considered to be their strengths and weaknesses.

- **Workspace configuration:** an “introduction” to the participant’s workspace and immediate surroundings. This section investigated the hardware used by the participant (e.g. monitors, input devices etc.) as well as any development-related artefacts (e.g. post-it note reminders, whiteboards, notepads etc.). It also established the software applications and webpages that the participant typically uses to develop websites.

- **Communication:** who the web developer communicates with during a typical day, and how. This section focused on how and when the participant communicates with clients, colleagues, and management in order to gain an understanding of their day-to-day interaction with others. It also established how web development requirements and progress reports are formalized in the participant’s organization.

- **Help and information:** who and what resources the web developer turns to when they need help. Having asked the participant to assume they have run into a typical
technical problem, this section investigated who and what resources they turn to for help, the type of information they are typically looking for, and the stage of development that such problems usually occur.

- **Standards-compliance**: *what standards-compliance means to the web developer and how they achieve it*. This section explored the participant’s understanding of a standards-compliant website, any challenges they have faced in making websites standards-compliant, and the extent to which their websites are standards-compliant.

- **Preview**: *how the web developer previews a website they have created*. This section required the participant to demonstrate the browsers, applications, and devices that they typically use to preview websites. It also established what the developer looks for when previewing a website and how dynamic webpages and third-party components are previewed.

- **Validation**: *how the web developer validates a website they have created*. This section required the participant to demonstrate how they validate a website. It also established how they tackle any validation errors, how frequently they validate a website, how useful they find the feedback from validation tools, and how easily validation checks fit into the participant’s workflow.

- **Users**: *what consideration the web developer gives to the users of their websites*. This section focused upon the usability of the developer’s websites, whether instructions are required, and established who is responsible for the development of feedback, instructions, and error messages.

- **Accessibility**: *what consideration the web developer gives to the accessibility of their websites*. This section explored the web developer’s understanding of accessibility, the importance they place upon accessibility, and whom they feel is responsible for making websites accessible. It also established the extent to which the participant’s websites are accessible as well as the factors that motivate or prevent the participant from making a website accessible.

- **Accessibility testing**: *how the web developer tests the accessibility of a website they have created*. This section required the participant to demonstrate how they test the accessibility of a website. It also established how they tackle any accessibility problems, how frequently they test the accessibility of a website, how useful they find the feedback from accessibility tools, and how easily accessibility tests fit into the participant’s workflow.

- **Future improvements**: *how the web developer could improve the accessibility of the website they have created in future*. This forward-looking section investigated the type of help and information that the participant would find useful to develop accessible websites in future. It considered the form such information might take, the granularity of the information, when such help and information might be useful, and how it might fit into the developer’s workflow.

The full contextual inquiry schedule can be found in Appendix C: Contextual inquiry schedule on “Accessibility support for web developers”

### 3.4.1.2 Contextual interview procedure

Participants were interviewed face-to-face in their place of work. Although each participant was willing to take part in the investigation, some seemed a little uncomfortable with the
contextual interview being conducted at their actual workstation, in close proximity to their colleagues. The possibility of being overheard by colleagues also raised the concern that participants may provide socially responsible to certain questions, particularly those related to accessibility. For this reason, the procedure was split into two parts: a traditional face-to-face interview, taking place in private, away from colleagues; and the more interactive, contextual inquiry, taking place at the participant’s workstation.

Participants were given an introduction to the i2Web Project and its aims. They were assured of the confidentiality of the information they would be providing. Both audio and video were recorded for later transcription. The questions asked in the private face-to-face interview were those relating to: the web developer; communication; help and information; standards-compliance; users; and accessibility. The interview then shifted to the participant’s workstation, where the more interactive sections were conducted. These included: the workspace configuration; preview; validation; accessibility testing; and future improvements.

Contextual interviews lasted between 90 and 120 minutes. When the interview schedule was complete, participants were asked whether they had any further comments they would like to make about their experience of web accessibility. Finally they were thanked for their time and asked whether they would like a copy of the report on the results of the study.

### 3.4.1.3 Participants

An initial round of contextual inquiries was conducted with seven participants. All were male. Four were aged 21-30 and three were aged 31-40. All of the participants were from the UK. Three of the participants worked for i2Web consortium organisations. The others were recruited via web developer mailing lists, online forums, the i2Web project website and personal contacts in the industry. The participants had between 4 and 13 years of experience of web development, with an average of 8 years. Three participants worked for large enterprises, 3 worked for SMEs and one participant was self-employed.

A preliminary review of the contextual inquiry data by members of the i2Web consortium raised concerns over the type of web developer under investigation. It was expected that the more complex nature of Web 2.0 websites and applications would require the use an Integrated Development Environment (IDE). However, despite each of the seven web developers claiming to carry out Web 2.0 development, only two of them used an IDE, and then only for a minority of their development tasks. One developer used the IDE, ECLIPSE, but only for coding, with debugging and validation carried out using separate applications. Another developer used the IDE, Dreamweaver, but stressed that it was purely for the code-highlighting feature, and that he preferred an older version because it was much simpler. The other five developers used simple text editors such as Notepad++ or vim for all of their development work.

To ensure that a broad spectrum of web developers had been covered, a further round of contextual inquiries was conducted with six more participants. Again, all were male. One was aged 21-30 and five were aged 31-40. Three of the participants were from Ireland, and three were from Italy. Each of the participants worked for i2Web consortium organisations and was recruited specifically for their use of an IDE to develop Web 2.0 websites and applications. The participants had between 1 and 15 years of experience of web development, with an average of 10 years. Three participants worked for large enterprises and three participants worked for SMEs.
3.4.2 Results

The following sections summarise the initial findings from both rounds of web developer contextual inquiries.

3.4.2.1 Keeping up-to-date

- Each developer felt his\(^{18}\) knowledge of web development was either up-to-date or only slightly “behind the curve”. Web development was frequently described as a “constantly changing field” but many noted that this was a good thing as it kept them “on their toes”.  
- The areas that developers felt they had a good understanding of were HTML, CSS, JavaScript and PHP.  
- Where developers identified a gap in their knowledge, it was typically with newer technologies, such as HTML5 and CSS 3. They frequently reported spending their free time learning about these topics, out of pleasure as much as necessity.  
- Each of the developers attempts to keep their knowledge of the field up-to-date by reading various blogs and technical news websites. These include: Slashdot, The Register, Ars Technica, and A List Apart, as well as the blogs and websites of specific web developers.  
- Some developers use forums, mailing lists and message boards that are specific to a particular technology (e.g. WordPress Forums or Mozilla Developer Center). One developer reported using Twitter to keep abreast of current trends.  
- Whilst some of the developers have had specific training in web development or related subjects, the majority were self-taught, relying on online tutorials, such as those of W3Schools, to further their knowledge.  
- The freelance web developer feels a particular pressure to keep up-to-date. Citing humility as a vital aspect of working freelance, one developer noted: “You can get delusions of grandeur working alone with no one to question you but you find out how amazing you are when you get a full time job and realise how far you’ve fallen behind. The danger is getting out of step and not realising. You’re so used to doing everything yourself, that you convince yourself you’re brilliant.”  
- Though web developers appreciate the convenience of PHP frameworks and JavaScript libraries for easier development of Web 2.0 applications, many are fearful of becoming too dependent on them. One freelance web developer stated: “A lot of the frameworks I’ve looked at do all the really clever work for you, which is good, and I will learn them but I prefer to do things manually, to hand-code from scratch. It takes me longer doing it manually but that’s the good stuff. When you screw up, it really sticks with you and you learn so much.”

3.4.2.2 Workspace configuration

- Each of the developers uses a dual-monitor setup, with one monitor typically being used for the development environment, and the other for previewing webpages.  
- The majority of the developers work in a Linux or Mac environment, with some expressing a particular dislike for Microsoft environments.

\(^{18}\) As all the developers interviewed were male, the male pronoun will be used in this section.
• Each of the developers uses a standard keyboard and mouse. One developer reported using a graphics tablet in addition to this.
• The physical desktops of web developers tend to be sparse. Many of the developers reported that their work is carried out entirely online and that no physical artefacts are required.
• Whilst some of the developers have websites relating to a specific technology open whilst they work (e.g. nodeJS or Dive Into HTML5), the majority reported using Google to find relevant information as and when it is required.
• Web developers in large organisations and SMEs usually work in teams of 4-5 members. Despite this, they carry out the majority of their work alone, with tasks assigned to them by a project manager. The larger the organisation, the more distanced the web developer tends to be from the client.

3.4.2.3 Help and information

• Some of the developers are reluctant to ask their colleagues for technical assistance, either due to their own pride or a fear of “being mocked”. Others reported that they might ask a senior colleague or management for assistance with a business problem but would typically turn to Google to solve a technical problem.
• Forums, mailing lists and message boards are popular with a number of developers although the amount of time it can take for a problem to be solved means that the developer has often solved it in the meantime.
• Rarely do the developers rely upon printed materials to assist with a technical problem, with one commenting: “Books? What are those?” Many of the developers felt that books date very quickly and that up-to-date information is more readily available on the Internet.
• The type of information that the developers reported they are typically looking for after encountering a technical problem is code syntax, such as the documentation of PHP functions. For this, they will typically turn to the official documentation – “straight from the horse’s mouth”.
• Each of the developers stressed that they are usually reminding themselves of a solution rather than looking for completely new information: “I don’t hold as much in my head as I used to, so if is often really straightforward syntax type stuff on websites that if I strained myself I could probably remember anyway”.
• The developers reported that they would strive to establish the reasoning behind a particular solution in order to learn from it. However, time pressures mean that “fixes” are often copied without a thorough understanding of how they work.

3.4.2.4 Web development tools

• Many developers do the majority of their website coding using a simple, pared-down text editor, such as Notepad++ or VIM. These developers rarely use larger development environments, such as Eclipse or Dreamweaver. Some consider the basic functionality of Dreamweaver (e.g. code highlighting or FTP facilities) to be fairly useful but do not regularly use the application.
• One developer felt that using a simple code editor to carry out more complex programming was possible but “very long-winded”.

• Many developers use an Integrated Development Environment (IDEs). The most common IDEs are Eclipse (typically used within a Linux environment) and Microsoft Visual Studio (exclusively used within a Windows environment).

• Some developers exclusively use an IDE whereas others only use a text editor. Some developers use an IDE in conjunction with a text editor. Many, however, use a different environment depending on the task.

• The use of an IDE seems to depend on the type of development. For server-side development in languages such as ASP, Java, and .NET, where more complex functionality is required (e.g. debugging, class navigation etc.), an IDE is more commonly used. For client-side development in languages such as HTML, CSS and JavaScript, where only basic functionality is necessary (e.g. code completion, syntax highlighting) simple code editors are typically used.

• Although the appearance of an IDE is often fully customisable, the majority of developers use the default layout. The layout varies depending on the technologies currently in use.

• Although numerous plug-ins are available for IDEs, the majority of developers adopt the “out-of-the-box” default features of their IDE, with one user of Microsoft Visual Studio claiming, “The IDE has pretty much everything I need”. The integration of Software Development Kits for mobile devices into Eclipse also proved very popular with one developer.

• One developer uses a number of IDE plug-ins for different coding environments. To avoid conflicts and to reduce clutter, he prefers to run multiple instances of the IDE (Eclipse), each using a separate plug-in.

• Features that developers consider particularly useful in their IDE include:
  o Syntax highlighting
  o Code completion
  o Code generation
  o Autosuggest for names of variables / functions / classes etc.
  o Class hierarchy navigation
  o Refactoring – for changing variable names across the entire application or moving a class
  o Debugging

3.4.2.5 Code generation

• When creating a new web development project, an IDE typically creates a basic file structure, including a number of default styles and classes and a default HTML page. None of the developers have considered whether the generated code is standards-compliant.

• Web developers using IDEs tend to make use of code generation whereby HTML is generated from a template master. In many organisations, this master is created either in-house by a dedicated UI team or by an external graphic design team.

• One developer claimed that because each ‘child’ page inherits from a single template master, any validation and accessibility checks need only be carried out once. The developer did acknowledge however, that it is difficult to assess the validity of a template master because it is never actually rendered.
• The use of template masters and file includes means that the actual rendered page may be drawn from various different files. This poses a problem when it comes to validation and accessibility checks, as the errors and warnings produced by these tools often refer to line numbers in the code. The line numbers of a rendered page does not correspond to the line numbers of the individual files that are used to create it.

• One developer commented that his IDE (Microsoft Visual Studio) has a tendency to insert unwanted code. This was aptly demonstrated during a validation check when the developer discovered that a piece of code that was causing the page to fail had been inserted by the IDE. The developer also noted that using the preview pane of the IDE caused it convert HTML elements into uppercase (e.g. `<a>` into `<A>`) rendering them invalid.

• Few developers make use of the WYSIWYG interface built into many IDEs, which allows components to be dragged from a ‘toolbox’ onto the page and manipulated like a word processor application. For developers that do use the WYSIWYG interface, it is mainly used as a shortcut, in addition to manual manipulation of the code. For example, a developer may drag a calendar component onto the page using the WYSIWYG interface and then swap to the code editor interface to refine the generated code.

3.4.2.6 Debugging

• One developer claimed to “hate the web” because it does not have a true debugger. He noted: “You can get too easily get away with not debugging webpages. You cannot debug, you have to “print out” and validate.”

• The majority of developers consider validation and accessibility errors to be another form of debugging, which they address line-by-line, bug-by-bug. However, many of the developers prefer to keep code debugging separate from validation and accessibility errors. One developer stated, “I wouldn’t want to mix bugs and code warnings with compatibility issues. I prefer to compartmentalise what the problem is.”

• The use of ‘breakpoints’ that allow a developer to step-through and debug a web application are considered to be “where IDEs really come into their own”. One developer stated: “If you manually write something by hand, you can go in and render it, check it with the validator, and say ‘yes it works’. However, if you’re doing something specific to the code base in the back end, and if it is half-working or there’s some kind of a loop that isn’t working, to try and figure out what’s happening, you need something like this”.

• Developers who do not use the debugging feature of their IDE tend to “walk through” the web application manually, using it as an end user would. Other than inserting many log statements into the code, this was considered the only other way to accurately test dynamic pages or sequences of pages. One developer noted: “Usually I always test [dynamic sequences of pages] in the browser. I am sure there are proper methods to do it but that’s the way I do it”.

• With regards to the debugging features of IDEs, one developer noted: “That functionality in itself is the reason why 95% of people would use IDEs. We forgive the other downsides for that functionality”.
• The majority of developers use (and are extremely positive about) the browser extension FireBug, which provides debugging tools for web development. Some developers reported using the Web Developer Toolbar, although usually only a limited selection of functions from it. Other browser extensions and add-ons that are commonly used include optimisation tools, such as YSlow and Google Page Speed.
• FireBug allows code to be edited in the browser on the fly. One developer felt it was time-consuming to make a change to the code in the IDE, compile the code, and preview it in the browser. FireBug however, allows developers to quickly preview changes to the rendered code before committing the change in the IDE.
• One developer felt that FireBug was useful for static pages of content but not for dynamic web applications. This is because the developer would have to manually walk-through the application to get to a particular page, which the developer claimed would be boring and time-consuming, particularly for large applications.
• FireBug is considered particularly useful when a Content Management System has been used to automatically generate code. One developer noted that: “For debugging pages with a complex structure I don’t know very well (due to it being generated automatically), Firebug helps me in finding the point I’m interested in.”

3.4.2.7 Previewing

• The developers typically preview their websites using the latest version of Mozilla Firefox. Many also test websites in multiple versions of Microsoft Internet Explorer, as this is a notoriously troublesome browser. Occasionally Google Chrome, Safari or Opera are used but it is generally felt that if a website displays correctly in Firefox and Internet Explorer, it will display correctly in the others.
• Few developers make use of the ‘design pane’ or ‘preview pane’ built into many IDEs, which displays a visual rendering of the code. If anything, developers may take a quick look to “check if they are on the right lines” but most prefer to preview their code in the browser “as the user would see it”.
• Many developers prefer to preview their websites in the browser, as opposed to using the preview options of an IDE. A lack of trust was cited as a reason for this, with developers doubting the accuracy of the IDEs rendering engine. One developer noted, “Nobody’s going to use the rendering engine embedded in Eclipse, everyone’s going to use a web browser. It’s easier to be sure that the final layout is what you expect.”
• Developers of websites and applications for mobile devices prefer, where possible, to test using the actual device rather than an emulator. One developer commented: “If you see an interface on the emulator, it could be quite different on the device. If I have the device I would rather try it there. Then you have the same thing as what the end-user sees.”
• Previewing is more common when developing for mobile devices, according to one developer, who stated that he “switched a lot” between coding and previewing. “You write something, test if it works and so on. If you write a lot of code, maybe you will not find the error but if you write a line at a time, you will find it more easily.”
• The majority of developers preview their web pages in a variety of web browsers (Firefox, Chrome, Internet Explorer etc.). Users of Microsoft Visual Studio tend to
exclusively preview in Internet Explorer because it is tied into the IDE by default, although some developers acknowledged they could change this.

3.4.2.8 Validation

- Almost all of the developers have an accurate understanding of a standards-compliant website, focusing largely on W3C standards. One developer, however, appeared to confuse standards-compliance with accessibility conformance and felt that a site would be accessible if it validated to HTML and CSS standards.
- Validation checks do not easily fit into the workflow of many of the developers. The developers usually only validate their webpage once it has been completed or if a major change has been made. Typically, they only validate the homepage or a small sample of pages. Nevertheless, many of the developers find it satisfying to produce a page that passes validation with one claiming to “still punch the air when it happens”.
- Currently, if validation and accessibility are considered, it tends to be towards the end of the development process. One developer stated, “I like to get [the application] working before I start worrying about whether I’m missing a tag off the bottom or whatever. I get the application working to the customer’s requirements, and then look at compliance.”
- Although each of the developers feel that validation is important, some note that they are prepared to accept a page failing validation if it means they can use the latest, emerging progressive enhancements. Often, such enhancements are incorporated into browsers but are not yet incorporated into the official standards.
- Whilst all of the developers strive to create good, well-formed code, many of the developers feel that validation or accessibility checks are particularly necessary. Furthermore, the developers tend to take such issues into consideration only if the customer has explicitly requested them.
- Each of the developers stated that the websites they make are usually standards-compliant, often using the strict doctype. A number noted that pages that are largely generated dynamically using JavaScript may technically validate even if the generated code is invalid. It was also pointed out that Web 2.0 applications are something of an “accessibility nightmare”.

3.4.2.9 Validation tools

- The developers usually carry out validation using the W3C Markup Validation Service website, which is relied upon “as if it is an absolute, infallible authority”.
- Some developers feel they are unable to validate their code using the W3C validation service because it is running on a local server and does not have a public URL. None of the developers were aware of the ‘file upload’ feature of the W3C validation service.
- A number of the developers feel that the output of markup validators is not very helpful, with a single error often being responsible for many subsequent errors. Also, the developers consider the error messages to be quite useful, as long as they have an understanding of the underlying problem.
- There is a conflict between the errors and warnings reported by validation and accessibility checkers and the apparent correctness of the code. Having received a
warning from a validation tool that a piece of code is out-dated, one developer noted: “It probably is out-dated but it works. I wouldn’t overly worry about that”.

- Attitudes towards declarations of conformance (e.g. validation statements or the W3C “badges”) was mixed. Some of the developers feel these provide a useful indication of their commitment to web standards. Others feel they are rather dated and, because they rely upon self-certification, are often used falsely or incorrectly. As a result, some of the developers do not trust them.

- Although many of the developers are aware of the validation and accessibility features of their IDE, none of them currently use them. One developer enabled the accessibility toolbar of Microsoft Visual Studio for the first time during the interview.

- One of the reasons cited for not using the validation and accessibility features of an IDE was a lack of trust with regards to the accuracy and currency of the tools. The majority of developers stated that they prefer to get information on validation and accessibility “straight from the horse’s mouth”, be it from the W3C validation service or the WCAG guidelines. One developer noted: “with the W3C, it is just your page and the validator as opposed to the IDE’s interpretation of the validator”.

- Developers seem unconcerned about whether validation and accessibility testing is included in the IDE, with one noting: “It really doesn’t matter to me if I have to write something that is compliant whether it is in the IDE or the browser. It’s just a step in the process. If it’s in the browser it just happens a step later.”

3.4.2.10 Accessibility

- Almost all of the developers have an accurate understanding of web accessibility although very few are aware of WCAG or WAI-ARIA.

- One developer reported that compliance with level AAA of WCAG was not possible simply due to the amount of work that would be required to meet the necessary checkpoints. As a result, level AA was the standard they agree to meet.

- One developer felt that accessibility testing should come earlier in the development process because “At the end it is really bad because if you know about it at the end of the process then the chances are you have to go back and re-write everything to accomplish it. So the earlier the better.”

- Some developers feel that a number of the accessibility guidelines seem spurious or “more for the sake of it”. They happily comply with such guidelines but feel there is little information available to explain the reasoning behind them.

- Whilst each of the developers recognises the importance of accessibility, many reported that they simply do not have the time or money to be able to make their websites accessible. Often it is not in the requirements of a website. One developer noted that they “would be doing the business a disservice if I spent too much time on [accessibility]”.

- Accessible code will often be sacrificed if it means being able to use cutting edge technologies or proprietary tools. Often the customer will demand certain features that cannot be made accessible.

- A number of developers felt exasperated at “the disproportionate amount of work it takes to make a website accessible for the small proportion of people it benefits”. One developer went a step further and stated: “My project is not for these people”.
• Although the developers acknowledge that there are many aspects of making a website accessible (the content, the website, the browser, the assistive technology etc.), they each felt that responsibility is ultimately on the developer: “Most things that aren’t accessible that could have been accessible will be the developer’s fault”. Nevertheless, one developer did note, “everyone has to dance in time”.

• Almost all of the developers feel that the information on accessibility is extremely limited or, at least, not very well promoted. A number noted that accessibility problems are not usually born out of negligence or malice but out of a lack of information.

• Although the majority of developers had considered making websites accessible for users with visual impairments, very few were aware of other disability groups or how to make websites accessible for them. One developer stated “I have very little idea of how [disability users] use a computer, so it is hard for me to visualise them using it or even picture myself doing it”.

• Attitudes towards declarations of accessibility conformance were varied. Some developers felt that they were useful for advertising a company’s approach to accessibility. Others felt that they were pointless, stating: “If you have an accessibility problem and you want the site to work, the fact that there is an accessibility page wouldn’t change whether the site works for you or not.”

• One developer felt that adding accessibility information, such as ARIA roles or alternative content for assistive technologies, could lead to “cluttered” code. It was not felt that efficiency and maintainability would come from writing accessible code in the same way that it might come from writing standards-compliant code.

• Many of the developers felt that Flash and Silverlight technologies would “throw up a million more problems for accessibility” and were not aware of any ways of making them accessible.

• Some developers do not want to shoulder the responsibility of making their code accessible. Whilst they typically acknowledge the importance of doing so, they do not feel capable of making assessments of accessibility – particularly on issues that cannot be ‘programmatically determined’ and which require a subjective assessment. One developer referred to a “developer mindset” that is not always in tune with that of the user.

• Whilst some developers are interested in the reasoning behind different accessibility guidelines and are keen to discover how disabled users use websites, others are only interested in what it technically takes to make their code accessible.

• A number of developers feel that a dedicated accessibility ‘guru’ or user interface team would be a useful addition to a development team.

• Web developers are keen to be detached from the responsibility of establishing requirements, instead preferring to work to the specific requirements of the project manager.

• One freelance web developer felt that accessibility is an “invisible” enhancement and that “all that clients see is the price”. Web developers who incorporate accessibility are frequently undercut by those who are prepared to sacrifice accessibility in order to be able to offer a cheaper website.

• Many web developers do not want to be involved in testing and documenting the finished product. One developer commented that his team currently included a
documentation specialist who would create user guides and manuals based on his notes and comments.

- One developer feels that if he had to create accessible code, he would prefer very specific requirements (e.g. “make the background white and the foreground black”) as opposed to vague statements (e.g. “ensure that the colour contrast of the page is sufficient”). He noted: “In that case I know exactly what I have to produce. I can just care about how to organise my code, the most efficient, maintainable way for testing it. I don’t have to deal with the decision about where to show this piece of information or organise this menu. These are not things I care much about.”

### 3.4.2.11 Accessibility tools

- The developers felt that the hardest aspect of making a website accessible is in testing it. None of the web developers were aware of any accessibility tools for testing websites, although some had used tools that test components of accessibility, such as colour contrast. Another concern is the assistive technologies used by end users. One developer stated: “If I knew that the users had up-to-date tools that could understand up-to-date standards and were running them, that would help.”
- One developer felt that the vagueness of the feedback from accessibility checkers was off-putting. Receiving numerous warnings that “you may need to check the alt description of this image” was considered “unproductive”.
- Many developers rely upon a basic knowledge of accessibility or “rules of thumb”, such as checking images for alt descriptions or choosing appropriate background and foreground colours. One developer stated: “If I like it, I hope the end user will like it”. Another developer felt that disabling images and style sheets in the browser was a sufficient test of accessibility, claiming that: “you develop an innate instinct for what is going to work and what isn’t”.
- One developer felt that a Content Management System was particularly useful for ensuring that a website is accessible as it allowed him to test a website template once and be confident that it be accessible for subsequent websites.
- One developer feels that the best way to integrate validation and accessibility testing into his workflow would be through automatic testing, similar to the Selenium web application testing system. That way, a unit test could be written and performed automatically numerous times at various stages of development. The developer feels that current methods of testing are too boring and time-consuming.

### 3.4.2.12 Improving accessibility support

- Almost all of the developers stated that a “black-and-white”, quantitative checklist of accessibility problems would be extremely useful. Many acknowledged, however, that some accessibility problems simply cannot be “programmatically determined”.
- In terms of feedback, the developers felt that any suggestions of accessibility improvements to the code would need to be accompanied by the reasoning behind them. One developer noted: “I think it would have to be more example led, like a miniature blog-post, saying “this is how you should do it with an example”. Another developer suggested a bullet-point headline list of problems in “human language” with the more technical code suggestions behind each point.
• One developer felt that an aggregation of the existing, disparate accessibility tools available (such as the colour contrast checker) would be useful, rather than an all-in-one checking tool.
• One developer felt that integration of accessibility checking into their company’s existing unit testing procedure would be a good approach.
• One developer suggested a traffic light system that could be built into the browser alerting users as to the accessibility of a website and shaming those who do not strive to make their website accessible.
• Few of the developers stated that they would want accessibility tools to be integrated into their text editor. The stripped down, simple environment of such tools is what often attracts the developer to them in the first place.
• Many of the developers expressed a desire for accessibility tools that are incorporated into their browser. This is where they already preview and test their websites and is the stage of the web development process during which the developers would appreciate accessibility information. One of the mobile application developers stated that he would like to have accessibility tools in the device emulator. A number of developers noted that something like the FireBug debugging extension, but for accessibility, would be ideal.
• Developers felt that inline active accessibility tools in the code editor - working similar to a spell-checker - would be very annoying. Almost all of the developers stated that they would prefer to ‘action’ an accessibility check themselves, at a stage in the development process that they felt appropriate. In addition, many developers do not build in accessibility features until the very end of the development process and so would not want to be constantly reminded of their failings.
• A dedicated accessibility application would receive some support from developers. It was noted that this would have to work across different operating systems and environments.
• Many of the developers felt that simulating the experiences of different disability groups would be useful and would “settle arguments” over whether something is inaccessible or not. However, it was felt that this might require a lot of time to interpret. One developer stated that they would not want to have to run a lengthy simulation for every change they made to the webpage.
• One developer noted that he is fortunate enough to be able to test websites using a range of assistive technologies and software. Recognising that this is not always possible due to the high cost, he suggested that some form of online simulation of the popular assistive technologies (e.g. JAWS, Supernova, ZoomText, ClaroRead etc.) would be useful.
• Some of the developers felt that accessibility information would be necessary at various stages of the web development process. They felt it would need to be available during the planning stages before the developer even starts on the code, during the coding stage whilst the developer is working on the code, and also once the website has been completed, to reflect on the code.
• One developer felt that he had only started taking accessibility seriously after watching a conference talk by a blind user demonstrating how inaccessible websites could be. The developer felt that simple videos of disabled users of websites would be a very effective educational tool.
• As well as tools that assist in the creation of accessible websites, one developer suggested a tool that would accurately log and report accessibility problems that have been encountered in the field, allowing the developer to recreate and remedy the problem.
• With regards to integrating accessibility support into their IDE, one developer felt the structure and format of his IDE should be maintained, “rather than something mad popping in from the left”. One developer suggested that accessibility support could be incorporated into the existing debug toolbar of most IDEs.
• One developer feels that accessibility plug-ins for IDEs would be useful. He commented: “I think it would be good for executing tasks and recording the results in a nice way. If the tasks fail for some reason (e.g. non-compliance), it would be good if it could say, “this task is failing for this reason – see WCAG...” and point me to the relevant code”.
• The use of “wiggly underlines”, similar to the inline spelling and grammar indicators in Microsoft Word, is considered by many developers to be potentially helpful, because it would draw attention to accessibility problems without being too obtrusive.
• Developers are suspicious about any form of “wizard” interface or avatar assistance (such as ‘Clippy’ in Microsoft Office), assuming that any code generated by such tools would be incorrect and have to be manually refined. One developer said, “In my experience, such tools create code I wouldn’t like to have, so I have to tweak it to make it the way that I want.”
• Developers are strongly against any accessibility enhancements being made automatically and want to be able to review any suggestions before actioning them.
• One developer felt that inline active accessibility tools would be useful stating, “What I want is a development tool that jumps up and tells me “that’s incorrect” or “consider this there, or that there”. However, the same developer also said, “Hints would be quite handy when coding but at the end of the day, you need something to validate what you've actually implemented, rather than the hints just suggesting you do.”
• The majority of web developers are happy with the “checklist” format of problems offered by current accessibility tools.
• Many of the developers felt that simulating the experiences of different disability groups would be useful. However, it was felt that this might require a lot of time to interpret, especially considering the range of disabilities.
• Some developers felt that current guidelines are too arbitrary and detached from the reasoning behind them. One developer feels that being told, for example, that blind users may struggle to reach a particular section of the page would be more useful than “some standard that says, “because of standard 64.XYZ please do this”. The developer went on to say “If you actually gave a developer a reason, I think, more than anything, that would probably make them more aware and realise that by not doing this you are omitting every blind person out there from using your website. Something along those lines would give you a very valid reason for doing it as opposed to just trying to hit a standard. It would personalise it. That would be an attractive feature, definitely.”
3.4.3 Discussion and Conclusions

The chapter presented the methods used to investigate the working practices of web developers along with initial results of the analysis of the data collected. This aspect of the investigation began with an online survey completed by 60 web developers that provided an initial understanding of the technologies, tools and techniques used. This was followed up by a series of in-depth contextual inquiries with 13 web developers, which allowed the complex work practices of this cohort to be explored.

3.4.3.1 Development environments

One of the more interesting and unexpected findings to emerge from the investigation was a distinct split between web developers who prefer to use a minimalist coding environment, such as Notepad++ or VIM, and web developers who prefer to use an Integrated Development Environment, such as Eclipse or Dreamweaver. This was evident in the results of both the online survey and the in-depth contextual inquiries. It was suspected by members of the i2Web consortium that the type of environment might reflect the type of work being undertaken, with the presumption that Web 2.0 developers would require a more sophisticated development environment. This does not appear to be the case as both the survey and contextual inquiries have shown that both IDE and non-IDE developers conduct Web 2.0 development. It may depend upon the specific task being undertaken, with some developers claiming to use both an IDE and a text editor, depending on whether they are carrying out server-side or client-side development. Another possibility is the web developer’s background, with those from a more traditional software development background preferring the familiar features of an IDE.

3.4.3.2 Accessibility tool integration

Few of the developers stated that they would want accessibility tools to be integrated into their text editor. The stripped down, simple environment of such tools is what often attracts the developer to them in the first place. Developers who use IDEs are more open to the possibility of accessibility tool integration. However, despite the flexibility and extensibility of an IDE, many developers were keen to “reduce clutter” and only install plug-ins and extensions that are absolutely necessary. It was noted that the majority of developers adopt the “out-of-the-box” default features and layout of their IDE. With regards to integrating accessibility support into their IDE, one developer felt the structure and format of his IDE should be maintained, “rather than something mad popping in from the left”. Another developer suggested that accessibility support could be incorporated into the existing debug toolbar of most IDEs. It is clear from these findings that any integration of accessibility support into an IDE should be as seamless and unobtrusive as possible.

3.4.3.3 Code generation

Although code generation is a popular, timesaving feature of IDEs, there was concern from some developers about the validity and accessibility of the generated code. Where code is generated from a template master, a number of developers acknowledged that it was difficult to assess the validity and accessibility of the code because the master is not actually rendered in isolation. Similar, the majority of developers avoid using the WYSIWYG interface
built into many IDEs. They are doubtful of the validity and accessibility of the code that such a feature might generate. Those who do use the WYSIWYG interface typically use it as a shortcut, in addition to manual manipulation of the code. These results suggest that developers are willing to make use of code generation providing they are able to trust that the generated code is valid and accessible.

3.4.3.4 Debugging

Although the debugging features of IDEs are extremely popular, and allow web developers to step line-by-line through an application, many developers prefer to debug their code in the web browser. Some developers prefer to keep code debugging separate from validation and accessibility errors and would not want them mingled together in their IDE. Others felt that it was difficult to test dynamic pages or sequences of pages in an IDE and the only way to achieve this was through the browser. To this end, the debugging browser extension, FireBug, proved extremely popular with the majority of developers. It allows code to be edited in the browser on the fly and allows developers to quickly preview changes to the rendered code before committing the change in the IDE. Many developers felt that a similar browser-based extension for accessibility support would be appropriate.

3.4.3.5 Previewing

Web developers frequently preview what they are working on to ensure everything is as expected and that any errors can be swiftly discovered. Almost all of the developers preview their websites in a range of web browsers or, for mobile development, by using an actual mobile device. Few developers make use of the ‘design pane’ or ‘preview pane’ built into many IDEs, which displays a visual rendering of the code. A lack of trust was cited as a reason for this, with developers doubting the accuracy of the IDEs rendering engine. Developers prefer to experience their website or application in the same way that the end-user will. These results suggest that any integration of accessibility support into an IDE must forge tight links with the browsers and devices in which developers prefer to preview their websites and applications.

3.4.3.6 Validation

Standards validation checks do not easily fit into the workflow of many of the developers. The developers usually only validate their webpage once it has been completed or if a major change has been made. Although each of the developers feel that validation is important, some note that they are prepared to accept a page failing validation if it means they can use the latest, emerging progressive enhancements. Some developers tend to consider validation only if the customer has explicitly requested it. The increasing reliance upon JavaScript to dynamically generate pages was a concern to some developers who noted that the page might technically validate even if the generated code is invalid. It was also pointed out that Web 2.0 applications are something of an “accessibility nightmare”. It is clear that whilst developers appreciate the quantitative, checklist nature of standards compliance, more effort is needed to incorporate into their workflow.
3.4.3.7 Validation tools

Although many of the developers are aware of the validation and accessibility features of their IDE, none of them currently use them. One of the reasons cited was a lack of trust with regards to the accuracy and currency of the tools. The majority of developers stated that they prefer to get information on validation and accessibility “straight from the horse’s mouth”, be it from the W3C validation service or the WCAG guidelines. The output of the W3C validation was considered by developers to be discouraging due to chains of dependencies resulting in an over-inflated number of errors reported. Also, the misuse of validation statements and the W3C “badges” of conformance has resulted in many developers not longer trusting them. Any attempt to incorporate such features into an IDE must ensure that developers are aware of the legitimacy and accuracy of the tool.

3.4.3.8 Accessibility

Attitudes towards accessibility are quite bleak. Few of these developers were aware of WCAG in spite of being aware of accessibility issues. Whilst each of the developers recognises the importance of accessibility, and accepts that it is largely their responsibility, many reported that they simply do not have the time or money to be able to make their websites accessible. As with standards compliance, accessible code will often be sacrificed if it means being able to use cutting edge technologies or proprietary tools. Developers reported that customers would often demand certain features that cannot be made accessible. Almost all of the developers feel that the information on accessibility is extremely limited or, at least, not very well promoted. Developers do not feel capable of making assessments of accessibility – particularly on issues that cannot be ‘programmatically determined’ and which require a subjective assessment. One developer feels that if he had to create accessible code, he would prefer very specific requirements (e.g. “make the background white and the foreground black”) as opposed to vague statements (e.g. “ensure that the colour contrast of the page is sufficient”). A number noted that accessibility problems are not usually born out of negligence or malice but out of a lack of information. Work is clearly necessary to prevent accessibility from being seen as a time-consuming optional add-on to web development.

3.4.3.9 Accessibility tools

The developers felt that the hardest aspect of making a website accessible is in testing it. Many developers feel that the current methods of accessibility testing are too boring and time-consuming. None of the web developers were aware of any accessibility tools for testing websites, although some had used tools that test components of accessibility, such as colour contrast. The vagueness of the feedback from accessibility and validation checkers was considered off-putting by a number of developers. As a result, many developers rely upon a basic knowledge of accessibility or “rules of thumb”, such as checking images for alt descriptions or choosing appropriate background and foreground colours. It is evident from these findings that accessibility tools must seek to educate and inform web developers without consuming too much of their time. Accessibility tools need to organize information in different ways from current practices and provide more information, particularly the reasoning behind accessibility problems, which they generally do not do at the moment.
3.4.3.10 Improving accessibility support

The points of particular interest for the i2Web Project that emerged from these contextual inquiries with web developers are:

- Despite acknowledging that some accessibility problems simply cannot be "programmatically determined", almost all of the developers want a clear, quantitative checklist of accessibility problems.
- Developers want accessibility tools to provide not only solutions and suggestions to improve the code but also the reasoning behind them. This could be in the form of a bullet-point headline list of problems in clear "human" language along with the more technical code suggestions behind each point. Simple videos of disabled users of websites are also considered to be a very effective educational tool.

The integration of accessibility testing into existing unit testing is popular with developers. Citing the Selenium web application testing system as an example, developers suggest that accessibility unit tests could be written and performed automatically numerous times at various stages of development.

- Few developers want accessibility tools to be integrated into their text editor. The stripped down, simple environment of such tools is what often attracts the developer to them in the first place. Developers who use IDEs are more open to the possibility of accessibility tool integration.
- Many developers expressed a desire for accessibility tools that are incorporated into their browser. This is where they already preview and test their websites and is the stage of the web development process during which the developers would appreciate accessibility information.
- Should accessibility tools be incorporated into their IDE or text editor, developers are keen to retain as much control as possible. Developers feel that inline active accessibility tools in the code editor - working similar to a spell-checker - could prove very annoying. Developers prefer to ‘action’ an accessibility check themselves and review the suggestions, at a stage in the development process that they feel appropriate. Furthermore, developers are suspicious about any form of “wizard” interface or avatar assistance (such as ‘Clippy’ in Microsoft Office), assuming that any code generated by such tools would be incorrect and have to be manually refined.

Developers are open to the idea of simulating the experiences of web users with disabilities as a tool in developing accessible websites and applications. However, they are cautious about the amount of time it would take to run and, in particular, to interpret the results from such simulations. As well as simulating disabilities and impairments (such as partial sight or colour blindness), developers are keen to reduce costs by simulating popular (and expensive) assistive technologies (e.g. the output from a screen-reader).

Overall these in-depth contextual inquiries produced much detailed information about the work practices and accessibility requirements of web developers that had not been elicited in previous surveys and interviews. Therefore we recommend this method for understanding the needs of stakeholders in the accessibility chain.
4 Requirements for Web accessibility experts

4.1 Introduction

There are a number of factors which need to be in place to ensure that websites and web applications are to be made accessible: accurate and easy to understand guidelines, tools to effectively support accessible development and tools to effectively support testing of websites and applications for accessibility. Surprisingly, we could find very little research on the needs of accessibility evaluators for the kinds of tools that best support them in conducting accessibility evaluations. This is in spite of the fact that many tools have been developed (the WAI website lists over 125)\(^{19}\), but it is not at all clear what requirements elicitation exercises were conducted to guide the development of these tools.

Van Isacker et al. (2010) conducted an online survey of both web developers and accessibility assessors about their knowledge of web accessibility and the tools they use. Unfortunately it is not clear whether “accessibility assessors” refers to individuals who are expert in web accessibility or individuals who are tasked to assess the accessibility of a website, but who may not be experts on web accessibility. Both groups are important audiences for web accessibility evaluation tools, but will have very different needs. Van Isacker et al. found that less than half (43%) of their accessibility assessors would be interested in using automated accessibility testing tools (surprisingly, they did not appear to ask them which tools they use), but nearly all (90%) would be interested in simulators of assistive technologies such as screenreaders. Unfortunately this survey did not dig deeply enough into accessibility assessors current practices or unmet needs in order to inform the design of new tools.

This paper reports on a study in which we conducted in-depth interviews with expert web accessibility evaluators, to explore their current practices, the tools they use, how well these tools support their work and what unmet requirements they have.

4.2 Method

4.2.1 Participants

14 web accessibility evaluators participated in this study. 3 were women and 11 were men. The age breakdown of the interviewees is presented in Table 4.1, below. Interviewees had an average of 8.7 years of experience with accessibility evaluations with the range being 0.5 years to 13 years.

<table>
<thead>
<tr>
<th>Age Range</th>
<th>Number of Interviewees</th>
</tr>
</thead>
<tbody>
<tr>
<td>20-29</td>
<td>2</td>
</tr>
<tr>
<td>30-39</td>
<td>4</td>
</tr>
</tbody>
</table>

\(^{19}\) [http://www.w3.org/WAI/ER/tools/complete]
The interviewees were selected from a variety of contacts both within and outside the i2Web Project. Four organizations within the project participated, providing a broad base of participants from both private and public organizations: FIT (2), NCBI (1), Public-I (1), University of York (2). For the remaining 8 interviewees, six were from private organizations and two from public organizations.

10 organizations were small to medium enterprises (SMEs) (with seven being very small organizations with less than 50 employees) and four were large organizations. For all of those reporting from large organizations and for one of the small organizations, the interviewee worked in a smaller unit/group within the organization.

From all organizations, three reported that they only did evaluations of web applications internal to the company, while two reported that they do only evaluations for external clients. The remaining nine reported that they do evaluations for internal and external clients, with two saying their work was more heavily weighted to external clients.

Interviewees were informed that all answers to questions during the interviews would be kept confidential and information would only be released in aggregate form.

For interviewees external to the project, they received a gift certificate from the Amazon online bookstore to the value of 30 GBP/30 Euros.

## 4.2.2 Interview schedule

The interview schedule was prepared through an iterative design process. The initial themes for the interview schedule were drawn from both the relevant literature on web accessibility evaluation as well as the experiences of partners in the project consortium. These themes were vetted with members of i2Web Project and then comments and questions were proposed for inclusion and review.

The themes covered within the interviews were the following:

- The organization and its culture regarding accessibility evaluations
- The experience and training of the interviewee in accessibility evaluations
- The tools used during automated and/or manual accessibility evaluations
- What features in the tools are used most commonly and how useful are those features to the evaluators
- Sampling methods used to select web pages for testing
- Tracking of tests performed on web pages
- Reporting to developers/commissioners
- Procedures for maintenance of web accessibility

The full interview schedule can be found in Appendix D: Web Accessibility Evaluator Interview Schedule.
4.2.3 Procedure

Two interviews were undertaken as face-to-face interviews, with the remaining 12 being conducted as phone interviews.

Participants were briefed regarding the overall purpose of the i2Web Project at the beginning of the interview, that being to produce better tools for accessible design and accessibility evaluations in the areas of Web 2.0, mobile web access and internet TV. Further to this, the participants were informed that the interview was targeted at trying to understand how people undertook accessibility evaluations, in terms of both tools and processes, in order to better inform the design of future tools.

The interviews lasted between 30 and 45 minutes, with the interviewer conducting a semi-structured interview, with questions on themes being prompted by the interviewees comments rather than following the order set in the interview schedule.

The interviewer kept notes during the interview and further, all interviewees agreed to having the interview recorded for purposes for analysis.

4.2.4 Data Analysis

A content analysis was undertaken on the data collected from the interviews. The data were analysed using a closed coding scheme, with a set of a priori codes being chosen from the themes developed in the interview schedule preparation. These a priori codes were:

- Organizational culture
- Personal experience
- Automated testing
- Manual testing
- User testing
- Testing tool
- Testing feature
- Guidelines
- Reporting
- Tracking
- Maintenance

4.3 Results

The following sections discuss the key findings from the analysis of the interview data.

4.3.1 Training and Experience

While many of the people involved in the interviews had more than 10 years of experience in web accessibility, none of the interviewees have ever undertaken formal training in accessibility. Four interviewees reported having training in screen-reader software, specifically JAWS, while the others reported being self-taught in terms of accessibility.

The reasons reported by interviewees as to why they had not undertaken training varied. The most common was that there was no training available at the time when they started
working in accessibility. Certainly that was the case that in 1999, a commonly reported starting point for evaluators that coincides with the release of Web Content Accessibility Guidelines (WCAG) 1.0. At that time, most people relied on the documents provided by the Web Accessibility Initiative (WAI). In many cases, evaluators still heavily rely on these documents to inform them about accessibility issues. When asked about why they had not undertaken training since their initial exposure to accessibility as a topic, most people responded that they had learned more through trial and error and experience evaluating than they believe a training course could provide to them.

When asked if they intended to take any training, almost all responded no, feeling that they had a depth of experience that goes beyond most training available, with only “edge cases” or “ambiguous situations”, such as new Web 2.0 technology, being areas for improvement in their knowledge.

Two people reported that cost was one of the factors keeping them from training, with many training courses being more of an investment than they were willing to make. Another key factor was the time required to commit to attending such training. Many evaluators came from small organizations or small teams from which the loss of an individual to training would represent a substantial loss of productivity and output.

In addition to this, a number of people commented that the “edge cases” and “ambiguous situations” are often due to the varying interpretations that can be applied to the WCAG guidelines and accompanying documents. Several participants said that the documentation provided by WAI is confusing, contradictory, or in some cases, incorrect in the opinion of the evaluators. The interviewees were unsure how training would help them in those situations.

4.3.2 Participation in Guidelines/Standards Bodies

Two of the interviewees currently belong to W3C WAI working groups. Although many people identified themselves as “curious” or “lurkers” when it came to following the WAI’s activities, most indicated that they did not have time to involve themselves in the “political machinations” of WAI. Indeed, three of the interviewees had declined invited expert status in WAI primarily due to the time pressures that would be put on them from such participation. A secondary reason for declining participation was the understanding that their organization would need to join WAI formally to continue activities started during the invited expert period.

Three of the interviewees reported that people in their organization are involved in other standards efforts such as IMS Global, the International Standards Organization (ISO) and the British Standards Institute (BSI).

From these results a question is posed to communities of practice in accessibility: how can people who are knowledgeable about evaluation and want to participate become involved in a more lightweight way than is currently available?

4.3.3 Tool Functionality

During the interview, each interviewee was provided with a list of types of features that are provided in a variety of WATTs and AATTs. Each interviewee was asked whether the WATT and/or AATT that he/she uses or has used in the past had each type of feature and how
useful was each of these to them during testing. The list of features used in the interview was the list of features surveyed in work by Trewin et al. (2010).

Table 4.2, below, shows the number and percentage of interviewees who now use or have used each type of feature. The most commonly used features of tools were: the ability to choose specific guidelines to test such as WCAG 1.0 versus Section 508 (50%, 7 evaluators), receiving a list of problems automatically tested by the tool (50%, 7 evaluators) and the list of problems encountered in the web page (50%, 7 evaluators). There is very little use of functionality that helps in understanding the problems that are reported by tools. For example, visualizations of problems that users will encounter are only used a little over a third of the time (35.7%, 5 evaluators). Also, evaluators seldom use options to help in prioritization of problems (21.4%, 3 evaluators) or the severity of problems found (21.4%, 3 evaluators), as most evaluators reported that they would prefer to rely on their own judgement for such decisions. Of particular note, WATTs are not used to maintain a checklist of what tests have been undertaken on a web page or within a website.

Finally, there is very little use made of any tools that would help pinpoint errors in JavaScript (14.2%, 2 evaluators). Indeed, for those interviewees who knew if their tools had such a feature, they reported that their tools either did not have the functionality to do such tests, or that such tests are so rudimentary that they are not useful in understanding what the accessibility issues are with the code.

Table 4.2: Functionality used in WATTs and AATTs

<table>
<thead>
<tr>
<th>Evaluation Tool Feature</th>
<th>Frequency of use: % (N)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Choose guidelines to be tested</td>
<td>50.0 (7)</td>
</tr>
<tr>
<td>Explanation of guidelines tested</td>
<td>50.0 (7)</td>
</tr>
<tr>
<td>List of problems automatically detected</td>
<td>50.0 (7)</td>
</tr>
<tr>
<td>Pinpoint problem in source code</td>
<td>42.9 (6)</td>
</tr>
<tr>
<td>Rendered visualization of problems</td>
<td>35.7 (5)</td>
</tr>
<tr>
<td>Explanation of problems found</td>
<td>28.6 (4)</td>
</tr>
<tr>
<td>Pinpoint problem in the Document Object Model (DOM)</td>
<td>21.4 (3)</td>
</tr>
<tr>
<td>Prioritization of problems found</td>
<td>21.4 (3)</td>
</tr>
<tr>
<td>Indication of severity of problems found</td>
<td>21.4 (3)</td>
</tr>
<tr>
<td>Example solutions</td>
<td>14.2 (2)</td>
</tr>
<tr>
<td>Pinpoint problem in JavaScript code</td>
<td>14.2 (2)</td>
</tr>
<tr>
<td>Explanation of code fragments causing problems</td>
<td>7.1 (1)</td>
</tr>
<tr>
<td>Run a preset interaction sequence (e.g. auto fill in of forms)</td>
<td>0</td>
</tr>
<tr>
<td>Automatic testing of all application states</td>
<td>0</td>
</tr>
<tr>
<td>Checklist for manual tests</td>
<td>0</td>
</tr>
<tr>
<td>Pinpoint problems in servlet</td>
<td>0</td>
</tr>
</tbody>
</table>
There were a number of surprises when discussing aspects of the functionality of WATTs and AATTs with the interviewees. The first issue that was raised was that of terminology. A number of evaluators used the terms “automated testing” and “manual testing” interchangeably. In addition, many of the evaluators referred to options that are typically considered to be aids in manual evaluation, such as the ability to turn off JavaScript or to reveal alternative text on pictures when assessing a web page, to be “automated tests”. But in these cases, the decision-making is not automated, although the evaluator is using a tool that speeds up manual testing. Similarly, when asked about tools for manual testing, evaluators often discussed at length what assistive technologies they would use in “manual testing”. Finally, some evaluators referred to testing in alternate browsers as being a type of manual testing accessibility tool. This is an interesting variation in the use of accessibility terminology that has the potential to cause confusion in the future when providing testing tools or methodologies for testing.

While many evaluators stated that their tools did have automated testing options, very few used AATTs on a regular basis. When asked why they did not use AATTs, interviewees often responded that it was because they did not trust the tools to detect problems as well as a human. Further, there was a deep distrust of the information that came from AATTs, with many evaluators being aware that the tools only tested a small number of checkpoints or success criteria in WCAG 1 or 2. Another factor that was indicated as a serious problem is the fact that the majority of tools do not test WCAG 2.0 success criteria. Indeed one interviewee commented that he was surprised to find that his testing tool had replaced the ability to select to test to a particular level of WCAG1.0 with a button that was simply labelled “Test WAI”. Further investigation by the interviewee revealed that this button in fact still tested to WCAG 1.0. Finally, a recurring theme among the interviewees was that the automated tools often produced test results that were difficult to understand or errors and warnings, or worse yet, results that were wrong in their understanding of the guidelines being tested.

Due to this lack of use of automated testing, many evaluators discussed their use of tools that aid them in manual testing in detail. In terms of the how manual testing is undertaken, a number of evaluators use “accessibility toolbars” to help them do manual testing. Interviewees discussed all of the following:

- WAVE toolbar
- Web Developer’s Toolbar
- Web Accessibility Toolbar from the Paciello Group
- AIS Web Accessibility Toolbar

Most evaluators discussed using functionality such as removing images and replacing them with alternative texts to speed up testing, or use of features that turn off JavaScript, CSS or other options. Others used these tools to visualize the markup of a page on the page rendered in the web browser.

Some interviewees use visualizations of a web page that show how users with disabilities will perceive it to assist them in testing. The most commonly mentioned visualization was a greyscale version of a webpage. This may help to understand how people who have a colour vision impairment or partially sighted people with poor contrast perception will see a page.

---

20 See definitions of “automatic testing” and “manual testing” in section 1.1.
But in most cases the interviewees stated they did not find such visualizations very useful, largely because they do not tell them anything about how the problem relates to the technical aspects of the webpage being viewed.

Evaluators also often use the assistive technologies that disabled web users use to access the web to assist them in understanding how webpages will be presented to different users. For example, some evaluators use a screenreader to interact with a webpage to understand the experience that a blind person will have with that page. Many of the interviewees commented that while they are not as expert as many blind people in using a screenreader, the experience of listening to the screenreader often revealed problems that otherwise would have gone unnoticed.

Many evaluators also use tools that are not specifically designed for testing accessibility (nWATTs), such as development tools and code validators. These are used to understand the code of a webpage. In particular development tools are used to select elements from a rendered page and move to the code to investigate problems. While a minority of people found the web accessibility toolbar they used helpful and useful in locating problems in the code, the majority used tools such as Firebug, an extension of the Firefox browser, or similar functionality in other browsers such as Safari.

4.3.4 Completeness of accessibility tests

There were very few accessibility evaluators who reported doing full conformance audits of websites using a full set of guidelines, such as WCAG 1.0/2.0. Most evaluators stated that this was due to the amount of time it takes to do such audits, especially if all pages in a web application need to be tested. However, in many cases they reported that few companies request conformance accessibility testing and that when such testing is offered it is declined due to the cost involved in performing the audits. Evaluators report that the majority of companies requesting accessibility testing do so to gain an idea of where their website is in terms of accessibility to users, or to try to show due-diligence in process in testing accessibility, rather than to show conformance to a particular set of guidelines or legislation.

4.3.5 Sampling Pages for Accessibility Testing

There were very few accessibility evaluators who do evaluations on all web pages in any given web application or site. In some cases, this is due to the fact that there are thousands of pages within the web application/site being testing; however, more commonly it was because the commissioners of the web applications did not want all pages tested due to the cost of testing on such a large scale.

As a result, most web evaluators test a sample of pages from a web application/website using a variety of different sampling strategies.

First, almost all evaluators mentioned having discussions with the web commissioners or developers about what the most important pages are in the web application/site to be tested. In this application-centric sampling strategy, the definition of “important pages” changed depending on the evaluator and the client with whom he/she is working. For example, an important page might be one that has critical information about the organization. Alternatively, it could be one that has potentially tricky interactive
components (e.g. complex forms, Flash animations). In most cases, what was considered an important page was open to interpretation.

Nearly all evaluators talked about a template-centric sampling strategy, where they identify a collection of pages that have a variety of different templates that are shared throughout a web application/site. This was often referred to as a “representative set” of pages by interviewees. Examining pages that share a particular template and identifying those with complex interactions then adds to this representative set. This strategy is a variation on the ad hoc sampling strategy discussed in various sources (Velleman, Velasco, Snaprud and Burger, 2006; Brajnik, Mulas and Pitton, 2007; W3C/WAI, 2011).

One common strategy mentioned for selecting web pages within a web application/site is that of using user goals or, as some evaluators referred to it, “user journeys” in a website. This user-centric strategy involves working with the development team or those who commissioned the website to identify key user goals and tasks in the application/site. These goals are then used to identify path that users will take through a website to achieve the goal. For example, in a banking website a key feature would be for users being able to find out their current balance. With such a goal in mind, the evaluation team records all pages that are along the expected paths of interaction from the home page to that end goal. The collection of pages from a number of these types of goals is then used to form the sample of pages tested in an evaluation. When asked why they use this strategy, the majority of interviewees who used it responded that it was because such strategies make it easier to communicate with the web developers and commissioners about where problems are in a web site. One evaluator stated:

“[We] always try to have testing be [related to] the use of a specific task to see if there are problems along the critical path so that you can say to a commissioner ‘A user can’t buy something on your page’ rather than ‘there is a missing alt text’.”

This quote exemplifies one of the transverse themes that cut across the interviews. Many evaluators felt that communicating with clients in terms of checkpoints, success criteria and guidelines has little impact on commissioners and developers. They felt that unless the business could relate the problems to concrete terms, either in terms of how the problems affect users or the impact to their business, there was a very low chance that recommendations that came out of testing would be used to improve a website or web application.

Finally, a minority of interviewees discussed using accessibility violations detected by automated tools or by early evaluations of an application as an indicator of what pages to test. These evaluators are using a similar approach to the sampling method discussed in King (2005).

No interviewees discussed using random sampling strategies such as those discussed in Henzinger et al. (2000) and Brajnik et al. (2007).

4.3.6 Tracking of Accessibility Tests and Problems

One of the most interesting aspects of the interviews was how evaluators approached the tracking of accessibility issues in a web application/site. This might include: what tests have been completed, what pages have been tested and what problems have been found. In descriptions of how pages and tests were tracked, there was a wide range of strategies and
methods used that comprised a style of testing. The style of testing usually varied in how much pre-planning went into performing tests. The four styles of approaches that were discussed can be described as the following21:

**Strategic:** The evaluator has a matrix comprised of two dimensions: the list of pages to be tested and the guidelines against which each page should be compared. In some rare cases this approach relates to full accessibility audits, but more often it is a subset of a particular set of guidelines being tested. Each page is tested with each test being performed in sequence. This matrix will then often serve as part of the report to the website commissioner/developer.

**Tactical:** The evaluator has a set of tests that will be performed on each page. Some evaluators use a test-centric approach, where one test is performed on each page, while others prefer a page-centric approach, where each page has a set of tests performed on it; however, the order of the tests is not necessarily the same for each page. This style of testing is often geared towards testing “important” aspects of the web application/site first (e.g. large forms, media players) and then following up with other tests. Usually this style of testing includes a standard document template that is used for tracking tests done and issues raised in tests.

**Opportunistic:** The evaluator performs concentrated testing only on a few pages, such as those where automated testing indicates there are a large number of accessibility problems.

**Scrambled:** The evaluator has only very minimal plans for testing. They keep the pages tested and the tests performed on each page in his/her head. Records are kept in a somewhat haphazard way and then compiled into a document afterwards for reporting purposes.

There was very little commonality in how evaluators approached testing from a methodological standpoint. The only true commonality was that usually problems were recorded in lists in word processing or spreadsheet programs. One interviewee commented:

“... if you could provide a tool that handles [tracking the tests completed] on webpages for me that would be great...”

This indicates that from a technical point of view there are some very simple solutions that would provide major benefits to evaluation teams.

### 4.4 Common Accessibility Problems

Interviewees were asked what the most common accessibility violations and problems are found in web applications/sites.

All interviewees responded that alternative text for images remains a serious problem on the web. Problems associated with alternative text include:

- Informative images without any alternative text
- Informative images without appropriate alternative text (e.g. too long, too short, incorrect)
- Decorative images with extremely detailed alternative text – an example of what one interviewee referred to as “accessibility gone mad”

---

21 Terminology for styles of evaluation borrowed from Hollnagel’s (1998) Cognitive Reliability and Error Assessment Methodology (CREAM)
Some of the interviewees commented that they believe that many of the alternative text problems are related to Content Management Systems (CMSs). Specifically, CMSs do not support authors in providing alternative texts at appropriate times, or worse, the CMSs automatically fill in alternative texts with inappropriate content (e.g. filenames). One interviewee pointed to the lack of advice regarding accessibility in authoring tools as being a serious problem that is only going to get worse as more and more systems move to involving end-users providing content for web applications (e.g. YouTube, Facebook).

Almost all interviewees mentioned form markup, specifically field labels and tab ordering between fields, continuing to be in a terrible state on the web in general. One interviewee went as far to say: “developers still just don’t get it”.

Related to this, most interviewees commented that keyboard access on the web is still not terribly well supported. Keyboard traps or completely keyboard inaccessible functionality are in the modern web, with it being particularly problematic with Flash content.

Finally, it is clear that the proliferation of JavaScript functionality, either through jQuery or the use of asynchronous JavaScript and XML (AJAX), is causing a number of problems for both users and evaluators. Evaluators complained about problems relating to the non-standardized implementations that people use just to get something “cool or shiny” up on the web. Spaghetti code (code that does not follow a logical sequence in presentation) is prevalent, which hinders the ability of evaluators to find what is causing accessibility issues. Further, there is also a reported tendency for applications to be designed and developed without accessibility in mind, with the intention of retrofitting applications afterwards. This, combined with the problems associated with increased interactivity for users, seems to be causing the web to become even less accessible than it was previously.

4.4.1 Reporting Accessibility Problems

Interviewees were asked what information they provide to developers and commissioners about the accessibility of their websites. The following is a summary of their responses.

Very few evaluation teams provide reports in the form of lists of violations that occur on a given page. In general, it was only those that followed the strategic testing strategy of evaluations that provide that type of detail. When asked why this was, the primary reason given by interviewees was that usually the violations are not meaningful to the developers and commissioners for them to do a proper cost-benefit analysis regarding the time it will require fixing the violations.

In the majority of cases, evaluators provide a high-level report document regarding the types of problems found on pages, without an indication of the number of instances on any given page. When asked about this approach in reporting, it was usually the case that evaluators were trying to achieve two different things with these reports.

First, the interviewees stated that they were attempting to avoid duplication of information about violations that are common between pages. Most evaluators following the template-centric sampling strategy and user-centric sampling strategy commented that because the errors will show up on several pages, reporting each individually “does not make sense”.

Secondly, this approach gives the evaluators an opportunity to cite examples of problems and suggest possible solutions. In the context of this type of report, the evaluators often
frame the problems in terms the impact on users, or the potential impact on the business in order to encourage fixing the violations detected.

4.4.2 Maintenance of Accessibility

Unfortunately, interest in maintenance of accessibility in organizations, either in terms of process or in terms of iterative evolution of web application designs, is very low. The majority of evaluators stated that after initial reports have been sent to the commissioning organization there is little or no contact regarding the results of the evaluations.

When asked if organizations return to have their websites evaluated again after fixing the issues, the answers were unanimous that this was a very rare occurrence.

4.5 Discussion and Conclusions

These results indicate that there is currently a gap in how tools support accessibility evaluators and the reality of practice of accessibility evaluators. In this section, the requirements and implications for design are presented for new tools in supporting accessibility evaluation.

The statements from evaluators regarding AATTs are particularly important. The current tools do not provide adequate information to the evaluator about what the tools test or what the problems are on a webpage. These statements are supported by the fact that so few evaluators report using AATTs. If new AATTs are to be widely adopted they must provide more information about what is being tested. It is not sufficient for a tool to simply “dump out” a set of problems to an evaluator. It must be possible for the evaluator to query what tests are being performed, and engage in a dialogue with the tool regarding how decisions are made by a tool. All of this information must be presented in the language of the evaluators and frame the results in terms of how the problems are likely to impact the user, and in terms of repair of the issue.

Further to this, most evaluators feel that they bring more knowledge and experience to an evaluation than can be captured in an AATT. When new automated tools are developed, they must be created in such a way that the evaluator can contribute their knowledge into the system in order to eliminate false positives or false negatives. For example, assume that an AATT returns a list of alternative text warnings, such as having an empty (alt=”) text for an image. Many will be decorative images not needing an alternative text, thus creating a false positive. However, in the first instance of such a test, the evaluator must manually check all warnings. In current AATTs, each time the automated test is run, all warnings are produced, and as a result each error may need to be checked by an evaluator each time. In a future AATT one could imagine a case where the evaluator could annotate the information about which warnings are real and which are false positives for propagation to the next time the tests are run. In this way, such manual checking would only need to be done the first time after elements on web pages change.

In terms of manual testing, there are a large number of evaluators who are relying on multiple tools in order to get the complete set of features needed to undertake evaluations. The use of different toolbars, browsers and development tools (e.g. Firebug) in an ad hoc way makes it very difficult to integrate different tests and related reports together. A more
unified approach, or at least a unified view of these different tools through a common interface, is something that would likely be welcomed by evaluators.

Web accessibility evaluators are in need of tools that will help them manage the pages they have been asked to test and what tests have been undertaken on those pages. These features must be flexible enough to accommodate different strategies that evaluators have when undertaking an evaluation. For example, in some cases evaluators will take the traditional approach of performing all tests on one page, and that is one strategy that should continue to be supported. However, the tools should also support the approach of applying one test (e.g. checking for alternative texts on images) to all pages in sequence. Tools that help in these seemingly mundane and tedious tasks will allow evaluators who are working in opportunistic and/or scrambled styles of evaluation to move their practices towards more strategic approaches. Hopefully, this will reduce the overall potential for missing problems in a given set of pages and increase the reliability of an individual evaluator.

The comments regarding the lack of training that many evaluators have, in conjunction with the rapidly changing technology environment, show a need to provide more structured support and help during evaluation sessions through tools. If it is the case that evaluators are reluctant to engage in formal training, and yet there are still issues that evaluators do not understand, then tools that providing comprehensive guidance and structured dialogues for repair would be of value.

It is absolutely essential that any future tools that are produced be able to generate a variety of different formats of reports. Reports that only contain lists of violations found in tests are not of interest to the majority of evaluators or their clients. While it is important to still provide this feature in situations where an organization must have a complete audit of their web application/websites against a set of guidelines, it is equally important that tools support annotation of problems by evaluators. Further, having the ability to collect together similar problems between pages, and then annotate those problems with examples or solutions, is a feature that would be received well by the evaluator community.
5 References


Appendix A: Interview schedule for website commissioners

Name of Staff Member: .......................... Organisation..........................
Date: ...... / ...... / 2011  Participant ID: ..........

About your organization

1. What field/sector is your organization in? (e.g. banking, education, ...)

2. (If not obvious) Is your organisation in the private or public sector?
   a. Private
   b. Public
   c. Third sector (charities, NGOs etc.)

3. What is the size of the organisation?
   a. Small (49 or less employees)
   b. Medium (50 to 250 employees)
   c. Large (more than 250 employees)

4. Is web development typically done in house or outsourced?
   a. In house (get more detail – what is done, how big is the team, what expertise do they have ...)
   b. Outsourced (get more detail – what is done, why, how)
   c. Mixture of both (get more detail – what is done in house, what is outsourced, why)

5. Does the organization have any formal or informal policy or expectation about ensuring that the website should be easy to use by disabled and older people?

About the website

6. Is the website that you are responsible for:
   a. The primary public website for the organization
   b. A subsidiary public website
   c. An internal website, intranet
   d. Other, please explain
7. How would you characterise the purpose of the website?

8. What is the intended audience for the website?

9. Does that audience include disabled and/or older people? [If yes, are they a small, medium, large part of the audience?]

10. Does the website use a Content Management System (CMS) (e.g. something like Droopal, a system where you enter the content and it formats it for you)? If yes, which one?

11. Does the website have any of the following characteristics: (for each, answer yes, no, don’t know)
   a. Content that refreshes automatically (e.g. tickers)
   b. media player and multimedia content
   c. interactive menus (e.g. drop down, expanding menus)
   d. comments from visitors
   e. Flash content
   f. social networking feeds (e.g. Twitter, Facebook feeds)
   g. sharing links for social networking (e.g. Facebook like button or retweet button)
   h. blog or wiki
   i. e-commerce
   j. If yes to i, do you use any external payment system (e.g. PayPal, Verified by Visa)

12. Approximately how many pages is your website? [don’t worry if they don’t know]

13. Do you have information about how many hits/visitors you get to the website? If so, can you give me just an approximate idea of numbers? [don’t worry if they don’t know]
Requirements for web developers and web commissioners in ubiquitous Web 2.0 design and development

I2Web project (Grant no.: 257623)

About you:

14. What is your position within your organisation?

15. Is commissioning/looking after the website a major part of your job description (or one of many things that you do)?

16. What aspect of the website are you responsible for? (Might be everything, just providing content, etc.)

17. What does the term “web accessibility” mean to you?

[“correct” answer is making the web usable to disabled and older people – give your assessment of whether you think they understand that]

18. What does the term “web usability” mean to you?

[“correct” answer is making the web easy to use for target audiences – might be mainstream, particular target audience - give your assessment of whether you think they understand that]

Depending on how they answer, explain what web accessibility and web usability mean

19. Do you have any knowledge and/or training in web technologies?
   a. Yes
   b. No
   If Yes, what kind of knowledge/training?

20. Do you have any knowledge/training in web usability?
   a. Yes
   b. No
If Yes, what kind of knowledge/training?

21. Do you have any knowledge/training in making the web accessible for disabled and older people?
   Yes – get more detail, what knowledge, training, where, was there a qualification?
   No – would they be interested?

Your Web Quality Processes

[In this section, use the appropriate wording depending on whether they do the quality processes inhouse or rely on external agencies; please find out what they do themselves and what they rely on contractors for]

REMINDER: Put in answer from Q4

a) In-house ☐  b) Outsourced ☐  c) Mixture of a & b ☐

Put in further details of who does what:

22. Do you/your contractor try to engage with the “users” (check whether they call them users/customers/visitors, use their term) of the website in planning how to develop it?
   a. If so, how?

23. Do you/your contractor do internal reviews of designs before they are released to the “users”?
   a. If so, how?

24. Do you/your contractor assess the website for usability?
   a. If so, what do you/the contractors do? (e.g. expert review, testing with users)

25. Then, how do you/the contractor track that the findings of such assessments are carried through into the website?
26. And how do you ensure that **usability** is maintained over time (e.g. if the website is updated after it is launched)?

27. Do you/your contractor assess the website for **accessibility** for disabled and older people?
If so, what do you/the contractor do? (e.g. automatic testing against WCAG, manual/expert testing against WCAG, testing with disabled/older users)

28. (If **not** specified in Q27) Does that include testing for conformance to the WCAG or other guidelines?
If other guidelines, which guidelines?

29. How do you/your contractor track that the findings of such assessments are carried through into the website?

30. And how do you ensure that **accessibility** is maintained over time (e.g. if website is updated after being launched)?

Thank them for their time. Ask whether they would like a report on the results of this study.
Appendix B: Survey of Web Developer Practices

This survey is part of the European-funded i2Web Project (see http://i2web.eu), which is investigating ways to develop Web2.0 applications that are more accessible to a wide range of users. The purpose of this survey is to gain an understanding of the type of Web2.0 websites and applications that web developers are currently involved in, as well as the tools and technologies they use to create them. The survey should take around 10 minutes to complete.

Please note that you need to be 18 years or older to complete this survey. Your survey responses will be strictly confidential and data from this research will be reported only in the aggregate.

The first 50 developers who complete the survey will receive a online gift voucher worth 10 euros (or local equivalent) at Amazon or any other online store. To receive the voucher, you will need to provide your name and email address at the end of the questionnaire, but that information will not be used for any other purpose.

N.B: For some of the questions, we have tried to provide an exhaustive list of options. However, if we have missed something, please select the option labelled “Other” and enter it in the text box below. If you have any queries, or encounter any problems with the survey, please do not hesitate to contact me, David Swallow, at: david.swallow@york.ac.uk Thank you very much for your time and support.

Please start with the survey now by clicking on the Continue button below.

1. What is your job title?

2. What type of organisation do you work for?
   1. Large company or organisation (250 employees or more)
   2. Small company or organisation (less than 250 employees)
   3. Charity, educational institution, library, museum, or non-profit organisation
   4. Independent contractor or freelancer
   5. Volunteer, student or hobbyist
   6. Other (please specify)

3. How many years have you worked in the field of web development?
   1. Less than 1 year
   2. 1 - 2 years
   3. 3 - 5 years
   4. 6 - 8 years
   5. 9 - 10 years
   6. More than 10 years
4. What is the general nature of the websites you are involved in (please select all that apply)?

1. Informational websites (these are largely text-based, with the occasional picture, possibly a feedback form)
2. E-commerce websites (these allow users to browse and/or buy products online)
3. Community websites (these allow users to interact with other users, via discussion boards, forums, chat, private messaging etc.)
4. Productivity websites (these provide interactive end-user services, such as currency conversion, driving directions, search engines etc.)
5. Internal websites / intranets (these are private websites used internally by employees of a company or organisation)
6. Rich Internet Applications / Web 2.0 Applications (these are complex web applications that have many of the characteristics of desktop application software)
7. Other (please specify)

5. What is the general size of the websites you are involved in?

1. Very Small (1 - 5 pages)
2. Small (6 - 50 pages)
3. Medium (51 - 100 pages)
4. Large (100 - 1000 pages)
5. Very large (Over 1000 pages)

6. Which of the following features are included in the websites you are involved in (please select all that apply)?

1. Blogs
2. Community features (e.g. discussion boards, forums etc.)
3. Connections with social networking (e.g. Facebook, twitter etc.)
4. Feedback forms
5. Image maps
6. Images
7. Interactivity (polls, quizzes etc.)
8. Mashups with web services (e.g. Google Maps, Wikipedia etc.)
9. Messaging (e.g. private messaging, chat, email etc.)
10. Multimedia (e.g. videos, animations, podcasts etc.)
11. Tables
12. Text
13. User feedback (e.g. comments, ratings, reviews etc.)
14. Web feeds (e.g. RSS, Atom)
15. Wikis
16. Other (please specify)

7. Which, if any, of the following Integrated Development Environments (IDEs) do you use to develop websites (please select all that apply)?

1. I do not use an IDE
2. ActiveState Komodo
3. Adobe Dreamweaver
4. Adobe Flash Builder
5. Adobe Flex
6. Aptana Studio
7. Code Crusader
8. Coda
9. Eclipse
10. Evrsoft FirstPage
11. FlashDevelop
12. HTML-Kit
13. IBM Rational Application Developer
14. JetBrains Intellij
15. JetBrains WebStorm
16. Microsoft Expression Studio
17. Microsoft Visual Studio
18. Netbeans
19. Open Dialect
20. Quanta Plus
21. RapidWeaver
22. TopStyle
23. uCMS
24. WinDev
25. Other (please specify)

8. Which, if any, of the following text editors do you use to develop websites?
   1. I do not use a text editor
   2. BBEdit
   3. Bluefish
   4. BlueGriffon
   5. CoffeeCup
   6. Crimson Editor
   7. Emacs
   8. EmEditor
   9. Freeway
   10. Geany
   11. HTMLPad
   12. iWeb
   13. jEdit
   14. KompoZer
   15. Notepad++
   16. OpenBEXI
   17. PSPad
   18. SciTE
   19. SlickEdit
   20. TextMate
9. The following are a list of typical features of IDEs and text editors. For each one, please indicate whether or not you use it, and in which environment(s). If you use a feature in more than one environment, then for that feature please tick all the environments that apply.

<table>
<thead>
<tr>
<th>Feature</th>
<th>IDE</th>
<th>Text editor</th>
<th>Other application</th>
<th>Do not use</th>
</tr>
</thead>
<tbody>
<tr>
<td>Automated accessibility tests</td>
<td>❏</td>
<td>❏</td>
<td>❏</td>
<td>❏</td>
</tr>
<tr>
<td>Code completion</td>
<td>❏</td>
<td>❏</td>
<td>❏</td>
<td>❏</td>
</tr>
<tr>
<td>Code debugging</td>
<td>❏</td>
<td>❏</td>
<td>❏</td>
<td>❏</td>
</tr>
<tr>
<td>Code generation</td>
<td>❏</td>
<td>❏</td>
<td>❏</td>
<td>❏</td>
</tr>
<tr>
<td>Code navigation (outlines, components)</td>
<td>❏</td>
<td>❏</td>
<td>❏</td>
<td>❏</td>
</tr>
<tr>
<td>Compiler/interpreter</td>
<td>❏</td>
<td>❏</td>
<td>❏</td>
<td>❏</td>
</tr>
<tr>
<td>CSS validation</td>
<td>❏</td>
<td>❏</td>
<td>❏</td>
<td>❏</td>
</tr>
<tr>
<td>Error reporting (e.g. compilation errors)</td>
<td>❏</td>
<td>❏</td>
<td>❏</td>
<td>❏</td>
</tr>
<tr>
<td>File management</td>
<td>❏</td>
<td>❏</td>
<td>❏</td>
<td>❏</td>
</tr>
<tr>
<td>File transfers (FTP upload/download)</td>
<td>❏</td>
<td>❏</td>
<td>❏</td>
<td>❏</td>
</tr>
<tr>
<td>HTML/XML validation</td>
<td>❏</td>
<td>❏</td>
<td>❏</td>
<td>❏</td>
</tr>
<tr>
<td>Model creation and editing</td>
<td>❏</td>
<td>❏</td>
<td>❏</td>
<td>❏</td>
</tr>
<tr>
<td>Source version control</td>
<td>❏</td>
<td>❏</td>
<td>❏</td>
<td>❏</td>
</tr>
<tr>
<td>Unit testing</td>
<td>❏</td>
<td>❏</td>
<td>❏</td>
<td>❏</td>
</tr>
<tr>
<td>Web page preview</td>
<td>❏</td>
<td>❏</td>
<td>❏</td>
<td>❏</td>
</tr>
</tbody>
</table>

10. If there are any features that we have not listed, or if you use a particular combination of environments that cannot easily be expressed above, please provide details below:

11. The following are a list of technologies that can be used to develop websites. For each one, please indicate whether or not you use it, and in which environment(s). If you use a
Technology in more than one environment, then for that technology please tick all the environments that apply.

<table>
<thead>
<tr>
<th>Technology</th>
<th>IDE</th>
<th>Text editor</th>
<th>Other application</th>
<th>Do not use</th>
</tr>
</thead>
<tbody>
<tr>
<td>ActiveX components</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Adobe Flash</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ajax</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Apache</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ASP</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CGI</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ColdFusion</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSP</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DB2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Firebird (RDBMS)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Google Web Toolkit</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Groovy</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HTML/XHTML</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Java</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JavaScript</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lotus Domino</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Microsoft Silverlight</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Microsoft SQL Server</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MySQL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>.NET</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Oracle</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Database/Programming Language</td>
<td>IDE</td>
<td>Text editor</td>
<td>Other application</td>
<td>Do not use</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----</td>
<td>-------------</td>
<td>-------------------</td>
<td>------------</td>
</tr>
<tr>
<td>PostgreSQL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SQLite</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sybase</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Perl</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PHP</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pyjamas</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Python</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Real Studio</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ruby</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Smalltalk</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SSJS (Server-Side JavaScript)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tersus</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Websphere</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

12. If there are any technologies that we have not listed, or if you use a particular combination of environments that cannot easily be expressed above, please provide details below:

13. Do you check whether the websites you are involved in are standards-compliant (i.e. they conform to published standards, such as HTML, CSS etc.)?
   1. Yes
   2. No

14. How do you check whether your websites are standards-compliant (please select all that apply)?
   1. W3C HTML validation service website
   2. W3C CSS validation service website
   3. Validation applications (e.g. HTML Tidy, Total Validator etc.)
   4. In-application support (e.g. through Adobe Dreamweaver)
   5. Browser toolbars and extensions (e.g. Web Developer Toolbar)
   6. Unit testing (e.g. Selenium)
   7. Manually compare websites against the the standards documents
8. Use own judgement
9. Other (please specify)

15. Do you check whether the websites you are involved in are accessible (i.e. they are usable by people of all abilities and disabilities)?
   1. Yes
   2. No

16. How do you test whether the websites you are involved in are accessible (please select all that apply)?
   1. Automated accessibility testing (e.g. CynthiaSays, imergo etc.)
   2. Manual accessibility testing
   3. Expert testing (e.g. by an accessibility expert)
   4. User testing with people with disabilities
   5. User testing with older adults
   6. Testing with assistive technology (e.g. using a screen reader)
   7. Other (please specify)

17. What is your gender?
   1. Male
   2. Female

18. How old are you?
   1. 18-20
   2. 21-30
   3. 31-40
   4. 41-50
   5. 51-60
   6. 61-70
   7. 71-80
   8. Over 80

19. In what country do you work?
   [comprehensive list of countries provided]

20. What educational qualifications have you completed (please select all that apply)?
   1. None
   2. High school
   3. Bachelors degree
   4. Masters degree
   5. Postgraduate diploma
   6. PhD
7. Trade, technical certificate or diploma
8. Other (please specify)

21. In what subject(s) are your educational qualifications (e.g. BSc in Computer Science, PhD in Psychology etc.)

22. If you are one of the first 50 people to complete the survey and would like to receive a 10 euro gift voucher, please provide your email address:

23. Please also provide your full name (many online stores such as Amazon ask for the name of the person to whom the electronic gift voucher is being sent):

24. Please state the online store you would like the voucher from (e.g. Amazon):
Appendix C: Contextual inquiry schedule on “Accessibility support for web developers”

INSTRUCTIONS

- Introduce the i2Web project and its aims to the interviewee.
- Assure the interviewee that his or her responses would be kept confidential with no personal identification information being publicly available in either project deliverables or publication materials.
- Ensure that the interviewee signs a consent form.
- Ask permission for the interview to be audio/video/screen recorded.
- Try to give the interviewee a sense of how long the interview will take at the start of the interview. Interviews should last no longer than two hours.
- Provide the interviewee with a break should the interview run for longer than an hour.
- Seek to obtain as much information as possible regarding the processes involved in the web development experience, particularly with regards to accessibility.
- Some sections of the interview can be conducted in a more traditional interview style. For many of the sections, however, the interviewee is required to carry out a task, with the subsequent questions and prompts reflecting upon that task.
- Whilst there is an order to the interview questions and prompts, feel free to choose topics according to the flow of the interview.

INTRODUCTION

About you

1. What is your job title?
2. Could you describe briefly what your responsibilities are in your role?
3. How many years have you worked in the field of web development (and in this role)?
4. Do you work as part of a team?
   a. If so, how large is the team?

Section: About your skills

1. Do you feel your knowledge of the field is up-to-date?
   a. If so, how do you keep on top of things?
   b. Can you name any resources that you have found particularly helpful?
2. Which areas of web design or development do you feel you have a good understanding of?
3. Which areas of web design or development would you like to know more about?
4. Have you participated in any training in web design or development?
a. If so, what was the subject of the training?
b. If so, what was the origin of the training?

**About your organisation**
1. What type of organisation do you work for?
2. How large is your organisation?
3. What is the main purpose of your organisation?

**About your work**
1. What is the general nature of the websites you are involved in?
2. Could you describe the type of business client you develop websites for?
3. Could you describe the type of audience that you develop websites for?
4. Could you describe the typical features of the websites you are involved in?

**WORKSPACE CONFIGURATION**

[Ask the developer to introduce their workspace]
1. Typical workspace configuration
   a. How many monitors does the developer have and what does he or she use them for (e.g. certain applications on certain monitors)?
   b. What input devices does the developer use and what does he or she use them for (e.g. mouse, trackball, keyboard, graphics tablet etc.)?
   c. What development-related artefacts does the developer keep on or near their workspace (e.g. post-it note reminders, whiteboards, notepads etc.)?

[Ask the developer to introduce their desktop]
2. Typical desktop configuration
   a. What applications does the developer typically have open and what does he or she use them for (e.g. development, design, FTP, chat, internet etc.)?
   b. What webpages does the developer typically have open and what does he or she use them for (e.g. help, information, communication, leisure etc.)

**COMMUNICATION**

[Ask the developer to think about who they communicate with during a typical day, and how]
1. With the customer
   a. How does the developer usually communicate with the customer (e.g. face-to-face, via telephone, email, Skype etc.)?
b. How frequently does the developer communicate with the customer (e.g. throughout the day, on a daily basis, weekly, monthly etc.)?

c. What is the balance of power between the developer and the customer (e.g. who tells who what to do)?

d. How does the developer and the customer establish what is needed from a website and is this formalised in any way?

   i. How much control does the developer have over this?

e. How does the developer report his or her progress to the customer (e.g. meetings, reports, etc.)?

2. With colleagues/team members

   a. How does the developer usually communicate with his or her colleagues/team members (e.g. face-to-face, via telephone, email, Skype etc.)?

   b. How frequently does the developer communicate with his or her colleagues/team members (e.g. throughout the day, on a daily basis, weekly, monthly etc.)?

   c. What is the balance of power between the developer and his or her colleagues/team members (e.g. who tells who what to do)?

   d. How does the developer and his or her colleagues/team members establish what is needed from a website and is this formalised in any way?

      i. How much control does the developer have over this?

   e. How does the developer report his or her progress to his or her colleagues/team members (e.g. meetings, reports, etc.)?

3. With the line manager/boss

   a. How does the developer usually communicate with his or her line manager/boss (e.g. face-to-face, via telephone, email, Skype etc.)?

   b. How frequently does the developer communicate with his or her line manager/boss (e.g. throughout the day, on a daily basis, weekly, monthly etc.)?

   c. What is the balance of power between the developer and his or her line manager/boss (e.g. who tells who what to do)?

   d. How does the developer and his or her line manager/boss establish what is needed from a website and is this formalised in any way?
i. How much control does the developer have over this?
e. How does the developer report his or her progress to his or her line manager / boss (e.g. meetings, reports, etc.)?

HELP AND INFORMATION

[Ask the developer to assume they’ve run into a technical problem. If necessary, provide an example]

1. Sources of help and information
   a. Which people does the developer turn to for help with a particular technical problem (e.g. colleagues, line manager, friends/contacts elsewhere etc.)?
   b. Which offline resources does the developer turn to for help with a particular technical problem (e.g. books, manuals, cheat sheets, company information etc.)?
   c. Which online resources does the developer turn to for help with a particular technical problem (e.g. websites, applications, help files etc.)?

2. Types of help and information
   a. What type of information is the developer usually looking for when faced with a technical problem (e.g. code syntax, design patterns, explanation/instructions, theory/reasoning behind something etc.)?
   b. What is the developer’s familiarity with the information usually being sought (e.g. a reminder of something already known, or exploratory information-gathering)?
   c. How authoritative is the information that the developer usually looks for (e.g. official technical standards, respected authors, blogs, forum posts etc.)

3. Immediacy of help and information
   a. At what stage of development does the developer seek help and information (e.g. immediately, during coding or at a later stage, perhaps following testing)?

PREVIEW & VALIDATION

1. What is your understanding of a ‘standards-compliant’ website?
2. To what extent are the websites you are involved in ‘standards-compliant’?
   a. What standards do the websites you are involved in conform to (e.g. HTML, CSS)?
i. What doctype do you typically use?

b. What, if any, challenges have you faced in ensuring that the websites you are involved are standards-compliant?

[Ask the developer to preview a web page they have created]

3. Previewing

a. What browsers/applications/devices does the developer use to preview a webpage they have created (e.g. Firefox, Internet Explorer, Safari, Chrome, or the preview options of an IDE such as Dreamweaver)?

b. What sort of things does the developer look for when previewing a webpage and in what order (e.g. layout, structure, Gestalt, text-size, colour scheme etc.)?

c. What browser extensions/add-ons/tools/plug-ins does the developer use to assist in previewing webpages (e.g. Web developer toolbar, accessibility toolbar, FireBug, Colour Contrast analyser etc.)?

d. What does the developer do to optimise his or her code (e.g. network monitoring via Firebug, Yslow, image sprites, CSS/JS compilers etc.)?

e. How does the developer preview dynamic pages or sequences of webpages?

f. How does the developer preview third-party components, such as media players or text editors?

[Ask the developer to validate a web page they have created. If the developer does not use a particular validator, provide a number of examples (e.g. W3C HTML, CSS etc.)]

4. Validating

a. What does the developer use to determine whether their website is ‘standards-compliant’ (e.g. W3C validators, a validation application, using the code support of their development application, own knowledge etc.)?

i. How do these help or hinder the developer?

b. How comprehensive is the developer’s validation of their website (e.g. just the homepage, a sample of pages, every page etc.)?

c. How useful does the developer find the information they receive (e.g. does it indicate the problem or cause further confusion)?
d. How **trustworthy** is the information they receive (e.g. is the developer confident that a page is standards-compliant if it validates?)

e. How does the developer **address** the validation errors they receive (e.g. addressing each error line-by-line, addressing the same type of error together, addressing all errors in one go)?

f. How **frequent** is the developer’s validation of their website (e.g. after every change, daily, weekly, monthly etc.)?

g. How does the developer validate **dynamic pages** or **sequences of webpages**?

h. How does the developer validate **third-party components**, such as media players or text editors?

i. What **declarations** of conformance does the user make on his or her websites (e.g. conformance statement, buttons and badges, none etc.)

   i. If conformance statements are used, who is responsible for those?

j. How easily do validation checks fit into the developer’s workflow?

**USERS**

[Prompt the user to consider a typical website, or their last website]

1. How expert does a user have to be to use the website (e.g. does it cater for beginners and experts)?

2. Does the website require instructions?

   a. If so, who wrote the instructions for the website (e.g. client, dedicated copywriter, the developer)?

   b. Who wrote the copy of the website (e.g. client, dedicated copywriter, the developer)?

   c. Does the website provide help in any form (e.g. help pages, pop-ups, contextual help, tooltips etc.)

3. Are there any areas of the website in which users may make an error?

   a. Does the user receive any feedback (e.g. an error message, an audio prompt etc.)?

   b. How does the website provide feedback to users (e.g. alert boxes, audio alerts, in-page text etc.)?

   c. Who wrote the feedback (e.g. client, dedicated copywriter, the developer)?

   d. Does the user get the opportunity to undo their error?
e. Does the user get the opportunity to confirm their actions?

4. Is there anything the developer would add or take away from the website?
   a. Are the pages of the website consistent (e.g. is the navigation, the language, the aesthetics... consistent from page-to-page)?

5. Have you ever attempted to use a website from the perspective of a disabled user?

ACCESSIBILITY

1. What is your understanding of the term ‘web accessibility’?
2. How important is web accessibility?
3. Where does the responsibility lie for making websites accessible?
4. To what extent are the websites you are involved in accessible?
   a. What are the typical accessibility features that you implement?
   b. Do you understand why the websites you are involved in are accessible?
   c. To what extent is accessibility promoted in your organisation?
   d. What factors motivate you to take accessibility into account?
   e. What factors prevent you from taking accessibility into account?

[Ask the developer to test the accessibility of a web page they have created. If the developer does not use a particular checker, provide a number of examples (e.g. imergo, CynthiaSays, WAVE etc.)]

1. Testing
   a. What does the developer use to determine whether their website is accessible (e.g. automatic tools, such as imergo; dedicated applications, such as Watchfire; tools built into their IDE or browser etc.)
      i. How do these help or hinder the developer?
   b. For what user groups does the developer check their website is accessible (e.g. blind, partially-sighted, deaf, physically disabled, dyslexic, older)?
   c. How comprehensive is the developer’s accessibility check of their website (e.g. just the homepage, a sample of pages, every page etc.)?
   d. How useful does the developer find the information they receive (e.g. does it indicate the problem is (down to the line number) or does it cause further confusion)?
i. What information does the developer take from an accessibility check?

ii. How easily are you able to make changes based on the information?

e. How **trustworthy** is the information they receive (e.g. is the developer confident that a page is accessible if it passes the check?)

f. How does the developer **address** the accessibility problems they receive (e.g. addressing each problem line-by-line, addressing the same type of problem together, addressing all problems in one go)?

g. How **frequent** is the developer’s accessibility testing of their website (e.g. after every change, daily, weekly, monthly etc.)?

h. How does the developer check the accessibility of **dynamic pages** or **sequences of webpages**?

i. How does the developer check the accessibility of **third-party components**, such as media players or text editors?

j. What **declarations** of accessibility does the user make on his or her websites (e.g. accessibility statement, buttons and badges, none etc.)

   i. If accessibility statements are used, who is responsible for those?

k. How easily do accessibility checks **fit** into the developer’s workflow?

[Ask the developer to think about how the accessibility of his or her web pages could be improved]

1. Types of accessibility help and information

   a. In what **form** would the developer prefer to receive information about the accessibility of his or her code (e.g. as part of development environments, dedicated application, dedicated website, simulations, visualisations, checklists etc.)?

   b. About what **user groups** would the developer like more information (e.g. blind, partially-sighted, deaf, physically disabled, dyslexic, older)?

   c. What **type** of information would the developer prefer to receive about the accessibility of his or her code (e.g. code syntax, design patterns, explanation/instructions, theory/reasoning behind something etc.)?

d. In what **detail** would the developer like to receive information about the accessibility of his or her code (e.g. the location of the problem, how to solve the problem, the cause of the problem etc.)?

e. To **whom** would the developer like to provide information on the accessibility of his or her code (e.g. clients, colleagues, bosses, general public, no one etc.)?

f. How **comprehensive** would the developer like the information about the accessibility of his or her code to be (e.g. just the homepage, a sample of pages, every page etc.)?

2. **Immediacy of accessibility help and information**

   a. At what **stage of development** would the developer prefer to receive accessibility information (e.g. during planning stages, during coding or once coding has been completed)?

   b. How **frequently** would the developer prefer to receive accessibility information (e.g. after every change, daily, weekly, monthly etc.)?

   c. How **granular** would the developer like accessibility information to be (e.g. element-level – about specific elements, such as an individual table cell, feature-level – about an entire feature, such as a table, page-level – about the entire page)?

   d. How **active or passive** would the developer like to be in receiving accessibility information (e.g. the developer is notified automatically about accessibility information or the developer specifically requests accessibility information)?

3. **Promotion of accessibility**

   e. How would the developer like to **declare** the accessibility of his or her web pages (e.g. accessibility statement, buttons and badges, none etc.)?

**ANY OTHER COMMENTS**

1. What would really help the developer to develop accessible websites in future?

2. What does the developer perceive to be the main problems in the design of accessible websites?

3. Would the developer be interested in taking part in a follow-up interview to discuss potential design solutions? (This may be carried out online).
Appendix D: Web Accessibility Evaluator Interview Schedule

ABOUT YOUR ORGANIZATION

31. What business sector is your organization in? (e.g. banking, education, ...)

32. (If not obvious) Is your organization in the private or public sector?
   a. Private
   b. Public
   c. Third sector
   d. Other (non-profit, joint funded)

33. What is the size of the organization?
   a. Small (49 or less employees)
   b. Medium (50 to 250 employees)
   c. Large (more than 250 employees)

34. Does your organization do accessibility evaluations for in-house applications?

35. Does your organization do accessibility evaluations for external clients?

36. Does your organization participate in any of the national or international web standards groups? (e.g. WAI, Section 508)

ABOUT YOU

37. How were you introduced to accessibility as a topic?

38. How long have you been doing accessibility evaluations?

39. Did you have any formal training in accessibility and how to perform evaluations?
ABOUT YOUR TOOLS

40. Do you perform automated accessibility tests on websites?
(If Yes) What tools do you use to perform those tests?

41. Why do you use those particular tools (e.g. tests performed, company selected etc.)?

42. Do you do expert tests of websites for accessibility problems?
(If Yes) Do you use any tools to help you when doing those tests?

43. Do your tools have any of the following features? (Yes/No/Don’t know for each, I would also ask if No, would you find that useful, how useful would it be to you on a scale from 1 = not at all to 5 = very useful)
   a) ability to choose different guidelines/levels (e.g. 508, WCAG1, WCAG2, different levels of WCAG)
   b) Explanation of which particular checkpoints within a set of guidelines are tested automatically and whether the test is comprehensive
   c) list of automatically detected accessibility problems
   d) explanation of each problem observed
   e) visualization of the experience of a person with a disability
   f) explanation of why a particular code fragment might cause problems
   g) checklist of manual accessibility checks (cannot be automated)
   h) example solutions to observed problems
   i) ability to pinpoint error on the rendered page view
   j) ability to pinpoint error in the DOM
   k) ability to pinpoint error in JavaScript
   l) ability to pinpoint error in portlet/servlet code
   m) ability to run a predefined interaction sequence, checking accessibility at each stage
   n) ability to automatically test all possible states of the application
   o) indication of the severity of a problem
   p) prioritization of accessibility problems

44. In your opinion, what are the most common accessibility problems you find on websites?

45. What resources (paper, online etc.) do you refer to, if any, for information about accessibility requirements and solutions during your testing?
46. Do you do any evaluation with users?

ABOUT YOUR PROCESSES

47. How do you select pages from a website to be tested in an accessibility evaluation?

48. How do you keep track of what pages you have tested within that set?

49. How do you keep track of what tests you have completed on a web page?

50. What information do you collect for the developer/commissioner regarding the accessibility of each page? For the site?

51. How do you communicate that information to the developer/commissioner?

52. Do you do consultations on how to fix accessibility problems based on your test results?

MAINTAINING ACCESSIBILITY

53. Do clients ask you to help in the accessibility maintenance process? If so, how?

54. How do you advise clients to maintain the accessibility of their website over time?

55. If you are asked to test a site again, after changes have been made, do you change anything that you do in your testing?