Accessibility Testing in Agile Software Development

Empowering agile developers with accessibility methods

Nikolai Sverdrup

Thesis submitted for the degree of
Master in Programming and Networks
60 credits

Department of Informatics
Faculty of mathematics and natural sciences

UNIVERSITY OF OSLO

Autumn 2018

Accessibility Testing in Agile Software Development

Empowering agile developers with accessibility methods

Nikolai Sverdrup

Accessibility Testing in Agile Software Development

http://www.duo.uio.no/

Printed: Reprosentralen, University of Oslo

Abstract

This thesis delves into some of the methods that can be used to do accessibility testing in software development. Challenging traditional conventions of working for accessibility in agile development, where slow and resource intensive methods dominate, I investigate several methods that could ease the task of creating accessible software. I will mainly be focusing on web accessibility in tandem with a wider world effort working on the same goal. I have deployed several methods to professional software developers working in many different specialties. There are five methods I deploy and evaluate: A type of glasses that emulate bad eyesight, a software tool that analyzes and reports accessibility faults on websites, Screen-readers used a debugging tool, a method where the testers act out the user experience of imagined disabled users and a dyslectic emulation tool. There is also a smaller evaluation of using traditional WCAG testing. A case study was used to try and get a closer look at real life accessibility testing and aiding in finding out what obstacles accessibility testing need to climb if the testing methods are to be deployed successfully.

By utilizing data from a case study, interviews and surveys, I compile information about how the participants experienced using the methods. There was also data gathered to see how the participants viewed accessibility and how it was handled in development, together with the tests creating a broader view of how accessibility is represented in the attitudes of the developers and their work routines. Overall the results from the trials where positive, with one method being made redundant and concluded not to be useful. From the investigation into accessibility attitudes, there was a tendency to acknowledge that accessibility is an essential aspect of software development, but few put effort into integrating accessibility knowledge and methods into their work process.

Acknowledgements

I would like to thank everyone who contributed and helped me with the thesis. Special thanks to my supervisor Viktoria Stray for all the helpful guidance. I am also grateful to Heidi Mork and Aleksander Bai for all the help and cooperation. A big thanks to UnIKT and Barne-, ungdoms- og familiedirektoratet for supporting the project, as well as the businesses and organizations that helped me in the research.

I would also like to praise my dog Frøya for getting me up in the mornings.

Nikolai Sverdrup

Contents

List of Figures

List of Tables

Part I

Background

Chapter 1

Motivation

Creating software that is accessible for disabled users is an important consideration for an increasingly digital society. In the same vain as creating wheelchair accessible buildings and writing braille on elevator buttons there is a need to take account for disabled users who wish to use software. There is already an increasing focus on making some software more accessible for users with disabilities, made clear by the numerous suggestions passed in to law in many countries. Putting accessibility into practise however still remains a challenge seen by the many web sites that have failed to achieve accessibility (Sánchez-Gordón and Moreno, 2014). The agile approach used by many developers are under scrutiny for not offering enough consideration for usability (Kane, 2003). Lack of incorporating usability, including accessibility methods, in broad agile software dogmas gives it a disadvantage from the get-go. The benefits of having a accessible solution is immediately apparent. Estimates indicate that people living with disabilities in Norway are as high as 25 % , with similar figures in other countries(Molden et al., 2009, p. 6) (Kittelsaa et al., 2015, p. 10). Poorly developed software with regard to accessibility is aggravating to disabled users and can make the product owner miss out on potential users, giving the issue both a social justice and economical motivation. The way accessibility is handled in many agile development processes is a source for less accessible solutions and increased cost. Agile principles highlights principles such as working software and regular testing, however incorporating accessibility throughout an agile process had proven difficult seen in research from Kane (2003) , Zimmermann and Vanderheiden (2008), and can be an expensive endeavor(Nielsen, 2003). studies indicate that usability in Agile is overlooked in development, let alone accessibility(Kane, 2003) (Zimmermann and Vanderheiden, 2008). There is thus a need to find methods that integrate well with agile approaches. Methods and tests that can be utilized often without being resource intensive and costly, so they can fit in with agile principles. Our purpose setting out on this project was thus to improve accessibility in software development. Agile software

development in particular as it growingly looks to become the underpinning set of values and principles to modern software development.

1.1 Research origin

The research conducted in this paper can be seen as a continuation of the work described in the paper "A Cost-benefit Evaluation of accessibility Testing in Agile Software development" by Bai et al. (2016) and A cost-benefit analysis of accessibility testing in agile software development: results from a multiple case study by Bai et al. (2017). The paper investigate the cost benefits of using an assortment of accessibility tools at different stages in development to avoid detecting accessibility flaws in the later stages of a software product in development. The continuation of the research aims to further validate some of the tools used in the report and further expand on the work by testing other methods. The Bai et al. (2016) article highlights a wish for more participants to test with in future research. This shapes a case where we set out to do a more user tested approach to the methods. The tools will also be examined in the context of the usefulness for the developers who could benefit of these methods.

1.2 Research Area

The research domain encompasses the practice of accessibility testing of software made in an agile environment. We look at every phase of a agile process where accessibility might be a issue and examine tools that can help identify those issues. Whenever a software product is under design consideration, in active development or in continual maintenance and support.

1.3 Research questions

The pilot project influences the research questions significantly as I build upon its the work and conclusions. At its core this thesis seeks to validate five accessibility test methods, to see if they work in testing and if they are suitable for agile development. from this the research question:

"What test methods are suitable for agile software development".

I mean the question to be both in terms of using the methods alone and together as a test suite.

Secondly, in order to not just look at the tools and methods in a narrow testing scope, but also to consider them for a broader purpose, I am interested in the surrounding issues and attitudes accessibility testing can face. So a second question will be:

"What are the challenges and opinions regarding accessibility testing in agile development."

1.4 Pronoun clarification for this paper

Throughout the thesis, there are both descriptions of work done by myself and work done by the research group that I was a part of when conducting data-gathering for this paper. I try to reference work done solely by myself by using singular pronoun and work that other group members have partaken in by using plural pronounces when describing the work.

1.5 Thesis structure

There are three main parts.

The background describing why I am researching this topic and defining the concepts that are utilized. There are three chapters with the Motivation chapter describing the origin of this paper, the accessibility chapter describing accessibility and the tools I will test, and the agile chapter describing agile development and how accessible development fits into its philosophy.

Part two describes how I have conducted my research, detailing the case study, the attitude survey and testing the accessibility methods.

Part three describes my findings and also features a discussion and a conclusion chapter.

Chapter 2

Accesibility in software

Accessibility in software is a broad topic to find aiding methods and tools for, because of the many different environments software is developed in. While ideally tools used to check for accessibility should be available for all kinds of different software development environments, we have narrowed the scope of the topic by focusing on the areas concerned with content made accessible with a web browser. The tools we look at reflect this as many are extensions and applications for web browsers. Web browsing is also a very universal and homogeneous experience that will affect a large number of disabled users and there exist a more diverse pool of tools available for testing against web solutions. Also playing into the selection of web-focused methods is the international effort ongoing to make the web universally accessible. The guidelines set for web accessibility is however usually appropriate for any software product (Pressman, 2002, p 380).

The international efforts are to be seen in many countries that have adopted regulations to provide accessibility standards which establish internationally recognized standards and guidelines.

In Norway, universal design of ICT is a legal requirement for both public and private sector. According to Section 18 in the Act relating to equality and a prohibition against discrimination (Equality and Anti-Discrimination Act), ICT-solutions must be universally designed. Further requirements are specified in Regulation on universal design of ICT-systems (FOR-2013-06-21-732). Section 4 of the Regulation obliges all solutions made for the web to be designed at least in accordance with the Web Content Accessibility Guidelines 2.0. Komunal- FOR-2013-06-21-732. Similar regulations as those in Norway are adopted in the US and in the EU.

2.1 Accessible content

Accessible content is content available in such a way that it is arranged to be accommodated for users who have disabilities, by removing barriers that can interfere with the user experience. What constitutes accessibility greatly fluctuates in context with the requirements set by a product and the requisites of the user. Some users are dependent on assistive technologies that can both be native to the system they are operating on or come in an add-on capacity. A sight-impaired person might be dependant on a screen reader or a user could be disbarred from using a physical keyboard or mouse. These ranges of different disabilities must be considered when one makes up for who will be using a product and how it can be made accessible (Bergman and Johnson, 1995).

2.1.1 Accessibility within usability

Accessibility and usability are two closely related terms that often can be used interchangeably. Accessibility can be seen as a subset of usability, where accessibility focuses on the particular details of things that might discriminate against a person with a disability, and usability can be used in more general terms to relate to the user experience of every user. There is a high degree of overlap between the terms and achieving accessibility also usually entails achieving usability. Many of the methods one can utilize in making an accessible product are techniques used for usability only further specialized for the needs of users with disabilities (Consortium).

2.1.2 Universal design

In Norway, the term "Universal design" ("Universal utforming" in Norwegian) is often used in context with a broad set of cases dealing with accessibility. The term will be occasionally used in this paper as it has been used when talking with participants in this project as many are more familiar with this terminology. I have used it strictly to refer to accessibility topics in the software domain.

2.2 Accessibility testing

Accessibility testing is any testing effort applied to determine to what degree a product is accessible to users with disabilities. The wide variety of disabilities makes testing a software product for features that might compromise its accessibility a complex problem to solve. Given the large number of different accessibility issues that might occur it

would be inefficient to rely solely on an human, unsupported by any aid or tool to undergo such a task. Tools and guides are therefore often included in an attempt at accessibility testing.

Testing for accessibility is much more then checking assertions and measuring if values are within limits, it is often very distinctive from other common activities of testing computer programs, such as unit testing of source code. One characteristic of accessibility testing is the very broad and diverse set of cases one has to account for in tests. From display issues to input compatibility, and how they affect software in ways that are difficult for computers to detect and often requires some form of human judgment to decide if there exist accessibility bugs.

Due to the wide variety of disabilities and the various issues associated with those disabilities, there is a need for a variety of testing methods to cover most needs. Accessibility testing also sometimes requires an elevated level of knowledge about accessibility issues from the testers in order to facilitate a correct understanding of the issues if they are to be discovered and it is generally an advantage in all form of accessibility testing, however, knowledge about accessibility is not generally thought in computer science educations.

2.3 WCAG

WCAG is a set of guidelines, created by the World Wide Web Consortium(W3C), to outline the requirements that are needed to achieve web accessibility. In its second edition, it gives a testable list of objective success criteria, each made to be easily tested against regardless of framework or technology in use. With three degrees of conformance, the guidelines can be fitted to be used different levels of content type, developer skills or design aesthetics, giving developers some autonomy in how many requirements have to be implemented (Reid and Snow-Weaver, 2008).

The guidelines are grouped into four key principle areas, together providing 12 guidelines to guide developers towards accessible content. Each guideline is further fleshed out with testable success criteria.

To be able to pass WCAGs standard the content needs to be perceivable for a user so that no content is made difficult to access or view. Content needs to be visualized clearly by such means as avoiding bad contrasts, making non-text content available with text alternatives, etc. Regarding content interactivity, it outlines how content should be operable through a keyboard and rules to follow ensuring a user would not be hindered in navigating a website. WCAG further defines what encompasses understandable content and gives instructions in how to make a robust solution (Consortium et al., 2008).

2.4 Description of testing methods to be used in this project

Ideally, the methods I will examine will give a wide coverage of common disabilities while not remain resource intensive, enabling the testing to be commenced early and regularly through the development. A challenge to accessibility testing is the wide range of possible user hindrances, and testing for all cases and permutations would be impossible to achieve.

In this thesis, we take a look at several testing tools and methods to see if we can make accessibility testing more agile friendly. Except for WCAG which serves as an example of a more traditional approach, the methods are chosen for their possible agile characteristics and that they together represent a wide test coverage for accessibility related issues in software. There where some specialty areas we felt we wanted to prioritize in that we made sure to have test methods covering visual and cognitive disabilities, as users with visual disabilities are a large group that are afflicted by bad accessibility in software and users with cognitive disabilities, because it is difficult to do testing geared towards this group. Before we began there was also a pilot project conducted using several similar methods to the one we use in this paper that further helped in narrow down what methods we wanted to use.

Test coverage of methods used In order to be a suitable alternative to WCAG the tests needs to cover a high amount of the common WCAG checks.

By only counting double-A rated or lower paragraphs and eliminating those dealing with less common web attributes such as SMIL, PDF, Flash, Silverlight, and only focusing on more commonly featured attributes that you would expect to find on most websites such as the paragraphs dealing with HTML, CSS, Aria, client, and server-side scripting, then there is left 36 paragraphs and the testing methods used in this paper(excluding WCAG) would uncover 83% of those. Not covered are these WCAG paragraphs

- 1.2.5 audio description of video media
- 1.4.4 re-sizing text without loss of function
- 2.2.2 Pause, stop or hide moving, blinking, scrolling text or auto-updates.
- 2.3.1 No more than three flashes on a page per second
- 3.3.3 Lack of error suggestions in input fields
- 3.3.4 error prevention in legal and financial data

These paragraphs are very difficult to detect by any tool, however 2.2.2, 2.3.1 and 3.3.4 deals with uncommon features. Some could potentially be uncovered with Persona testing(described in 2.8), but it would depend on using the right persona.

2.5 WCAG 2.0

WCAG is widely used as a standard tool to check for accessibility, and we suspected from previous use and testimonials that it is not suitable as an all-encompassing accessibility test method, and we will confirm those suspicions later in the result section. The WCAG standard, now in its second edition, can directly be applied as a tool for use in testing. The success criteria specified in WCAG are made to facilitate testing, as the conditions described are made with measures one can verify (Reid and Snow-Weaver, 2008). WCAG is presented as a set of specifications categorized into sections and subsections. It is categorized into 4 broad principles, with together feature 12 guidelines, containing across all the categories 61 numbered paragraphs of success criteria. Each criterion has its own detailed page, specifying the intent, examples, techniques, key terms and related resources associated with the paragraph.

All the different levels and categories of WCAG are constructed to work as one comprehensive guide as stated: "All of these layers of guidance (principles, guidelines, success criteria, and sufficient and advisory techniques) work together to provide guidance on how to make content more accessible." (Consortium et al., 2008)

Although the World Wide Web Consortium(W3C) website features a profusion of resources made to help in integrating accessibility in a product, such as policy guidelines, planning activities, tutorials and evaluation resources, at its core WCAG features its paragraphs of success criteria as its primary guide to verify accessibility. The success criteria are accompanied by a set of techniques meant to be used for aiding in the implementation of the guidelines. The techniques and success criteria will altogether total 379 different pages, each with a description, examples, additional resources, test procedures and expected test results. In addition there are 79 different points of common failures, that can help developers identify faults.

Practical testing When using WCAG for testing one has to select what level of conformance testing is going to be abided by. Testing relying solely upon the WCAG standards could conceptually be approached in different ways, but simplistically applied the tester can go through the list and check every point, and then compares the software with the standards described in WCAG. Some criteria can be quickly be skipped if

they are not relevant, such as criteria describing video content if the software does not contain video media. The techniques and other additional aids provided by WCAG can be incorporated into the testing.

2.6 Cambridge Simulation Glasses

A tester wears blurred glasses as seen demonstrated in 2.1, while interacting with the interface or object that is under examination. For our testing, we used the Cambridge Simulation Glasses(CSG). The CSGs are thin enough to stack several pieces of glasses in sequential layers, enabling the tester to be in control of the degree of reduced vision. Before using the glasses the tester performs a quick eyesight evaluation using a Snellen chart, that comes with the glasses or can be printed out. The chart provides how many glasses the tester should wear to simulate a 95 % coverage of the general population. While wearing the glasses the tester interacts with whatever is to be evaluated for accessibility, and any possible shortcomings should be made readily apparent for the tester as they might struggle to use the solution.

2.7 Screen reader

A screen reader is a software tool mainly intended for use by people suffering from mildly impaired vision to complete blindness. It reads whatever the user is currently highlighting, and when used in a web browser interprets the site by parsing the HTML code and communicates it back to the user, usually by having the computer read it out audibly but the content can also be displayed using a braille display. Text, graphical items, and HTML elements are some of the components that a screen reader can interpret on a website. When operating on an application aided by a screen reader, the keyboard is usually used for navigation as this can make the current highlighting more apparent and are easier to navigate with if the user has visual impairments. Some screen readers offer more advanced functionalists and different flavors, which can lead to different behavior and interpretations. For our use in testing, we have used NVDA screen reader and VoiceOver, both free to use. They are both similar in functionality, however NVDA is made for the Windows operation system and VoiceOver is made for IOS (Bigham et al., 2008).

For testing purposes, a screen reader can be convenient as a tool to check that a website is can be interpreted by a screen reader, and secondly, it proves that a website is easily

FIGURE 2.1: Cambridge Simulation Glasses

navigable with a keyboard. An important functionality to use is the ability to the interpreted text relayed in text format, as the audible interpretation can be difficult to understand without experience.

2.8 Personas

Personas come from an interaction design technique where a solution can be examined by a designer or developer pretending to be an imagined user. The character or "Persona" to be used in the role play, features personality traits and a backstory to facilitate a process where someone can act as the Persona.

Personas are similar to user testing with people that have disabilities. Finding and testing a product with actual disabled users can yield very accurate results, and some

usability companies offer specially trained accessibility testers, with people who themselves can possess the disability they are testing for. This is a very valuable process and can be a good way to verify a products accessibility, but can consume a lot of time and resources. Eklund and Levingston (2008) argues that this can lead to accessibility testing becoming a validation exercise, where accessibility is reviewed once or twice in the development cycle, and often at the end where change is likely going to be more expensive. The cost of user testing also makes it difficult to incorporate it into regular agile practices, even when user consultation is encouraged (Eklund and Levingston, 2008) Personas could potentially help alleviate some of these concerns since it can open the possibility for testing with a users mindset in mind while being less resource intensive (Schulz and Fuglerud, 2012, p. 146).

FIGURE 2.2: Persona character example (Till Halbach)

Without the disability traits the exercise features made up characters with characteristics that represents a typical user, and when used with a multitude of different Personas acts as a way of understanding the different needs and viewpoints of the user base. This concept can be easily augmented to fit in with accessibility testing by adding extra disability traits to Personas. As with regular Personas where the characters are

preferably made up by using data from real users, personas with disabilities should ideally be constructed by analyzing characteristics of disabled product users. Collection of the data can be approached in a multitude of methods, such as with interview and surveys, and can be a one-off exercise after completion as the personas can be retained as long as they represent the users in an adequate manner. Holding on to the personas for a long time also further increases its usefulness as the people using them gets more familiar with the characteristics of the personas, making it more approachable to immerse oneself in a persona and help to build sympathy for the character (Schulz and Fuglerud, 2012, 147-148). An example of a Persona character can be seen in figure 2.2

As Pruitt and Grudin (2003) conclude, testing for challenges relating too mental or intellectual disabilities are difficult to quantify. Where other testing tools can be highly automated and many disabilities can be adequately covered with software tools, there are issues that make creating tools that can cover mental or intellectual disabilities a larger challenge due to their nature. Human judgment and participation are necessary and Personas can be a tool to carry this responsibility. Personas can take on the characteristics of any disability and thus fits very well in this void as a method testers can use to make better solutions.

2.9 Dyslexia simulation

The Dyslexia simulator is a small and simple browser extension that tries to simulate how a dyslectic user or someone with some other reading disorder might experience a web site. Dyslexia is a common disability, and the condition can severely hamper a users ability to comprehend a website. Dyslexia impedes the ability to read, and the manner in how software is presented can significantly affect the difficulty of interaction for a dyslectic user. This is especially apparent in sites that feature a lot of text or require the user to write (W3C). The tool tests a website by shuffling the letters in words around, making it difficult to read. An example of how the text is scrambled can be seen in figure 2.3. This does not perfectly simulate what a user with dyslexia experiences, but instead gives a good indication of the same areas a dyslexic person might find challenging. When testing a solution the tester operates the interface as normal with the dyslexia software running, and if there are areas that become difficult to perceive then the tester can use that as an indication for something that might become difficult for someone with reading difficulties to perceive. The Dyslexia simulation tries to highlight issues related to a neurological disability which is challenging to test for since it is difficult to make software that can detect areas affected by such disabilities and it is difficult for a tester to understand what the disabled user might experience.

There are several WCAG paragraphs detailing the issues related to reading disabilities and steps one can use to help in the matter. Most of the paragraphs where I would say the dyslexia simulation will substitute using WCAG are paragraphs at the AAA level.

Dyslexia simulator version and work For this thesis we used an extension for the Google Chrome web browser that I made. The dyslexia simulator in the pilot study was a script the user had to run for each use. I developed a web extension that is available on the chrome extension web store and can then more easily be installed and used.

hTe bojecitve of tihs etcnhiqeu is to neuser taht the ettx of hte
Wbe pgae is not dififcult to reda. Usesr iwht idasbiltieis that
maek it dfifiuclt to dcedoe worsd and senetcnse are likley to ahev
torbule raeidgn adn unedsrtandign ocmlpex txet. If hte txet odes
not erqiuer erdaign abliiyt mroe davnacde tahn the olwre escnodayr
eudcation elevl, no suplpeemnts or latreantvie vreisnos rae eneedd.
In odrre to rdeuce teh cmolpextiy of teh txet: eDvelpo a snigle
otpic or usbtpoci epr apargrpah. Use the ismlpest estnneec ofmrs
cosnsietnt iwth teh prupsoe of the cotnetn. Fro eaxpmel, teh
ispmelts sneetcne-ofmr ofr nElgish cosnsits of Sbuejct-eVrb-Ojbetc,
as in oJnh iht teh abll or Teh Web iste ocfnrosm to WACG 2.0.
Use sentecnse htta are no olgnre htna teh ytpiacl accpeetd elngth
fro seocndray eudaction. (Ntoe: In Egnilsh that is 25 wrosd.)
oCsndier diivding logner sentecnes into tow. Use setnnecse that
cnotain no omer tahn two conujncitno.Indciaet oligacl reltaionshpis
ebwteen prhsase, esntenecs, aparrgaphs, or estcions of the tetx.
vAodi prfoesisonla ajrogn, salgn, and other etrsm with a
sepcaiilezd meaingn that amy ont be lcear to poeple.

FIGURE 2.3: Dyslexia simulation

2.10 SiteImprove Accesibility Checker

SiteImprove is a browser extension that can analyze a web-page for breaches of most WCAG paragraphs. When activated the extension will automatically analyze the currently opened web-page for noncompliance with either the A, AA, or AAA level of the WCAG technical standard. An example in figure 2.4. It distinguishes between the different categories of WCAG standards and organizes them into groups. Each instance of error can give a direct link to the WCAG manual for a more detailed explanation of why the error exists and comes with suggestions in how one can get in compliance with WCAG. Other notable features are the ability to highlight where the error exists on the site itself and in the public page source code by highlighting in the browsers developer tools. The tool is the most automatic of the ones we have tested and requires little

human judgment of determining if there exist an accessibility bug or not. The tools are also probably one of the more content heavy that we are testing. SiteImprove is similar to another popular tool named WAVE and there are several sites that provide similar services. How the analytic capability of these tools rank against each other is unknown to me, however, SiteImprove seemed to be the best candidate for this type of tool as it is actively supported and seem to be the most feature-rich of the tools in the category. There was also a preliminary pilot done before we started testing these tools where both WAVE and Siteimprove was tested, and SiteImprove was selected as the preferred tool based on feedback from tests.

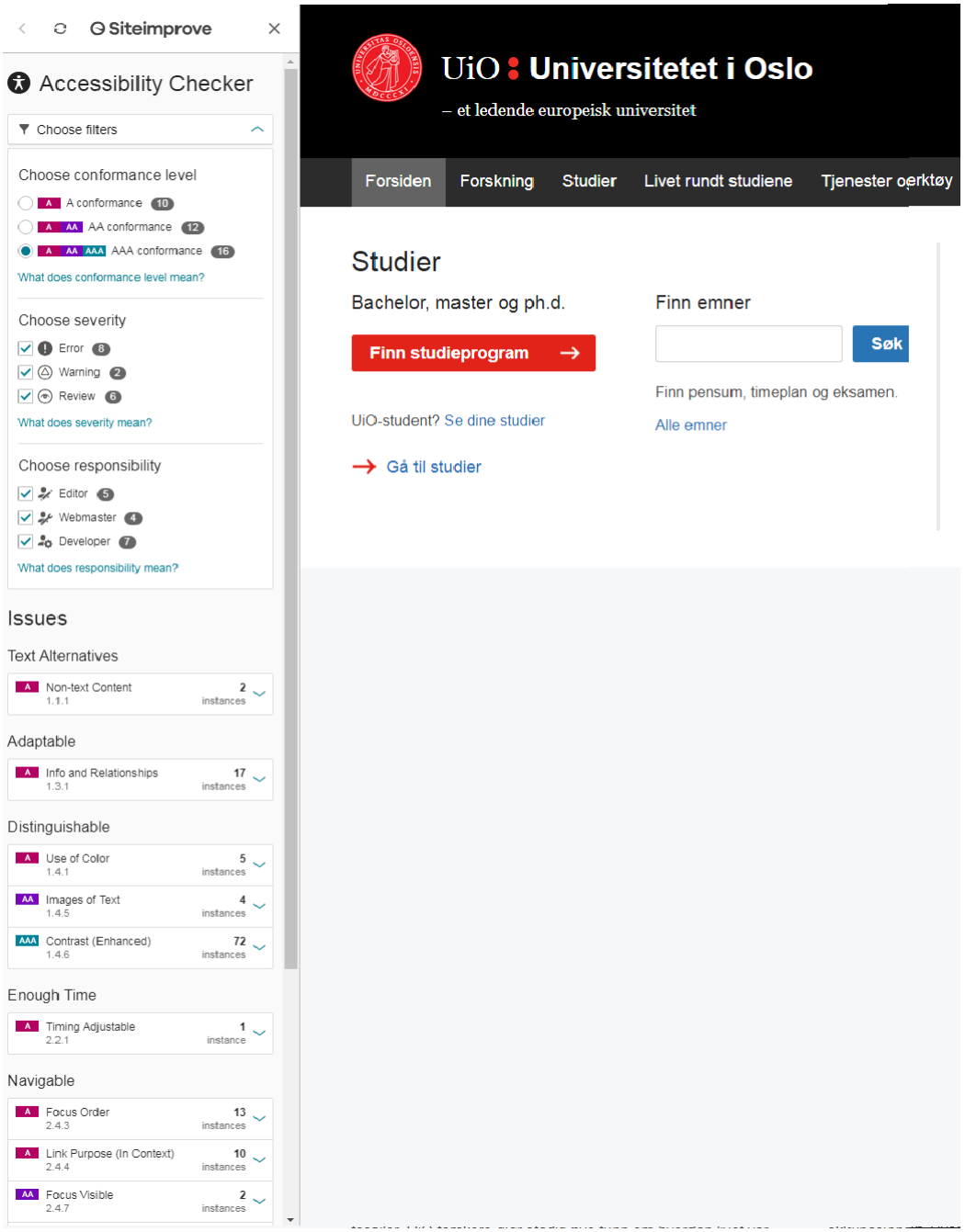

FIGURE 2.4: SiteImprove

2.11 The cost of accessibility

Failing to make a product accessible can be a costly affair, as it will increase the probability of having to make late and expensive changes in the development process, and

further leading to prolonged development time. A product where the user cannot achieve their goal through the usual procedure, will also lead to a reduction in users and increase the need for additional support and assistance to help guide the users, such as help-desk services and on-site support (Bias and Mayhew, 2005, p. 18-33). For those that do involve usability in their development process will find that the cost for testing procedures can take its fair share out of the development budget, and especially for accessibility testing where the requirements can be very specific (Bai et al., 2016).

Investing in usability testing early and throughout the development process can yield a substantial return on investment. Detecting mistakes early lowers the cost of fixing the error and avoids big time-consuming problems to accumulate at the end of the development process. Accessibility testing, however, has the unfortunate fate of often being conducted at the very end of development (Sánchez-Gordón and Moreno, 2014). Reduction in the amount of work needing to be done in the maintenance phase can also be a substantial benefit, as a significant portion of the life-cycle expenditure of a software product can congregate in this later stage (Bias and Mayhew, 2005).

2.12 Background and roles of the participants

Both in the case study and during the testing of accessibility methods we encountered participants who identified with different work roles. The roles dictate what their main work assignments look like, what they focus on and significantly affect how accessibility can have an impact on their work, as the context of where accessibility shows up changes from different types of work. Following is a brief overview of the roles I encountered during my research with a description of what they do. It is mostly based on information gathered during the case study.

UX designer UX designers work on how a software product looks and how the users interact with the software. They take the requirements and develop sketches the developers can make into software. The designers I met often communicated with the end users to establish problems they had with current design and to gauge how the end users wanted to interact with the software.

They are often the first in line to encounter accessibility flaws. Many errors can be avoided if the designer possesses and applies knowledge about accessibility. They can check colors for bad contrasts, make the visuals easy to understand and so on. Some of the tools we test in this paper are also fully compatible with design sketches so the designers can already perform accessibility testing at a very early stage in development.

While designers often take responsibility for a lot of accessibility-related issues there are many aspects of accessibility that are outside of their control. When we spoke with designers they conveyed a concern that they felt they were often held solely responsible for accessibility issues and they expressed a desire for other work roles to take a greater part in creating accessible products.

Programmer The programmers create and maintain the software parts of the product. We have participants who work on many different aspects of software, both in front-end type of work and back end. Programmers do a great deal of the work towards making a software product accessible. For websites especially as they must take care to write the website in such a manner so that it is compatible with screen reader and keyboard. This will encompass such things as making sure that keyboard navigation happens in a logical order, making everything reachable from the keyboard to highlight and for the screen reader to read, putting in descriptive alternate tags in elements that are not understandable to a blind user and so on. Software that relates to accessibility in some form usually falls into the front end of development, due to the nature of it being responsible for the parts the user interact with.

Tester Testers work on validating that the product the developer has created is in working order. They put the product through a trial of tests and report back on any shortcomings. If accessibility is a concern testers should be familiar with things that might make a software product non-accessible. They should also be familiar with the tools that can help their ability to detect accessibility flaws.

Team leader A smaller group of those I encountered are those who identified themselves as working in leadership positions. Some of the leaders worked as developers while some only worked with managing the other workers, with tasks such as shielding the team from outside interference, communication with higher management, etc. Leaders have a high degree of influence and can, therefore, affect how prioritized accessibility will be for the team, by making accessibility checks mandatory or sending workers to educate themselves on accessibility issues.

Other roles I encountered very few who did not fit the roles previously described. Some of those people worked with web content, creating the type of visual content that would feature on their website, and would not be a part of the team working with the development of the website. Their inclusion is relevant as they do create content that will feature on websites and will therefore possibly create things that might break with

accessibility standards. And it makes sense that content creators should be aware of the issues relating to accessibility and possess knowledge that can aid them to identify these flaws. We also had one person from marketing who might be interested in including disabled people in their marketing demographics.

Chapter 3

Agile

3.1 Agile development

Modern software development often features some form of agile development process whether it would be scrum, extreme programming, kanban, a hybrid of many or some other method. Employing agile practices have become mainstream (Stray et al., 2017). Users of agile claim some of the benefits of adopting its practices and philosophy is increased fluidity in planning, more transparency in a project, increased productivity and higher quality work (Version one, 2017). The main principles in agile are rooted in its Agile Manifesto which outlines twelve principles that describe vital concepts to identify what values agile represent. (Beck et al., 2001)

While many of the principles could be interpreted to justify the need for agile accessibility methods I will here highlight the principles I deem the most relevant for promoting accessibility in agile software development.

- **Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.**

  If valuable software includes it being accessible then and the delivery is to be both early and continuous then it would require accessibility testing to be conducted at a low enough cost for it to be included in every delivery where it is relevant.

- **Continuous attention to technical excellence and good design enhances agility.** Good design is at the core of creating accessible content.

- **The most efficient and effective method of conveying information to and within a development team is face-to-face conversations.** Some of the methods we explore in this paper, notably personas and Cambridge Simulation

Glasses enables a high degree of human interaction to be conducted for testing purposes and understanding of the accessible.

3.2 Agile testing

The values of agile can have many implications for how software testing is conducted. Regardless of how a team integrates Agile principles into their work process, certain patterns should be expected. Delivering frequently coupled with a strong emphasis on working software necessitates constant testing of both new features and regression testing to be able to verify progress. Many Agile processes like XP demands that everybody is involved in writing tests and everyone is responsible for quality. This breaks with the notion of relying solely on dedicated testers, instead directing everyone in a team to test their own work (Talby et al., 2006). Certain ways to go about testing gets discouraged by Agile values. Relying on testing late in the development process and using resource heavy expert-testers become problematic. Regular testing necessitates time and cost-effective methods so as to not go over allotted time and budgetary constraints.

3.3 Accessible software with Agile values

Achieving accessibility in software is no matter of course, made apparent by the numerous failures to do so (Sánchez-Gordón and Moreno, 2014). There is a great deal of concern that usability is not given its due importance in agile methodologies, and accessibility existing in a more strict subset of usability, one can deduce that the same concerns would apply to accessibility and speculate if accessibility is even more deprived of its appropriate focus, due to higher technical demands (Kane, 2003)(Eklund and Levingston, 2008).

Testing for accessibility entitles testing for a multitude of different conditions and disabilities in various states. When ensuring a product works for users with eye-sight related issues, for example, one has to take in to account for different levels of visual perception, color blindness, complete blindness, tunnel-vision, etc. All affecting the user experience in different ways, and creating different problems one has to account for. Without any tools, a developer would have to retain an extensive amount of knowledge in order to meet the demands of accessibility testing, which would be hard to put into practice. One solution is to hire in expert help, people with real disabilities or expert knowledge testing the product, however, this is an expensive solution, and in an Agile process, which often feature short iterative cycles, it would be difficult to fit such costly methods

into regular testing procedures. This can push accessibility testing to the late phases of development, which is what one tries to avoid in agile processes as it breaks with it principles about delivering working software often (Eklund and Levingston, 2008). Using the WCAG standard, another commonly practiced method, as a guide for testing is also suspected to not integrate well with agile methodologies (Bai et al., 2016, citepd in. p.17) . The extent of how thoroughly Agile is practised can fluctuate and conceptions are lingering from the days of the waterfall model might still affect accessibility as it there were traditionally practiced late in the development (Luján-Mora and Masri, 2012).

An important and emphasized point in Agile is the drive to prioritize delivering working software often and to use it as a measure of progress. For software to be considered working it needs to be thoroughly tested and ready to be delivered to customers or production. In the event of software failing to be sufficiently accessible then it cannot be considered to be working. For the goal of working software to be successful, accessibility testing needs to be a part of every iteration that involves features which can affect the accessibility of the software.

The agile tool Agile mostly concerns itself with methodologies, however, the tools used by Agile developers still need to fit into these methodologies, if they are to work. A carryover from more traditional development processes are heavyweight tools that have complimented heavyweight methods. Kelter et al. (2002) lays out the requirements for agile tools as needing to be flexible and adaptable while avoiding unnecessary details. The tool needs to be easy to use and should have a low skill floor to minimize the skill required to get value from the tool and make it possible for team members in different roles to utilize it. Tools should be able to integrate into other services and be configurable. There is also possible pitfalls that might befall agile tools, as reducing tools to the bare bones can leave it to skimp out on requirements and quality. Constantine (1995) delves into the problems relating to tools in software development, especially for development of usability work. He highlights that developers crave for good tools that don't slow down, confuse or complicate work. Tools are more favorable then guidelines as both user interface standards and usability guidelines is not a very effective way to create usability within software.

The accessible agile test tools One of the largest challenges about testing for accessibility is the number of different disabilities one have to account for. Using one agile super-tool that can take account of everything that is defined in WCAG is not realistic due to the diversity of different disabilities and the different ways it affects the users.

When testing with accessibility in mind, you need to evaluate text, pictures and other media, comprehensibility, compatibility with other accessibility tools, such as a screen reader, contrasts and much more. Several tools that can each fill out its own niche and together cast a broad enough net to catch most of the accessibility errors. Some of these can be highly agile methods to be used in iterations while the more traditional high-cost methods can be used more sparsely and in the latter parts of development.

3.4 Related research in accessibility testing in agile development

In preparation for the task at hand, a preliminary literature review was conducted to uncover previous research about accessibility testing in an agile development environment. Using scholarly literature search engines such as IEEE Digital library, Scopus, Google Scholar, and web of science, an overview was established mapping out research relevant to the project. The search term used was "agile", "agile software", "accessibility testing" and "testing for accessibility". Summarizing the work that can be said to be related, I would say of those papers encompassing both agile and accessibility testing, few ventures into practical solutions to the problems at hand. A fair amount of recognition is given to accessibility testings absence in agile development, and the solutions proposed rarely give a substantiated framework to adopt. Given the low amount of relevant data that has been accumulated, there can be concluded that research regarding accessibility testing in agile development is fairly absent.

Dissatisfaction surrounding traditional accessibility test methods is a theme I found to be prevalent in other research. Usually as a cause to delve into thoughts surrounding more agile solutions. As Luján-Mora and Masri (2012) argues, accessibility is difficult to achieve with traditional software development practices, mainly due to its habit of being implemented to late in the development process and proposes that agile development can significantly help to improve web accessibility due to its focus on regular testing. The paper does not mention the difficulties in regular accessibility testing in agile development. Similarly, Eklund and Levingston (2008) argues how usability can benefit from agile development. They propose a set of steps one can use to incorporate usability techniques and agile development. Exercising smaller tests and reviews to cover more of the development process and to extensively rely on external consultants with expert knowledge on accessibility testing. Ferreira et al. (2007) concludes after several case studies that usability testing in iterative environments can lead to increased testing, however, it has to be made part of the development strategy and integrated into the work process. The research shows that there are several different ideas on how

to go about testing for accessibility and usability, but it is difficult to discern how well some of the proposed methods are performing. Meszaros and Aston (2006) ventures into early and frequent testing after conducting a case study describing the introduction of early accessibility testing in an agile development process, where using testing methods such as paper prototypes to locate accessibility bugs, wizard of Oz testing and usability stories. This resulted in a higher acceptance by end users, however, the quality of the accessibility testing is a little unclear. Kane (2003) highlights the lack of usability testing in agile development, and proposes techniques to integrate more accessibility testing into already established agile practices.

Part II

Research

Chapter 4

Research overview

4.1 Overview

Research roadmap The journey for this thesis began as a continuation of work described by Bai et al. (2016) in 1.1. Initially I set out with a case study in mind as the main road to collecting data, which focus on mainly qualitative methods. Guided by the case study approach described by Eisenhardt (1989), the pilot project and my own initial search for related research, I could move towards developing research questions and focuses, and I could begin crafting the methods of which I would use to collect data. The methods and activities were put to work, making sure avenues of data collection were overlapping and synergetic. I tried to be flexible in this phase, and some adjustments had to be made, and some opportunities were taken. We made good use of early sessions to work out any problems in the methodology and adapt it to fit the particular situations we where facing. Such as using sites from outside vendors when testing to rule out a lot of the internal system knowledge the developers possessed. By employing interviews, observations and questionnaires combined with simple statistical models I would work to strengthen and triangulate the evidence needed to build theories. Multiple avenues where pursued to gather data. As Patton and Yin highlights, multiple sources of information is critical to credible data (Baxter and Jack, 2008, cited in).

Research structure I have utilized different methods to collect data and to ensure clarity I have elected to split the information into three parts for the remainder of this thesis. One part is everything that relates to the case study, where I conducted observations and interviews over multiple days in a Norwegian bank. The second part revolves around a survey we sent out to gather data on accessibility views and attitudes. The survey largely constituted the people we would late use for user testings, however,

it was also shared on an accessibility forum for developers. The third part is about the user testing of accessibility methods. The user testing involved interview where participants were observed using different methods to test accessibility. Their reactions where documented and the participants filled out a survey for each method they tested reviewing the methods.

4.2 Preserving Confidentiality

Before collecting information that could potentially be personal or lead back to any participant, an application was filed and approved with the The Data Protection Official for Research at the Norwegian Centre for Research. appendix C

Chapter 5

Case study observations

Part of the study was done as an observation of a software development department in a significant Norwegian bank. To develop the knowledge needed for understanding the issues and potential solutions to agile accessibility, data was compiled by observing workers in their daily routines, documenting their behavior and interviewing workers with different responsibilities.

The bank is involved in banking, insurance, and pension services and develops software for both internal uses and for the customers to use. A large part of the bank's software focus is web services for customers, such as online banking.

Mainly I spent my time with one of the teams dealing with a financial solution for web customers. The observation was conducted to get a closer understanding of how accessibility was handled in an agile software environment and to gain exposure from different perspectives. It also allowed us to establish rapport with many of the participants, as a lot of the testing would be conducted there. 18 workdays where spent observing the workers in their daily routines, and being a part of stand up meetings, retrospectives, and team gatherings. Meetings where documented and details regarding participant number, conversation topics and other trends were noted down. Particular attention where applied in matters where accessibility could play a part. Conversations were conducted with team members regarding their work process and how they dealt with accessibility.

Overall accessibility rarely came up as a subject for the work currently being worked on. This can have both been because accessibility was not a particular concern in the team and because they where not in the process of doing work that would directly lead to accessibility issues. It did help in shaping how I viewed the tools in the discussion

segment as I could better understand how the tools we where testing could be useful in continuous software development.

Observations	**Observed**
Full workdays attended	18
Stand up meetings attended	15
Meetings attended	3
Interview with worker	4
Retrospective exercises	2

TABLE 5.1: Overview of events observed by me

Three in-depth interviews where also conducted with team members covering the team-leader, developer, and user experience designer roles, and gave me a better understanding of how the firm handled accessibility testing, and how accessibility was viewed and handled in different work roles. The interviews followed a semi-structured form, with a set of questions with the topic of how accessibility was a part of the work process, as a guide to bring up appropriate responses. Questions where developed following guidelines from "What is Qualitative Interviewing?" (Edwards and Holland, 2013). Interview guide in appendix E.

Chapter 6

Attitude and background survey

Before we had participants test accessibility tools, a survey was sent out to get background information on the participants and to get an understanding of how accessibility was viewed among different people. Gauging how the participants viewed accessibility was an important step towards getting a better understanding of what can be done to improve accessibility. If accessibility was viewed as unimportant then the impact of better accessibility tools would likely become negligible. We were also interested in how accessibility was incorporated in the work process to get a view of how integrated accessibility was before we exposed them to our methods.

The survey consisted of 28 questions starting with some general background information relating to age, experience, and work-roles. Largely the survey probes about the attitudes and knowledge the question taker have regarding accessibility. This information will enable us to potentially spot trends regarding what type of people are interested or use time on accessibility related issues. It will also potentially allow us to review the methods with more knowledge about the people who potentially would use them, and let us understand the issues that might arise when testing with accessibility in mind in a broader context, rather than just looking at how the test methods perform on their own in a confined setting.

This initial survey contains three data sets. 20 surveys were filled out by an accessibility interest group composed mainly of people who likely have a high interest and some experience in dealing with accessibility challenges. Another 27 was done by a software consultant company of which many also tested accessibility methods, and the last data set was completed by the 47 participants who reviewed our testing methods, all involved worked in some capacity with software development.

6.0.1 Background information form survey

The participant's background were mostly male with 71% of the participants and the age range where from 20 to 61 years old with a distribution as seen in figure 6.1. Mostly everyone involved considered themselves to mainly be involved with the software development aspect of their product in a developer, tester, designer or leader capacity, and the rest were involved with content creation. Visualized in figure 6.3 Accessibility related issues mainly fall into the front end realm of software, which have been reflected in our survey with 52 % of the participants reporting that they were mainly involved with the front end aspects of their respective software products. There was also an additional 35% who reported to work front and back end. 90% reported to have a higher education as seen in figure 6.2

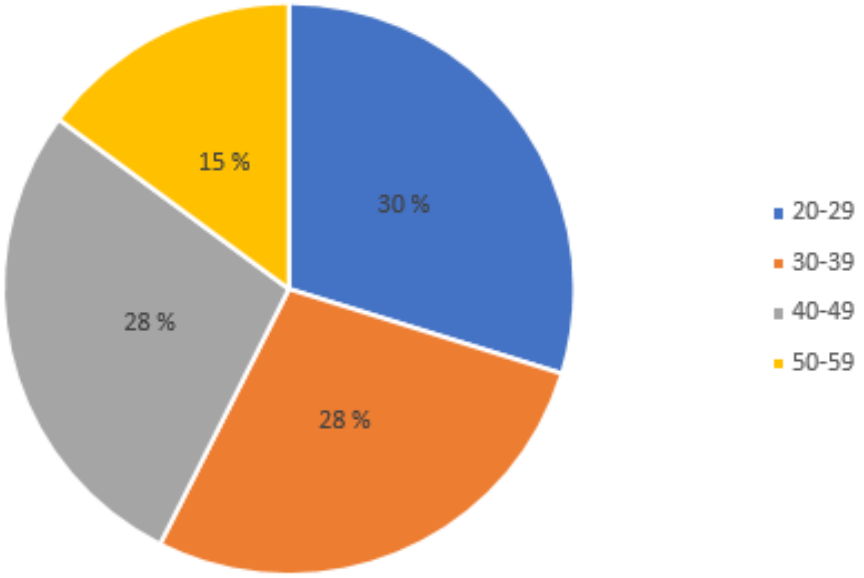

FIGURE 6.1: Age distribution of the participants

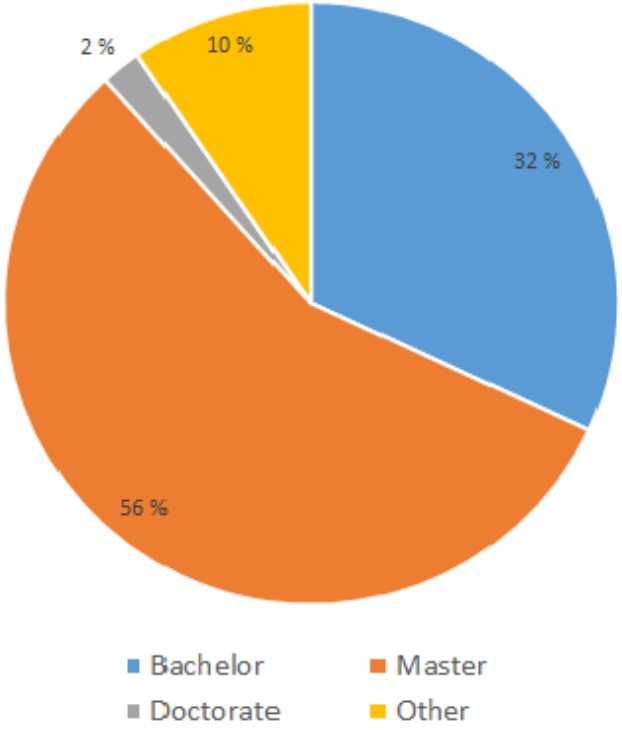

FIGURE 6.2: Education distribution of the participants

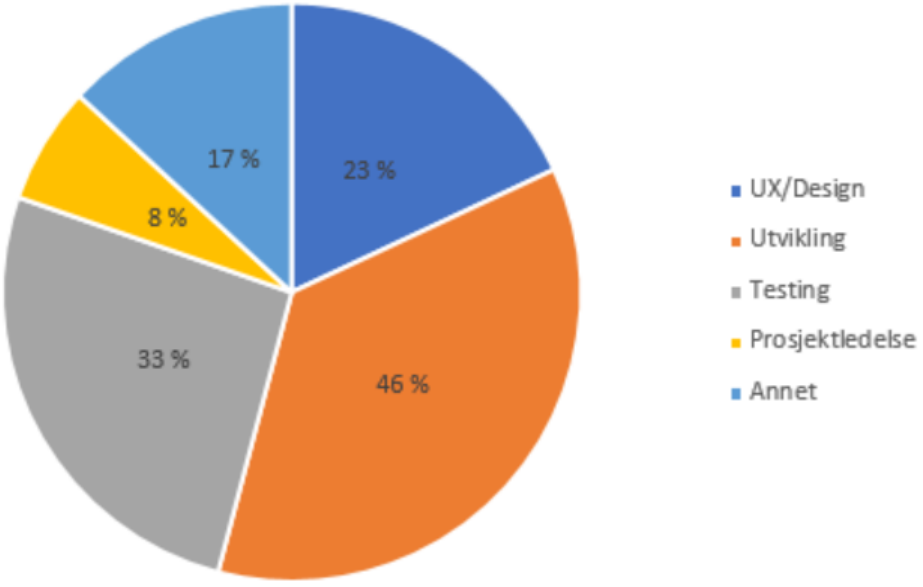

FIGURE 6.3: Work role distribution of the participants

Chapter 7

User testing of accessibility methods

The reviews of the different testing methods where conducted by hosting interview sessions where the participants got to try out testing for accessibility. It involved several different companies who develop software in different capacities. Two of which where large private Norwegian banks, one being the same bank described in the case study. We also visited one private software company developing software security solutions. From the public sector, we had interview participants from one college, a government agency affiliated with road infrastructure and a government-owned lottery company, all with their own in-house software development department. They all create software that is available to the public and are thus in a position to offer products that are of interests to people with disabilities. The interviews took form in a qualitative semi-structured arrangement.

Questions and testing tasks were developed after taking account for what we wanted to find information on and how much time we could get with participants, as recommended by Edwards and Holland (2013), every tester had filled out the background survey before the interview.

Interviewing while the participants were trying out different methods was done with every method except for testing with WCAG, as it was considered too time-consuming and could not be fitted into the allotted time for the interviews. Participants who reviewed WCAG did so outside the conducted interviews and reported their findings through the USE Questionnaire. One interview session was scheduled for one hour each, where we had time to test three methods, which involved the participants getting a quick brief of the tools with instructions on how to use them. The number of participants for each interview ranged from one to four, depending on how many interviewers and

interviewee where available. We hosted up to 4 interviews for each day, with two or three being the number for 7 of the days we interviewed participants with testing methods. There were 9 days where interviews were conducted, always with multiple sessions. I attended 6 of those.

The participants would then be instructed to solve pre-made tasks that required them to use the tools to locate accessibility faults in a website. After a pilot attempt of the interview format, it was decided to use a publicly available website not affiliated with the participants, after we found the participants knowledge of their own products to color their ability to judge accessibility. As the testers used the tools they would comment on their experience and be prompted with questions by the interviewers to gauge the experience. The tasks where each made with the intended tool to use in mind and is further described in the method evaluations, section 8. After completing the tasks the participants filled out a questionnaire as described in 7.0.1. Having finished testing the participants were asked a series of questions asking them to express their thoughts on each tried method, comparing the methods and identifying which one they preferred and which one they disliked. They were further asked about how easy or hard the methods were to use, the ramifications of accessibility testing and how they though accessibility testing could be integrated into their own routines.

7.0.1 Method specific tasks

The different tools demanded different assignments to the participants to facilitate the tools finding bugs relevant to the tool. The assignment for each method is described in this section and the tasks themselves can be viewed in appendix A.

SiteImprove Before we began the participants were instructed to install the extension on their own computers and were also given a quick overview of what it does at the start of the interview. The SiteImprove User Test was conducted without any specific tasks or websites in mind, as the tool does not need very special web features to be able to highlight its capabilities. Rather the participates would use the site that we had tested with on previous tasks and focus on the feedback provided by the tool. The participants would explore the tool and investigate the different layers of information. The interviewers would guide them if they got stuck or had any questions pertaining to its use. After completing the tasks many participants also switched to sites they where working on or was more familiar with to get a better sense of what the tool was telling them.

Personas interview Prior to being tasked with evaluating Personas, the interviewees were first given an introduction to how personas worked. The Persona character that where to be used was pre-made and each participant were given a character sheet describing the Persona and its characteristics. We mainly used a character with a cognitive disability. The character was described as a young female who would easily lose her concentration, get confused, struggle with memorization and with a fluctuating motivation. In addition, the characters education, social status, personality, interests, and profession were described in detail. The participants were then asked to imagine themselves in the role of the Persona and then try to complete a set of tasks given to them. The tasks instructed the participants to find information on an airliners website, with goals such as "Find information on bringing a cat on a flight". The participants would comment on their progress and get interviewed as they progressed through the tasks.

Screen reader interview The screen readers we tested was the NVDA for machines running Microsoft operating systems and Voice Over for Apples MacOS. The participants used their own machines for the most part, and the operating system they used decided which screen reader would be used. In case the participants had to borrow a computer NVDA was used. The participants were given a short introduction to the tool and how to use it. They were also encouraged to instruct the screen reader to display the output in a text box, instead of having the words said out loud, as the computer-generated voice can sometimes be difficult to understand for novice users of a screen reader and some found the machine voice to be unpleasant to listen to. Many also expressed quality of life improvements to have the output in text format.

The participants were given tasks to navigate through a website, where the primary objective was to order some airplane tickets. The participants got to experience using the screen reader and experience some of the faults a screen reader can uncover. While completing the tasks the participants were prompted with questions regarding the tool.

Simulation glasses After a brief introduction and determining how many glasses each tester should wear, the participants who tested the simulation glasses where given tasks to complete while wearing the spectacles. The tasks involved interacting with a web interface.

Dyslexia simulation Before we began the participants were instructed to install the extension. They were instructed in what the tool does and how it can help them discover accessibility vulnerabilities. Participants were tasked to navigate a website with the

dyslexia extension turned on. The websites visited were chosen for containing a lot of text to allow the extension something to work with.

WCAG Using WCAG for testing is time intensive for anyone not intimately familiar with all of its content. It can easily take over an hour to go through the list and use it to evaluate a website, so due to the time it takes to do any sort of meaningful testing with WCAG, the subjects conducting the assessment were doing so outside the allotted interviews. No data was recorded other than the survey they had to fill out post-test. Some of the participants did, however, have previous experiences with WCAG and where able to state some of their opinions on WCAG and how it contrasts with alternate methods during an interview regarding the other tools. The participants were given an excel document with all the WCAG checkpoints both in Norwegian and English, and were then instructed to use it to test a website for no more than one hour and then fill in the post-test questionnaire.

USE Questionnaire for tool evaluation In addition to interviewing the participants while testing the methods, every participant filled out a survey for each method they tested. The survey for evaluation of the testing methods is a post-session rating metric named the Usefulness, Satisfaction, and Ease of use (USE) Questionnaire. The USE Questionnaire consists of thirty questions and it assesses for each method if they can be a valuable addition to a testing process, by rating a product by stating general questions about satisfaction with usability and ease of use on a seven-point Likert scale, where the question taker can state rate how satisfied they are with their experience(Albert and Tullis, 2013) .

See appendix D for the full question list.

End interview At the very end of each interview session, we asked the participants a set or prepared questions to gauge what they thought about the methods. They were asked to compare the methods in different ways and to talk about how and if they would consider to use them in development. Some discussion did naturally ensue, and it did give the participants the opportunity to reflect on the experience of accessibility testing in a broader sense. All the questions can be found in appendix B.

Part III

Results

Chapter 8

Findings overview

The findings are presented in the same three parts as in the research portion of the thesis.

In the first chapter I present my findings in the case study, followed by a chapter of what we learned from the survey. The third part is about the user testing. It outlines the results we received from each of the methods we tested, both in terms of feedback from interview and the statistics we received from the USE questionnaire.

Chapter 9

Case study results

My time in the case study was mostly used passively observing and casually engaging with the workers. No notable results that I can tie into accessibility testing came of it. When I went for an active approach by setting up interviews with the workers, I was able to get some insight that helped in understanding the challenges surrounding accessibility in software development.

9.0.1 Interview results

Team leader The team-leader was very positive for developing accessibility products and would encourage the workers to think about it, but was not directly involved in any processes that could enhance accessibility.

UX designer More fruitful was the interview I had with one of the teams UX designers, who also had a lot of the responsibilities surrounding accessibility. What I learned from the interview was that accessibility was only briefly covered during the designer's education and that the designers are burdened with most responsibilities when it comes to accessibility, which could be frustrating as the designers were few and there are aspects of accessible software that fall outside the main domain of the designer's duties. Knowledge about accessibility was mainly gained through internal knowledge sharing within the companies designers and when outside accessibility consultants were involved. The designer found WCAG to be very frustrating as it was complicated to use. Wave, a tool similar to SiteImprove, was used a lot, but it was viewed as a bit unreliable. Further on, the designer wished for more diversity in accessibility tools.

Developer The interview with the full stack developer did not produce a whole lot of opinions about accessibility and software as it was not something he was particularly concerned or interested in. Nevertheless, it was educational to talk to a developer who was not involved with accessibility to try to understand the reasons for its absence. Accessibility not being a theme in work discussions or in software educations was his stated reason for not knowing much about accessibility in software, although he felt it was something he wanted to know more about.

Chapter 10

Results from attitude survey

Here are the results from the attitudes regarding accessibility part of the preliminary survey. The questions quantifiable with numbers are presented with graphs to help visualize the data. The graphs show the number of people who answered on the Y axis and the X axis represents how positive they are to the statement or question presented in a Likert scale 1 to 7, where 7 is the most positive value.

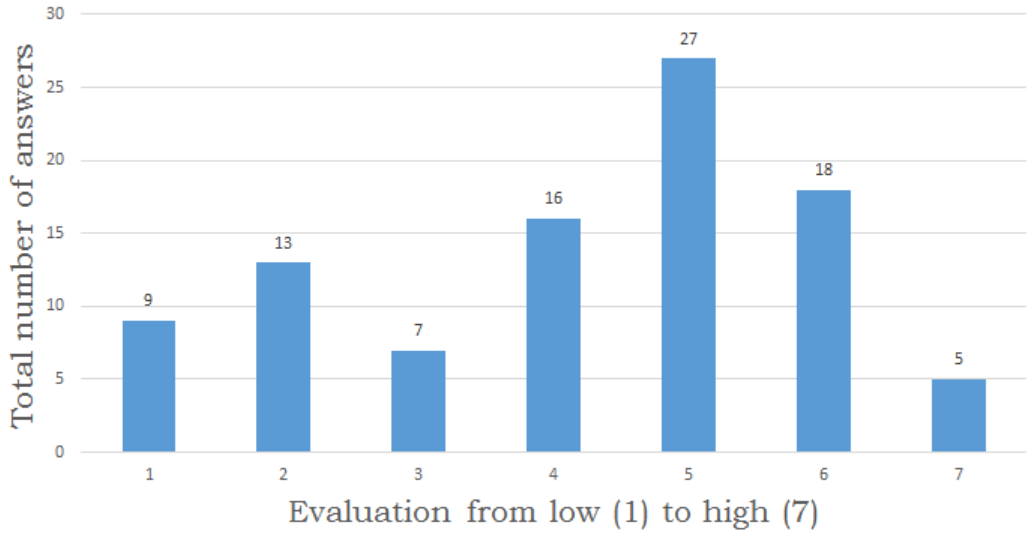

FIGURE 10.1: Evaluation of how well a accessible product can contribute to increase the marked potential for software projects

How well an accessible product can contribute to increasing the market potential for software projects Most people agreed that accessibility contributed to increasing the market potential of their product. This show awareness of the developers

that they lose potential users if the product fails to be accessible. There is still a noticeable portion of people who did not think that an available product would make any difference.

How well they taught their leadership was acquainted with accessibility Most put themselves in a neutral stance, and there are people on both sides of the scale, the leaders themselves tend to state that they have an excellent relationship with accessibility while UX designers give them a lot of the bad reviews.

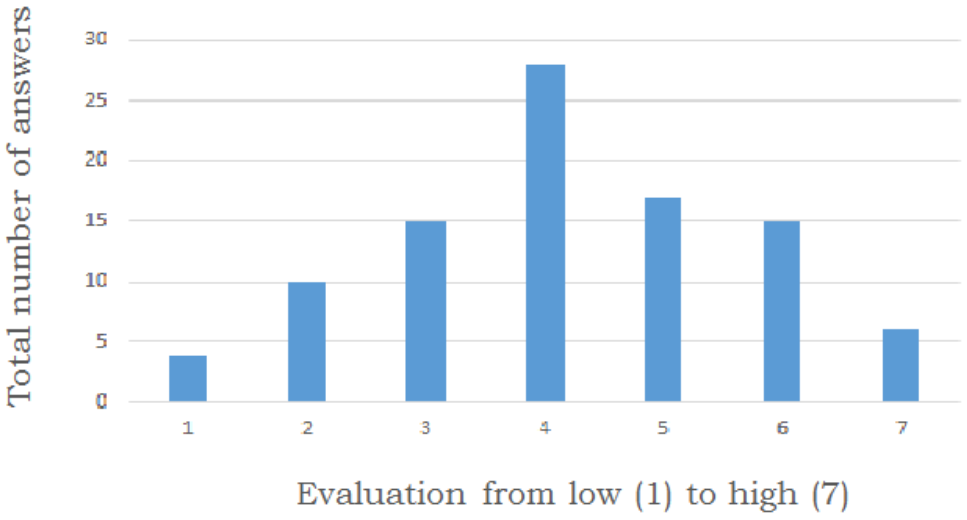

FIGURE 10.2: Accessibility rooted in the leadership

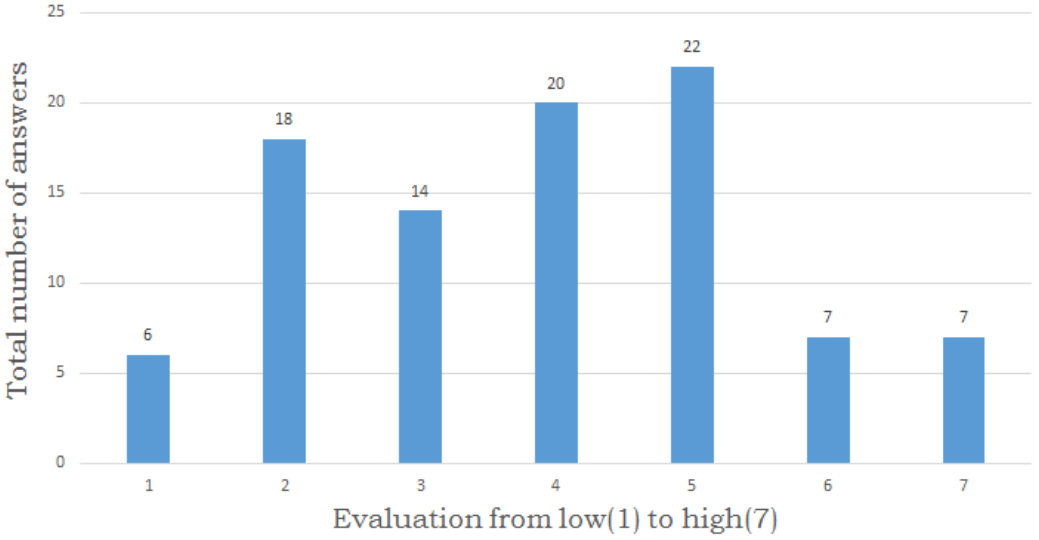

FIGURE 10.3: Evaluation of if there is a lack of knowledge and tools to practice accessibility in the work process

If there is a lack of knowledge and tools to practice accessibility in the work process Mixed reactions on this question, but there is a sustainable lack of overly agreeable responses with most people taking up a middle ground. It would indicate that most people feel that there are measures and tools to aid in accessibility, but that it could be better.

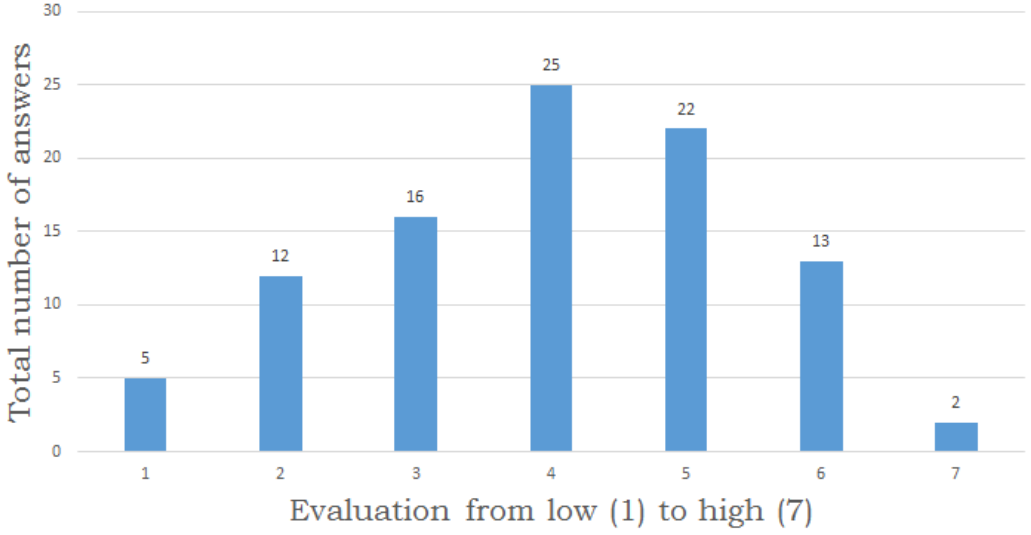

FIGURE 10.4: Evaluation of the question-takers own knowledge about accessibility

Assessment of the question-takers own knowledge about accessibility Most of the participants gathers around the middle values. Indicating that most of the participants consider accessibility to be an area where they could potentially gain more

knowledge.

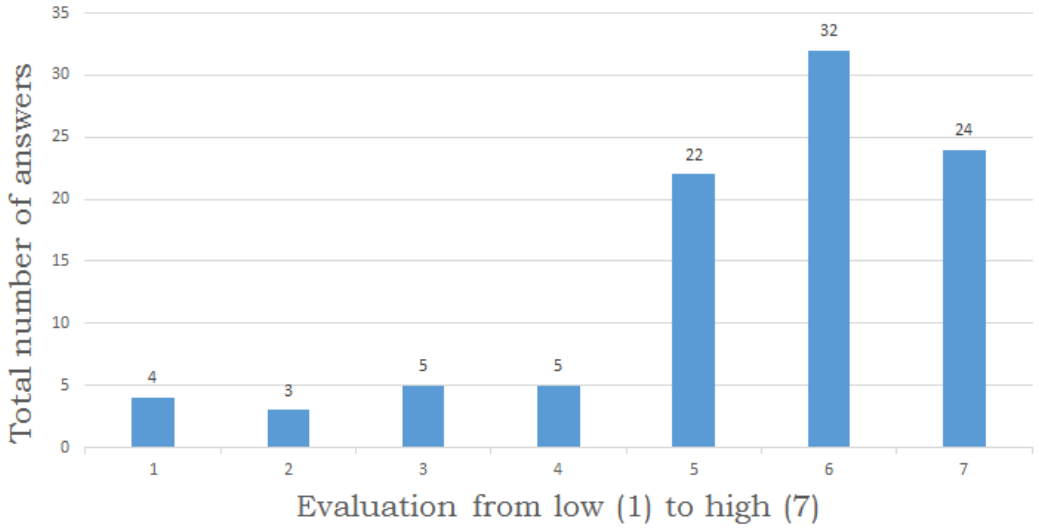

FIGURE 10.5: Evaluation of how accessibility helps to increase the user friendliness for software products

Evaluation of how accessibility helps to increase the user friendliness for software products There was very positive results on this question so it is very conclusive that most developers see the relation of accessible software and user friendly software.

45% of the people asked stated there only existed some routines for creating accessible products, and only 12% thought it well integrated the work processes. The participants who answered that it is integrated into the routines are largely composed of those who have testing as their main occupation. The answers tell us that methods for advancing accessibility in software products under development are not practiced with regular work or integrated in the way they have organized their work process.

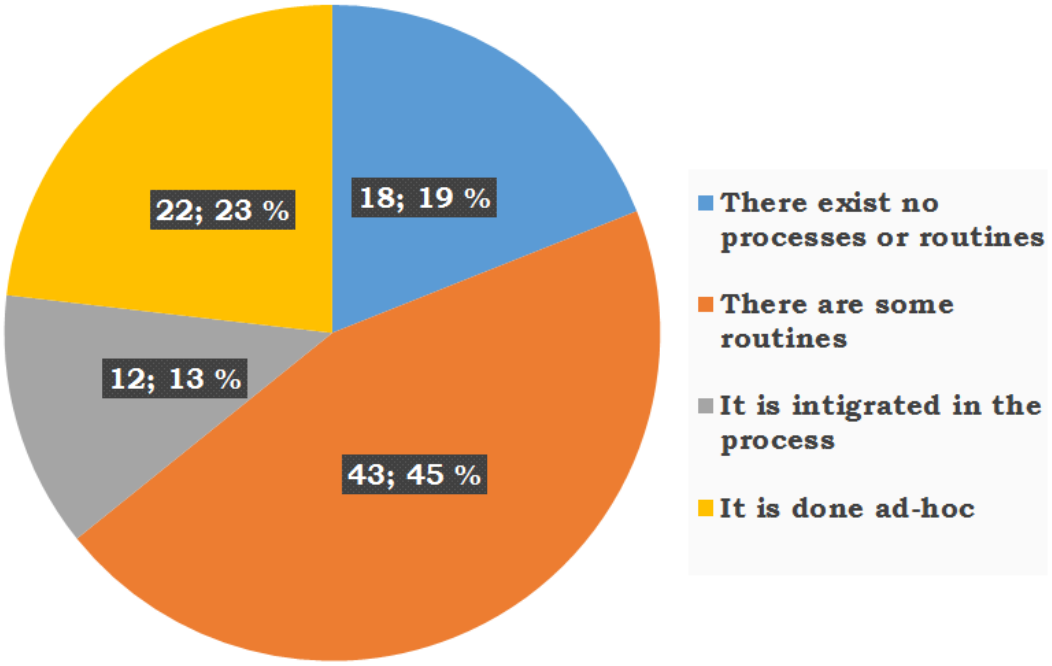

FIGURE 10.6: Evaluation on the extent of accessibility testing in the work process

The rest of the questions were not formulated with answers on a simple number rating system, but rather the participants would be asked which among a certain number of statements they mostly would agree with or be prompted to write their own answer. This is not very informative when visualized as graphs, and I, therefore, give a brief summarization of the general tendencies seen in the responses.

Who has responsibility for accessibility 34 % answered that nobody was responsible for accessibility, which would indicate that the suspicion we had that many did not give accessibility any thought or care. If this stems from lack of knowledge or lack of care is not clear at the moment, but it is certainly a negative trend that so many are not involved or know of any involvement in accessibility when it comes to their own work. The majority of the other votes where the split between everyone sharing the responsibility and several workers having responsibilities to accessibility with one person having the main responsibility.

What do you connect with the expression "Universal design"? This was an open question where the participants where free to type out heir own answer. Most like to answer that it is something to do with making a product accessible to everybody.

Who do you think should have the responsibility of handling accessibility? On this question 47 % answered that they wanted the team as a whole to be equally responsible, followed by the two next categories of designers and experts with 16 and 15 %. Many people seem to want everybody involved, but the questions regarding their actual work suggest more towards that there is only a small number of people who actually work with accessibility. Designers being in the second largest category also further confirms the concerns of the designers I talked to in the case study, who said that they had to take responsibility for too much of the accessibility work and wished for a more wider responsibility throughout the team.

Do you use external experts to help make accessible products? 58% answers no and only 19 % answered yes with the rest uncertain.

Chapter 11

Results from testing the accessibility methods

The USE questionnaire results from survey the participants filled out post interview is given as a table, and as a radar chart with the average score for all the methods included. The score ranges from 1 to 7.

11.1 SiteImprove

Category	Score
Usefulness	5.51
Ease of use	4.85
Ease of learning	5.08
Satisfaction	4.96

Table 11.1: USE score for SiteImprove

SiteImprove User Testing The feedback we received was overall positive for the tool, but many participants reported minor complaints. While the tool is more comprehensive than most of the other tools in this study, many issues where related to problems not in relation to the complexity of the tool. Some functionalities where not always working for instance, such as the ability to view the highlighted problem area directly in the page source, which was a feature highly praised by those who could get it working, and caused frustration for those who experienced technical difficulties.

Most seemed to greatly value the concrete and detailed feedback that can be lacking in some of the other tools. Many also stated that they needed more time with the tool to

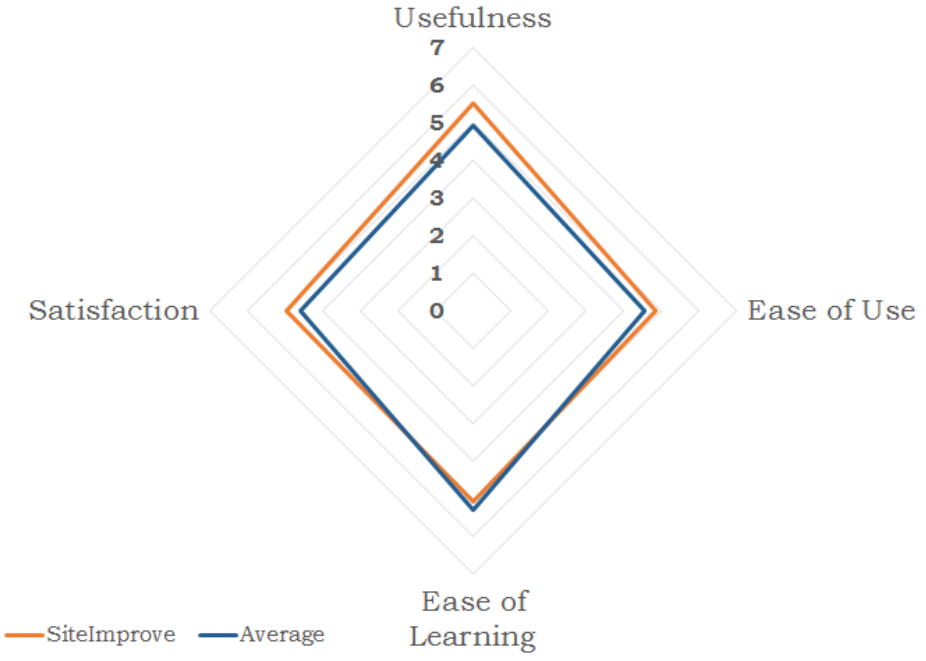

FIGURE 11.1: SiteImprove radar chart

become familiarized with all the content, but no one showed feelings that they would not use the tool due to complexity.

SiteImprove requires the user to be proficient in English in a much higher degree than the other tools, however, when we asked non-native English speaking participants if they thought this might be problematic, there where no one who thought so, often reasoning that they already were using tools in English.

Quotes from the participants who tried SiteImprove:

"I like that it can differentiate between different WCAG levels."

"It points out that there are faults, but it is hard to spot where the faults are."

"English is fine, its not a problem for developers"

As seen in 11.1 SiteImprove performs above average in three of the four categories, only falling below in Ease of Learning. SiteImprove performs best in Usefulness. This correlates well with the feedback received during interviews, as several participants stated they thought the tool could be very valuable, but they needed more time to learn it better before it becomes useful. It covers a lot of the WCAG paragraphs in a very definite and well-defined manner, which would account for it being rated one of the most useful of all the tools tested. The overall complexity of the tools and some technical difficulties are the only negative responses we encountered.

11.2 Personas

Category	Score
Usefulness	4.44
Ease of use	4.21
Ease of learning	5.44
Satisfaction	4.56

TABLE 11.2: USE score for Personas

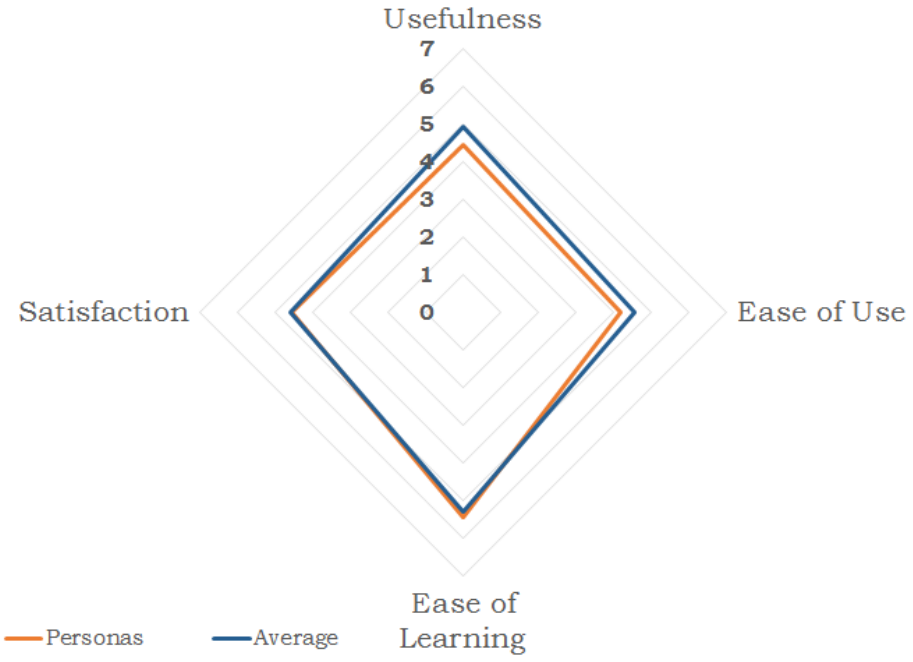

FIGURE 11.2: Personas radar chart

We encountered some very divided opinions with the participant group who tried Personas testing. The reactions where often either strongly positive or strongly negative. Of the ones who had a negative reaction, it was often stated that they could not get into the character. That they did not possess the ability to role-play as another person other than them self.

Quote from a participant who tried Personas:

"Personas is difficult to get into, it's hard to put aside personal knowledge and preferences"

This seemed to have a negative impact on the participant's ability to find accessibility flaws, as the participants could not or where not interested in identifying the weaknesses that would become problematic to the character they were supposed to assume.

Many participants also seemed to have little to no problem getting into the personas, and many praised it for its flexibility, stating that one could use it in many different

phases of development. Many also liked and preferred it to be done with several other people, in a workshop type of environment. With Persona testing, we also saw some accessibility flaws uncovered that had not previously been addressed with the other methods, such as problems stemming from mental issues. The participants acting out the persona character wherein a higher degree aware of the website as a whole, and would in a large degree evaluate the entire user flow of completing a task, rather than focusing on smaller components independently.

Personas achieve very average USE results, and it's difficult to draw conclusions other than that Personas is generally viewed in a positive light. Personas are also a very abstract tool as the Personalities used to have an endless degree of variation and its implementation will differ when different people are using it. Statements from participants interviewed, and the good USE score indicates that it can definitely be a useful and cheap tool in the right hands, but it would demand the right type of tester, someone who can get into the character, for it to have an effect.

11.3 Screen reader

Category	**Score**
Usefulness	5.49
Ease of use	4.36
Ease of learning	4.56
Satisfaction	4.81

Table 11.3: USE score for Screen reader

Using the screen reader as a tool can quickly help uncover accessibility flaws, however, the tool is difficult to learn. It has to be kept in mind that the tools main intention is to provide aid for users with seeing deficiencies and is not designed to be a tool for software development and testing. As many of the other tools we tested it puts the tester in the same perspective as the user but the screen reader distinguishes itself from those as the most technically difficult tool to use. The participants reported a definitive learning curve when testing. Disabling the audio aspect and using the voice-to-text function is necessary for the tester to comprehend the output of the tool unless the tester can take the time necessary to get used to it. The computerized voice was also described as unpleasant. Much of the negative comments on the screen reader stems from reasons dealing with its complexities. Screen readers offer a lot of keyboard shortcuts and other functionality that is intended for the regular users, not for occasional testing. While a tester does not need to learn these functionalities, they still take up space in the application and can confuse the tester who is only interested in the most basic functions.

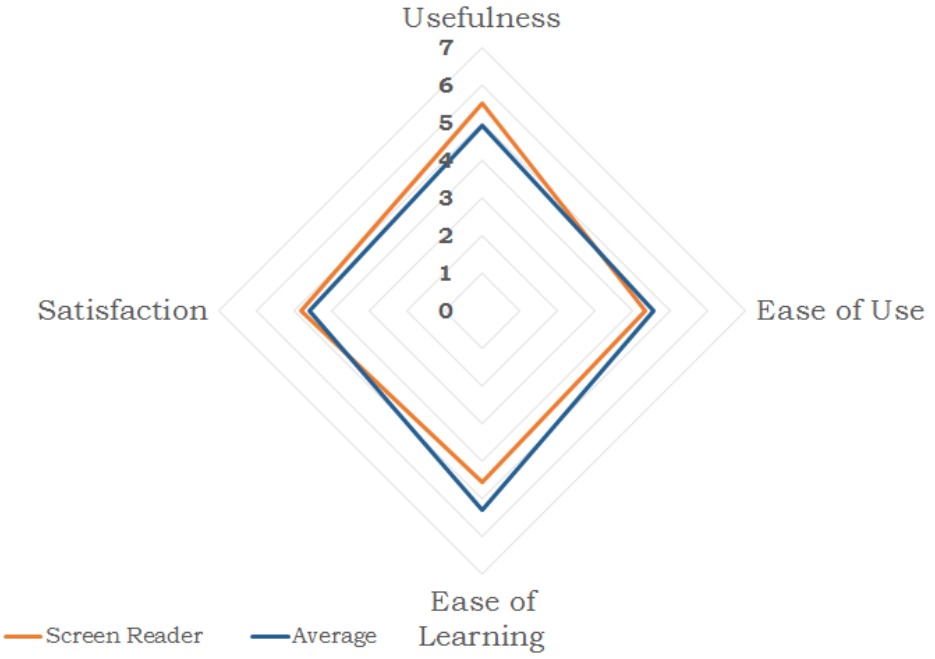

FIGURE 11.3: Screen Reader radar chart

Those who tried both voiceover and NVDA, seems to prefer voiceover in this regard, as they see it as simpler to use with a friendlier interface.

Judging by the USE score the screen reader is most successful in the usefulness category, which is promising given it is not used for what it was designed to do, but that is more reflected in the low ease of learning score, where we see that while the tool can work great it is not designed for this type of use, and there is no surprise that this makes it more difficult to use it the way we want it to function.

Several participants identified screen reader as a method they though would uncover the most flaws, however many were also expressing that they found the tool to be difficult to use.

11.4 Simulation glasses

Category	Score
Usefulness	5.41
Ease of use	6.12
Ease of learning	6.80
Satisfaction	5.54

TABLE 11.4: USE score for Simulation glasses

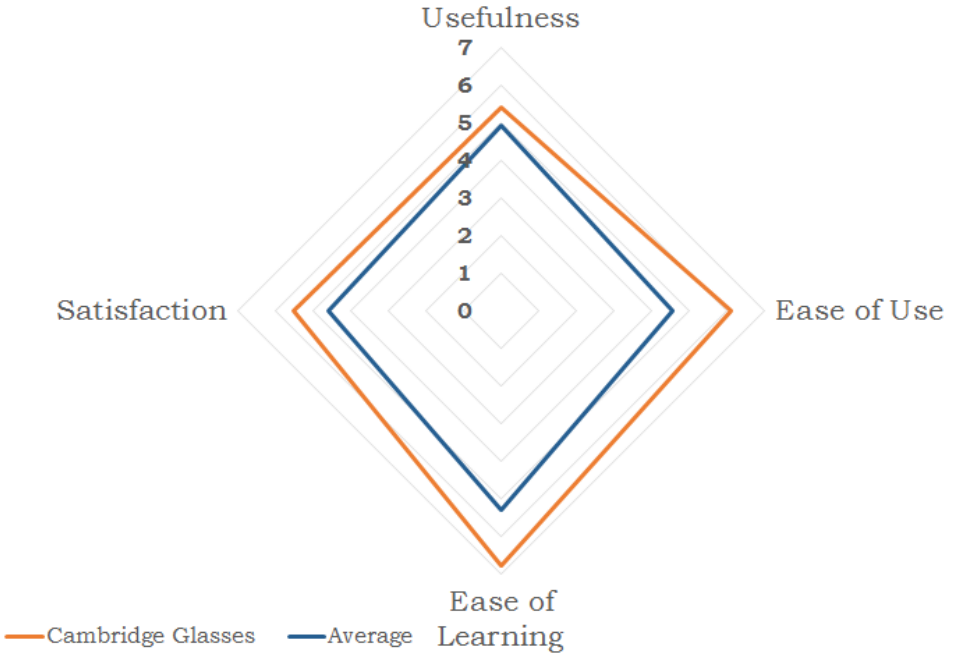

FIGURE 11.4: Cambridge Glasses radar chart

Feedback from interview Reception for the simulation glasses where overly positive. Other tools that scored high in ratings still usually had participants that expressed negative reactions, however the simulation glasses where always received with splendor reactions. Participants greatly appreciated how little effort it was to use the glasses as a tool while providing good results. The participants quickly identified accessibility faults when using the glasses, mostly in the categories dealing with attributes that might be affected by vision, which was to be expected. Bad contrast and insufficient text sizes were the main culprits uncovered, but the site we used was featuring a lot of those. Participants where saying in a much higher degree then other methods that they got a better understanding of the challenges encountered by disabled users. They also expressed that they though it was very beneficial to view a site as a whole with the glasses, rather then focusing on minor attributes as other tools might do. Several participants, especially those who had some previous dealings with accessibility testing where delighted that due to the quick and easy configuration and use of the glasses, they could now more easily make other people understand some of the accessibility concerns they had, stating that it could often be difficult to show other people how bad contrast and similar visual accessibility flaws where affecting a site in a negative way.

Quotes from the participants who tried the Cambridge Simulation Glasses:

"Glasses are also good for sketches, you can start accessibility testing early"

"People will understand more when I'm talking about contrasts"

"The glasses gave a higher degree of understanding, unlike a contrast checker that tests one ting at a time. The glasses enables you to see everything in its entirety"

The Cambridge Simulation Glasses scores above average in every category. It was only 0.2 points from reaching the limit of 7 in the Ease of learning category and also did very well in satisfaction and ease of Use.

There is little doubt that the simulation glasses are the overall highest rated tool that we tested. The physical aspect was an important feature in making it enjoyable for many of the stated evaluations. Not having the tool be depended on software gave a very broad set of uses, from reviewing sketches on paper to software on any device or system. The glasses being very intuitive and needing no more adjustments after putting them on seems also to be a contributing factor in its success. The Cambridge simulation glasses seem very well suited to agile methods. The glasses ability to be used regardless of the medium undergoing testing makes it a very versatile tool. Many noted its ability to be used in every stage of development that features a visual design and that it can be used by any team member regardless of skill.

11.5 Dyslexia simulation

Category	Score
Usefulness	4.97
Ease of use	4.98
Ease of learning	6.41
Satisfaction	4.73

TABLE 11.5: USE score for Dyslexia simulation

Interview The dyslexia simulation tool is the functionally smallest tool we tested. It response we received was generally positive and nobody viewed the limited functionality of the tool as a negative. Many liked how it is very simple to use and learn. The tool did sometimes run into technical issues which was the cause of some disgruntlement. Some liked how it visualized the problem to a high degree.

Quote from a participant who tried:

"It visualizes and clarifies the problem a lot . It makes it easy to convey the problem to other people."

Interview The feedback indicates that the dyslexia extension is a easy tool to learn and use, but that its usefulness might be its largest flaw. Characteristics that make

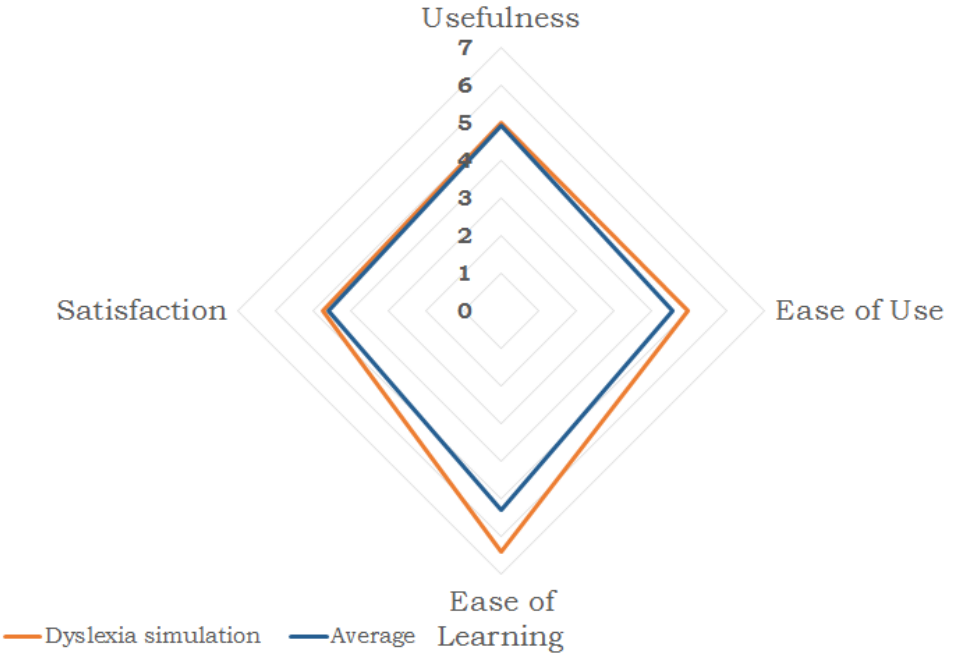

FIGURE 11.5: Dyslexia simulation radar chart

it easy to learn is likely in its automatic behaviour with little need to do any pre-configuring. The tester only needs to press the start button.

The dyslexia simulators have exceedingly good ratings in "Ease of learning", while the other metrics are very average. Its automated nature, combined with very few functionality will most likely be the cause for participants being quick in understanding the tool.

11.6 WCAG

Category	Score
Usefulness	3.68
Ease of use	2.89
Ease of learning	3.50
Satisfaction	2.95

TABLE 11.6: USE score for WCAG

It should be kept in mind that WCAG was evaluated differently than the other methods and it was evaluated for different reasons. We knew from the pilot project and from trying it out ourselves that WCAG had shortcomings as a tool for testing. Including it in this projects enables us to compare the other tools against an industry standard and to further confirm our suspicions on WCAG. We only managed to get 19 people to try out WCAG and report on it as many declined to do it, possibly because of the negative

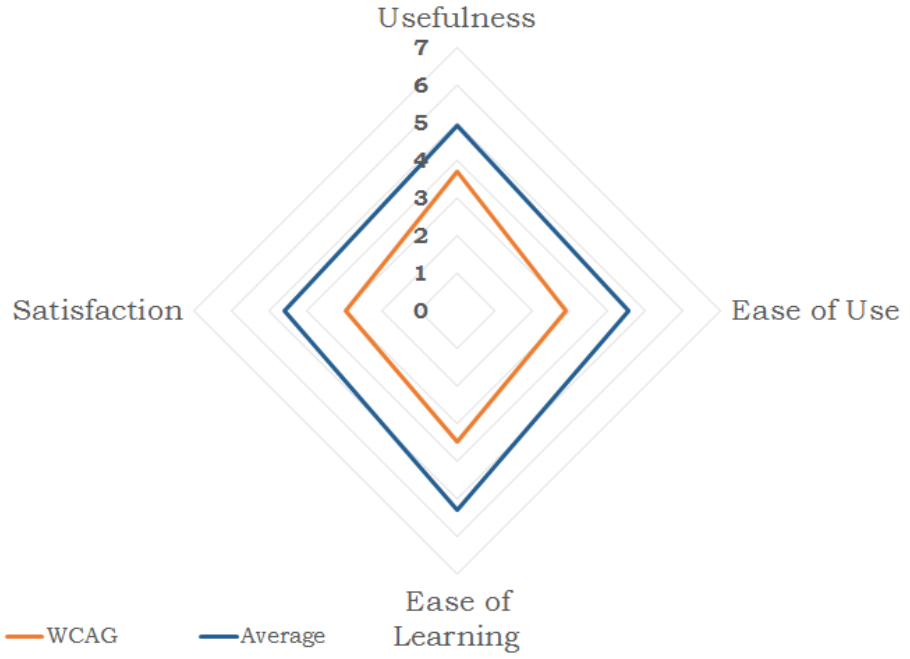

FIGURE 11.6: WCAG radar chart

associations we felt many of the participants had towards WCAG. WCAG is the least preferred tool in all categories with a considerate margin. WCAG strays the most from the mean in the Ease of learning category indicating that users might find it difficult to understand the WCAG paragraphs, which correlates well with the suspicions we had of WCAG prior to testing and from accounts from people we talked with during the case study. In interviews participants who had WCAG experience often also indicated that they found the WCAG standard complicated to use. WCAG perception as difficult to use can become problematic as the user will be frustrated when using it and some might be reluctant to use WCAG. This frustration can also be interpreted from the low satisfaction score. Also, a cause that will attribute to the dissatisfaction is the language used to convey WCAG paragraphs. It was relayed form participants with WCAG experience that it can be complicated to use and the highly detailed pages describing each paragraph can be very time-consuming to read through if there are multiple paragraphs that need to be looked into. Especially it would be hard to justify its regular use in an agile work environment where inefficient and time-consuming testing tools might not work well with regular testing, contributing to accessibility getting pushed into to later stages of a project and overall be neglected, if WCAG remains the dominant approach to test for accessibility. Some preferred to use simplified alternatives that explain WCAG in a more uncomplicated manner. As Power (Power et al., 2012) also concludes WCAG 2.0 can contribute to a lack of accessibility because developers do not understand the guidelines. WCAG scores highest in usefulness which one would expect from the de facto standard for defining web accessibility.

Quote from a participant who have previous WCAG experience:

"I use Uu.difi.no for front end work(Authors note: Norwegian Agency for Public Management and eGovernment website for universal design of ICT) *, its simpler WCAG"*

Chapter 12

Discussion

Rather than split the discussion into three parts as I have in the research and result parts I will here look at the problems I have encountered and the knowledge I have gained, and combine it into a discussion that utilizes all the different components and layers.

Luján-Mora and Masri (2012), Eklund and Levingston (2008) and Ferreira et al. (2007) all argue for smaller and more agile methods to make finding accessibility flaws more effective. Our results from using lightweight and low-cost test methods have yielded positive results for most cases which would suggest the practical research in this paper is further confirming the theories proposed in those articles. I consider this thesis potentially largest contribution is giving theoretical thinking within accessibility testing some much-needed evaluation in a context where it is practiced closer to real life. I found little research that offered concrete solutions, and they mostly offer theories about possibilities in agile development. Continuing I will discuss how the work in this paper can serve as a proof of concept of how agile accessibility testing can be done efficiently with decent results and if the theories from other researchers fit with my findings.

The Luján-Mora and Masri (2012) and Meszaros and Aston (2006) articles call for accessibility testing to begin earlier in development, arguing that more agile methods will enable accessibility testing to commence at earlier stages where flaws can get detected at a stage where change will have a lower cost. The methods we have tested in this paper are very easy and quick to use. Most of them can be used very early in development, some even before any code is written, simply going by design drawings. I don't think there is any doubt that accessibility testing can be included early in development. Personas and Cambridge glasses, in particular, depends on little work to exist, as long as there are some visual concepts that could be said to potentially posses accessibility faults.

Eklund and Levingston (2008) and Ferreira et al. (2007) argue for more frequent testing where more cost-effective methods are used several times throughout the project rather than relying on few but large and costly testing activities, such as only hiring accessibility experts to do accessibility testing. The methods I have tested all fit the description of low-cost and quick in use. Using any number of them, especially the more prominent ones will definitely enable testing to be done often. If they are to completely replace more traditional and expensive test endeavours I cannot say, but I think there are definite signs from the results that the methods we tested can take up a large portion of the testing effort within accessibility, and that there is a good chance for it to improve the accessibility of a product in comparison to one that exclusively uses and rely on the expensive methods. While our results from testing the methods at least provides a foundation to argue that most of the methods we tested will help in testing accessibility while keeping costs low, in real-world cases it will ultimately be the Agile teams effort to integrate these methods into their work process and utilize them efficiently that will determine if accessibility testing becomes successful.

Both Luján-Mora and Masri (2012) and Constantine (1995) argues that accessibility work needs to be a team effort if its to have any effect, as it promotes agile principles such as self-organization and communication within the development team, and makes it possible to inspect even the smallest of details when everyone is involved, instead of only having a few experts working on accessibility. Much of these concerned I feel are addressed in several of the methods we tested. While some were more easy to use and learn than others, they all were usable without extensive training and learning sessions. The Cambridge glasses, Personas, and SiteImprove are definitely tools that can be used across the development team and utilized in such a way that it becomes a team effort where testing can be commenced at regular but short intervals. Personas are also well suited to be used as a team exercise and it can help in enabling discussion and knowledge-sharing about accessibility within the team.

12.1 Individual methods discussion

The results from the USE questionnaire and the feedback from the interview gives a good indication on how the methods perform in a reasonably controlled environment. If they are to be used in a broader context then there are potentially other factors which might affect the testing. Following is a discussion of each method, taking account of the results and impressions discussed in the previous chapters and viewing them in a more pragmatic manner.

Site Improve SiteImprove is the most useful tool we have tested. It is quick in getting results and as a stand-in for WCAG testing, it is the method that would cover the broadest set of WCAG paragraphs. One key factor in its success is obviously the automation aspect of the tool. It cuts down the time spend finding bugs dramatically, and the tester is immediately presented with clearly defined statements of what is wrong and how one can improve. If only the tool could detect all aspects of accessibility inhibiting flaws in a website, then there would be little reason for the other tools to be used, from a developer and tester viewpoint. SiteImprove can also not be used until a website has been implemented as software, rendering it not as useful for designers.

SiteImprove definitely hits many of the marks for features other researchers have conceptualized as agile accessibility tools, being cost effective and easy to use. One aspect remarked by many of the participants who tested our methods where the positive ability some of them had to view the product in its entirety. SiteImprove lacks this feature, instead analyses very small segments at a time.

Cambridge glasses The Cambridge simulation glasses are along with SiteImprove the method that works well in many situations. It also works well in cases where SiteImprove can't help. It is not dependent on a website having to reach a certain level of development and can be used very early in development if sketches or prototypes are made. The glasses are also platform independent and can be used on any type of software or any sort of visual representation making it a very universal accessibility tool. Being able to view the product from a higher perspective rather then fixating on small details also contributes to why so many liked to use the glasses.

Screen reader The screen reader is very distinct from the other methods in that it is something that is not designed for the purpose of testing. It is designed to be used for severely visually impaired users, who have a very different view of what the tool should do. The tool can potentially be a very powerful accessibility testing tools. It both verifies that a site is compatible with screen readers and it also forces the tester to check if the site is keyboard navigable. However, since its being used against its intended use there is a clear obstacle for the tester to overcome in that it's difficult to use. The overall concept is fine and understandable, but screen readers are full of shortcuts and features to help the disabled and not the tester. The computerized voice which is usually on by default is unpleasant for users who are not heavily exposed to it, and the option of having the output in the text is not necessarily obvious. Constantine (1995) writes "What users want is good tools", and a testing tool that is not user-friendly to the tester will be problematic regardless of how powerful it is. This flaw pushes the screen

reader towards the inaccessible, and ultimately it might have the consequence of the tool not being used as often as it maybe should be. If the tester can overcome the design flaws, the screen reader is very useful and could be used as often as when changes are made that affect keyboard navigation or things that might affect the way a screen reader interprets a site.

Personas Personas demand more from the testers than the other methods. Its open-ended characteristic makes it only useful if the tester commits to using it actively. The methods give no output other then what the testers themselves can come up with, not like the other tools that can give more active aid to the tester. The outcome of its success is therefore likely dependant on how well its implemented, more so than the other methods. I think for it to flourish personas should be initiated at the beginning of development. Preferably done as a group exercise to facilitate role play and to ensure many people can get familiar with the personas. When the personas have been developed after modeling them after people likely to exist in the customer base development can continue. As the work progress the workers can have the personas in their consciousness and when developing features they might be reminded of some of the disabling traits of the personas and then avoid features that might impede the product use for that particular persona. The developers might also start to refer to the personas when they discuss the feature, as it might be more undemanding for someone to understand the needs of an imaginary customer that they are familiar with rather than name such and such feature might impede use for such and such disability. There is also the positive effect of the persona character being used over time as the developers will get more familiar with the persona, which can stick around for many years of development.

The feedback from the interview conducted during the Personas tests resulted in divided opinions. While many enjoyed using Personas an equal amount expressed severe dissatisfaction, often claiming that they could not get into the Persona role. Personas get more useful over time s you get more familiar with the characters, but its possible that many simply do not take pleasure in the role-play aspect of personas or are not capable of assuming the mindset of a different person. Both of which are crucial if Personas can be utilized in a serious manner.

Dyslexia plugin I believe the dyslexia plugin to be the worst of the litter. While it received alright results when we tested it, I think for this method the results were affected by the limited environment it was tested in as the testing mostly takes account for general testing with the tool, and does not look at it too much in the grander scheme of things. I feel after having seen all the methods in use and become accustomed to

accessibility testing that the dyslexia tool is mostly redundant. The tools only tests text where WCAG recommends to not go over 80 characters and to have a minimum set of line spacing and text size, which is all apply covered by running the site improve tool where you will get very accurate feedback whereas with the dyslexia tool the tester has to come to his own conclusion. The dyslexia tool was added partly because we wanted tests that covered mental disabilities, and dyslexia being fairly common seemed to be an excellent area to focus on, however it was overlooked that there was an overlap between what the two methods really tests for at the time, possibly because we did not have as much experience with them or accessibility testing at the time. We also did not have the results we have now, so it was not at the time possible to accurately say which one would be the better tool. Due to this, I can not recommend the dyslexia plugin as a worthwhile testing exercise.

12.2 Recommendations for industry reuse

The preliminary study (Bai et al., 2016) of which this thesis is a continuation of suggest the methods should work together as a suite of tests, where the cheapest ones in use are to be deployed early if they are suited for it and the other are gradually introduced as the development comes to a finish. Ideally, they will all compliment each other, overlap on accessibility issues to increase detection of faults while also covering niche disability areas to have a wide coverage of every issue present in WCAG. For software developers who consider to use these methods, using several of them is important for good coverage. Personas and Cambridge glasses can be utilized early, as they don't use up many resources. As the product gets going site improve and the screen reader can start to see occasional to regular use. Cambridge glasses and SiteImprove especially are so cost efficient and useful that they can be justified for repeated and regular use. While Personas could be a part of a set testing routine, I think it works just as well as a continuous subconscious though that can also foster discussion, given that there are early exercises and effort to be familiarized with the characters.

While the results have been mostly positive for the methods we tested, they can not completely replace traditional methods. Rather the new methods should replace them for the most part of active development. Testing with WCAG or hiring experts yields great results but is very resource intensive. Nearing the end, when much of the product is nearing completion and there is little chance or time for significant changes, the high resource methods can be utilized. Going through the WCAG paragraphs manually to make sure everything is within regulation, hiring professional help or any other exercise

that is seen as valuable. Observant readers will notice that I left out the dyslexia plugin as I do not really consider it necessary as described in its discussion.

12.3 Secondary effects of using simulation methods

The methods and tools I have evaluated have been used with the intention of testing software and finding accessibility faults. In the larger scheme of accessible software, I suspect that using these tools, especially the ones who simulate a disability, might have secondary benefits. Exposure to the type of simulation testing such as with the Cambridge glasses, personas, screen reader(preferable if the tester can avoid looking at the screen) and the dyslexia simulator(Note that the dyslexia simulator don't accurately simulate the experience of a dyslexic user, however the effect still works for this discussion) can give the tester a heightened sense of awareness of issues that could come into conflict with disable users and foster more empathy and understanding. This is already manifested very clearly in Personas where discussion and different perspectives emerge quickly, but the just using the other tools could give similar effects. Many comments were made when we tested the simulation tools that it gave the testers new insights and understanding. The new knowledge about how disabled users perceive software can give developers a subconscious ability to shape software in a more accessible direction, outside of directly applying accessibility methods. So while the methods described in this paper might not be chosen to be utilized regularly for testing out in the software development world, there can be a significantly insightful experience for developers who have tried a simulated disability, who can later put that experience to use under later development. This insight is mostly lost in tools like WCAG, text-based guides and analyzing software such as SiteImprove.

12.4 Whats missing in the current methods

There is still a long way to go before accessibility can integrate itself with agile work processes. One area that will need to improve is to further take accessibility to agile. Both in making existing tools more compatible with agile processes and creating more tools so that developers have a wider variety of assets to aid them in making accessible software. Reducing the resources needed for accessibility testing by continuing the work towards more lightweight and comprehensible tools, as well as making the surrounding issues with disabilities more prevalent and the knowledge of how it affects software interaction more obtainable. This will aid in weaving accessibility into the development process, making it more successful and effective.

Platform independence Another barrier is that the current tools are very platform dependent. Tools are not uniformly available on operating systems, browser and developer tools. Screen readers, for instance, are usually only developed for a specific operating system, making it difficult for teams who have people using different operating systems to work on the same tool. The different tools have their own characteristics and integration and learning can be made more arduous because of this. We saw the same problems with some of the browser-based tools where some were only available for selected browsers. Tools that support software not intended for web or mobile are also rare. The lack of diverse tools available on many platforms hampers any attempt to make accessibility a more uniform process as it now has to be individually tailor-made depending on the available hardware and software available to developers and with the right combination it can be very difficult to cover a wide range of disabilities with tools.

12.5 Limitations

Time spent with tools While most of the participants managed to grasp and utilize the tools under testing, it would have yielded more accurate results if they had been given more time to familiarize them self with each tool. The participants would then get a better understanding of how to utilize the tools and use them in a more effective manner.

It would be interesting to see if the firms we visited used the methods after we left. The impression had after coming back to the case study firm after three months since we had introduced the methods was that they had not been used. They were however not in a process of doing major UI overhauls and did not have a product with major accessibility deficiencies.

Open survey risk The survey we send out detailing accessibility attitudes received some answers after it was shared on an internet accessibility interest group. While I know which data belongs to this data group, there remains the fact that I don't have control over who really answered it. This opens the possibility that the data from this group can be compromised. For instance, actors who fall outside of the intended user group of this study can have influenced the data.

Ineffective case study Unfortunately, the case study did not yield as much data as I had hoped for. While giving me a good insight into the daily life of a software company, accessibility was rarely a topic at the time I was present. In hindsight, I don't believe it was a productive use of my time.

Chapter 13

Conclusion

13.1 Summary

The overall goal for this project was to explore methods to further improve agile accessibility testing, as accessibility in software products is an underdeveloped area. Mainly by testing a set of methods that can help aid accessibility testing, with supporting data from a case study and an investigation into accessibility attitudes. Chiefly the focus is on methods suited for web development.

13.1.1 Case study

Part of the study took form as a case study in a software development firm. I observed the workers and conducted some interviews. It did not directly yield much data.

13.1.2 Attitude survey

I conducted studies on the attitudes regarding accessibility in software development, by sending a survey to 94 people involved in software development. It resulted in an improved understanding of how accessibility is viewed and handled by professional developers and I received some interesting results regarding the challenges and opinions regarding accessibility testing in agile development. There was a clear positivism surrounding accessibility work, most seem to agree that its important for software products to be accessible and that it helps in making the product more marketable and better, this view is especially held among the leader roles, but the enthusiasm drops a bit for everyone else. This coupled with statements collected during the case study there seem to be a genuine resolve to want more accessible software, but not as strong passion to

realize this by actively integrating accessibility work consistently. Programmer tends to be the worst at accessibility. Both in general knowledge and in passion to improve on it. Many expressed the view that it was outside their work responsibilities. As many accessibility problems arise because of a lack in the software code areas this view is problematic. In the survey, it was generally agreed upon that it should be a shared responsibility of handling accessibility, but in practice, this was not the case. Again programmers are not usually interested in those responsibilities, and it lands more towards the designers and testers to ensure that the finished product is accessible.

13.1.3 User tests of accessibility methods

I conducted user tests by getting accessibility test methods in the hands of professional software developers and collecting data from their responses. I have created an overview of how well each method tested was received.

I have tested the following methods:

SiteImprove An automated test tool that checks a website for any WCAG infractions. Did very well in user testing and provides a lot of testing features. It covers many WCAG paragraphs while being fast and easy to use.

Cambridge simulation glasses A tool that blurs the tester vision, allowing the tester to experience a software product from the perspective of a user with reduced vision. Was very well received in testing, while not the most analytic tool the participants reported that they enjoyed using it and it works for testing on many different platforms.

Dyslexia simulator A browser extension that scrambles text making it harder to read. While many found this tool to be alright it is made redundant by SiteImprove.

Personas A method where the testers act as users with imagined disabilities and other traits to gain more insight. It created divided opinions, as not all the participants enjoyed getting into a character.

Screen reader Originally a tool made for the use by blind people to browse the web, however, we use it as a testing tool to verify if a site is compatible with keyboard and screen readers. It was one of the hardest tools for the participants to use, however, if that hurdle can be overcome it is a very good tool.

WCAG We also did a smaller test on how developers felt about using the WCAG standard, which is a testing method that we felt where inadequate and from the dissatisfaction spawned from WCAG came the interest in the other methods we tested. While it did not get the same rigorous testing the other methods received, the opinions I managed to gather on WCAG indicated that it was not enjoyable or easy to use.

Are the test methods suitable for agile software development? Overall the results were very positive for the methods we tested, they mostly scored high, and all did a lot better than the responses we collected from WCAG use. Some did notably better than others. The Cambridge glasses and the SiteImprove methods were well received, both working well in finding accessibility flaws and being easy to use by the tester. Personas and the screen reader was also successful methods, but they had some flaws, both being a bit difficult to use by the tester. I believe all those methods to be suitable for agile software development. The dyslexia method was found redundant by not covering any unique accessibility areas.

Appendix A

Scenarios for user testing

Scenarioer

Scenario 1 - Du skal søke om lån til bolig

1. Du er fast ansatt og tjener 600000 før skatt
2. Det bor ingen barn under 18 hos deg
3. Du skal søke med en partner som er fast ansatt og tjener 900000 kr før skatt
4. Du skal bruke 1 000 000 av dine sparepenger
5. Du skal selge en bolig som kan bli solgt for 250000
6. Du har ikke gjeld på bolig
7. Du kan stille hytta i sikkerhet med tilleggssikkerhet på 500 000

Scenario 2 - Du skal søke om lån til bolig

1. Du er fast ansatt og tjener 900000 før skatt
2. Det bor 1 barn under 18 hos deg
3. Du skal søke alene
4. Du skal bruke 1 000 000 av dine sparepenger
5. Du har ikke bolig du kan selge
6. Du skal ikke stille sikkerhet i en bolig, hytte eller annen eiendom

Scenario 3 - Du skal bestille verdivurdering

1. Du er fast ansatt og tjener 900000 før skatt
2. Det bor ingen barn under 18 hos deg
3. Du søker alene
4. Du skal bruke 1 000 000 av dine sparepenger
5. Du skal selge en bolig, og vil bestille bestille verdivurdering

Scenario 4 - Du lurer på noe

1. Du er fast ansatt og tjener 900000 før skatt
2. Det bor ingen barn under 18 hos deg
3. Du søker alene
4. Du skal bruke 1 000 000 av dine sparepenger
5. Du har ingen annen bolig eller gjeld
6. Du er ferdig med finansierings kalkulatoren, men lurer fortsatt på noe

Oppgave 1: Lavpriskalenderen

Du har lyst til å besøke din kjæreste som bor i Bergen en helg i februar. Det har ikke noe å si hvilken helg, men du vil ha så billige som mulige billetter.

1) Gå til www.sas.no
2) Finn lavpriskalenderen til SAS. Bestill en billig reise Oslo- Bergen som går fra fredag ettermiddag til søndag ettermiddag i februar. (Du vil betale med penger, ikke poeng)
3) Etter at du har fylt inn navnet ditt så ønsker du å legge til bonusprogram EBB600575700.
4) Finn et ledig sete ved vinduet

Når du har kommet til setevalg får du en telefon fra din kjære som ønsker at hun kommer til deg i januar i

Oppgave 2: Er flyet i rute

Du har tilbudt deg å hente en kamerat som kommer med fly fra London i kveld

1) Gå til www.sas.no
2) Finn siden der man kan se om flyet er i rute
3) Søk opp flightnummeret som er SK506
4) Sjekk stausen på flyet, er det i rute?
5) Du blir nysgjerrig på linken "Operert av 4 carriers", og klikker på denne

Oppgave 3: Bli medlem i Eurobonus

Du flyr ofte med SAS og har bestemt deg for å bli medlem i SAS EuroBonus

1) Gå til www.sas.no
2) Finn registreringsskjemaet for å bli medlem av Eurobonus
3) Fyll inn fornavn og etternavn
4) Når du kommer til feltet for fødselsdato skriver du inn datoen på gammel vane på formatet dag-måned-år (ddmmyy), og tabber videre til neste felt
5) Fyll inn resten av skjemaet
6) Åpne "Betingelser og vilkår"
    a) Hvem skal du kontakte for å få vite mer om Eurobonus?

Oppgave 4: Reise med katt

Du skal på ferie til familiens sommerhus i Danmark og vil undersøke hvordan katten kan være med på flyturen

1) Gå til www.sas.no
2) Finn ut hvor raskt du må bestille plass til katten på reisen
3) Finn ut hvor mye det koster å frakte katten i kabinen

Oversikt

1. Du vil registrere en ny bruker
2. Du har solgt bilen din
3. Du ønsker ekstra informasjon
4. Du vil opprette en tilleggsavtale
5. Du vil endre innstillinger

Du vil registrere en ny bruker

1. Opprett en ny privat bruker
2. Logg inn
3. Opprett en ny avtale på DL12345
    a. Avtalen skal gjelder fra i går
    b. Velg den avtalen som gir mest mulig rabatt
4. Undersøk om du har rett til refusjon av depositum dersom brikken blir stjålet.
5. Hva er forventet leveringstid?
6. Logg ut

Du har solgt bilen din

1. Logg inn med den nye brukeren
2. Finn frem informasjon om hva du skal gjøre når du har solgt bilen
3. Avslutt avtalen som gjelder DL12345
4. Velg å overføre tilbakemelding til konto 1645.01.12345
5. Logg ut

Du ønsker ekstra informasjon

1. Logg inn med puma1 bruker
2. Finn ut om faktura 12586788 med 15.02.2017 er betalt
3. Hvilke passeringer fikk PP45709 den 30.03.2017?
4. Logg ut

Du vil opprette en tilleggsavtale

1. Logg inn med puma1 bruker
2. Finn ut hva tilleggsavtale betyr
3. Bestill tilleggsavtale for DK76058 på en egen avtale
4. Logg ut

Du vil endre innstillinger

1. Logg inn med en bruker
2. Endre leveringsadresse til Baker street 212B
3. Bytt språk til Engelsk
4. Logg ut

Appendix B

Interview Questions

Questions for interview after test sessions

1. What did you think about the different test methods?
2. What method did you feel discovered the most faults?

2.a What is the reasoning for that. What type of faults did it discover?

3. What method was the easiest to use?
4. What method was the hardest to use?

4.a what specifically was harder?

5. Witch of the methods could you see yourself using the next time you perform accessibility testing?
6. how likely is it that you will conduct accessibility within the next three weeks?
7. how often would you like to test for accessibility?
8. How does accessibility impact the products you are developing?
9. How do you think these methods could be integrated into the daily routines?

Appendix C

Confidentiality

C.1

Privacy evaluation

FIGURE C.1: Privacy

Viktoria Stray
Postboks 1080 Blindern
0316 OSLO

Vår dato: 13.11.2017 Vår ref: 56615 / 3 / AGL Deres dato: Deres ref:

Vurdering fra NSD Personvernombudet for forskning § 31

Personvernombudet for forskning viser til meldeskjema mottatt 16.10.2017 for prosjektet:

56615	*Integrasjon av tilgjengelighetstesting i en smidig utviklingsprosess*
Behandlingsansvarlig	*Universitetet i Oslo, ved institusjonens øverste leder*
Daglig ansvarlig	*Viktoria Stray*
Student	*Nikolai Johan Sand Sverdrup*

Vurdering

Etter gjennomgang av opplysningene i meldeskjemaet og øvrig dokumentasjon finner vi at prosjektet er meldepliktig og at personopplysningene som blir samlet inn i dette prosjektet er regulert av personopplysningsloven § 31. På den neste siden er vår vurdering av prosjektopplegget slik det er meldt til oss. Du kan nå gå i gang med å behandle personopplysninger.

Vilkår for vår anbefaling

Vår anbefaling forutsetter at du gjennomfører prosjektet i tråd med:

- opplysningene gitt i meldeskjemaet og øvrig dokumentasjon
- vår prosjektvurdering, se side 2
- eventuell korrespondanse med oss

Vi forutsetter at du ikke innhenter sensitive personopplysninger.

Meld fra hvis du gjør vesentlige endringer i prosjektet

Dersom prosjektet endrer seg, kan det være nødvendig å sende inn endringsmelding. På våre nettsider finner du svar på hvilke endringer du må melde, samt endringsskjema.

Opplysninger om prosjektet blir lagt ut på våre nettsider og i Meldingsarkivet

Vi har lagt ut opplysninger om prosjektet på nettsidene våre. Alle våre institusjoner har også tilgang til egne prosjekter i Meldingsarkivet.

Vi tar kontakt om status for behandling av personopplysninger ved prosjektslutt

Appendix D

USE Questionnaire

USE Questionnaire

Usefulness
It helps me be more effective.
It helps me be more productive.
It is useful.
It gives me more control over the activities in my life.
It makes the things I want to accomplish easier to get done.
It saves me time when I use it.
It meets my needs.
It does everything I would expect it to do.

Ease of Use
It is easy to use.
It is simple to use.
It is user friendly.
It requires the fewest steps possible to accomplish what I want to do with it.
It is flexible.
Using it is effortless.
I can use it without written instructions.
I don't notice any inconsistencies as I use it.
Both occasional and regular users would like it.
I can recover from mistakes quickly and easily.
I can use it successfully every time.

Ease of Learning
I learned to use it quickly.
I easily remember how to use it.
It is easy to learn to use it.
I quickly became skillful with it.

Satisfaction
I am satisfied with it.
I would recommend it to a friend.
It is fun to use.
It works the way I want it to work.
It is wonderful.
I feel I need to have it.
It is pleasant to use.

Lund (2001)

Appendix E

Interview guide

Introduction

Introduce myself

Explain the confidentiality of the interview?

Get permission to record audio.

Interview questions

What is your position?

What do you work with?

How familiar are you with Accessibility when it concerns software?

Is it a part of the work-process at in your team?

Who do you think is responsible for accessibility?

Do you think accessibility is a responsibility for your role?

Is accessibility encouraged by the management?

Is it something you need to take take initiative for yourself?

Do you think accessibility gets highlighted enough?

If the interviewee works with accessibility

How do you work to improve accessibility in your work?

How often is accessibility a part of your work process?

Do you use a guide to verify accessibility?

What methods do you use to test accessibility?

For each method

Ask about how much time it uses, how much resources it drains, how useful it is and if

it is easy to use?

Do you think you need more or different tools?

What are your attitudes towards accessibility?

Bibliography

Gro Marit Rdevand Ivar Solheim Till Halbach, Riitta Hellmanm. Utformingsveileder kognitiv tilgjengelighet av nettsider og nettsteder. URL http://iktforalle.no/tilgjengelige-nettsider/veileder-hele.html.

Mary-Luz Sánchez-Gordón and Lourdes Moreno. Toward an integration of web accessibility into testing processes. *Procedia Computer Science*, 27:281–291, 2014.

David Kane. Finding a place for discount usability engineering in agile development: throwing down the gauntlet. In *Agile Development Conference, 2003. ADC 2003. Proceedings of the*, pages 40–46. IEEE, 2003.

Thomas Hugaas Molden, Christian Wendelborg, and Jan Tøssebro. Levekår blant personer med nedsatt funksjonsevne. analyse av levekårsundersøkelsen blant personer med nedsatt funksjonsevne 2007. 2009.

Anna M Kittelsaa, Sigrid Elise Wik, and Jan Tøssebro. Levekår for personer med nedsatt funksjonsevne: Fellestrekk og variasjon. 2015.

Gottfried Zimmermann and Gregg Vanderheiden. Accessible design and testing in the application development process: considerations for an integrated approach. *Universal Access in the Information Society*, 7(1-2):117–128, 2008.

Jakob Nielsen. Return on investment for usability. *Jakob Nielsens Alertbox, January*, 7, 2003.

Aleksander Bai, Heidi Camilla Mork, and Viktoria Stray. A cost-benefit evaluation of accessibility testing in agile software development. *ICSEA 2016*, page 75, 2016.

Aleksander Bai, Heidi Camilla Mork, and Viktoria Stray. A cost-benefit analysis of accessibility testing in agile software development results from a multiple case study. *Int. J. Adv. Softw*, 10(1), 2017.

Roger Pressman. Software engineering: A practitioners guide, 2002.

Eric Bergman and Earl Johnson. Towards accessible human-computer interaction. *Advances in human-computer interaction*, 5(1):87–114, 1995.

World Wide Web Consortium. Accessibility, usability, and inclusion: Related aspects of a web for all. URL https://www.w3.org/WAI/intro/usable.

Loretta Guarino Reid and Andi Snow-Weaver. Wcag 2.0: a web accessibility standard for the evolving web. In *Proceedings of the 2008 international cross-disciplinary conference on Web accessibility (W4A)*, pages 109–115. ACM, 2008.

World Wide Web Consortium et al. Web content accessibility guidelines (wcag) 2.0. 2008.

Jeffrey P Bigham, Craig M Prince, and Richard E Ladner. Webanywhere: a screen reader on-the-go. In *Proceedings of the 2008 international cross-disciplinary conference on Web accessibility (W4A)*, pages 73–82. ACM, 2008.

John Eklund and Ciaran Levingston. Usability in agile development. *UX research*, pages 1–7, 2008.

Trenton Schulz and Kristin Skeide Fuglerud. Creating personas with disabilities. In *International Conference on Computers for Handicapped Persons*, pages 145–152. Springer, 2012.

John Pruitt and Jonathan Grudin. Personas: practice and theory. In *Proceedings of the 2003 conference on Designing for user experiences*, pages 1–15. ACM, 2003.

W3C. Reading level:understanding sc 3.1.5. URL https://www.w3.org/TR/UNDERSTANDING-WCAG20/meaning-supplements.html.

Randolph G Bias and Deborah J Mayhew. Cost-justifying usability: An update for the internet age, 2005.

Viktoria Stray, Nils Brede Moe, and Gunnar R. Bergersen. Are daily stand-up meetings valuable? A survey of developers in software teams. In Hubert Baumeister, Horst Lichter, and Matthias Riebisch, editors, *Agile Processes in Software Engineering and Extreme Programming, XP 2017*, pages 274–281, Cham, 2017. Springer International Publishing. ISBN 978-3-319-57633-6. doi: 10.1007/978-3-319-57633-6_20. URL http://dx.doi.org/10.1007/978-3-319-57633-6_20.

V Version one. 11th annual state of agile development survey, 2017.

Kent Beck, Mike Beedle, Arie Van Bennekum, Alistair Cockburn, Ward Cunningham, Martin Fowler, James Grenning, Jim Highsmith, Andrew Hunt, Ron Jeffries, et al. Manifesto for agile software development. 2001.

David Talby, Arie Keren, Orit Hazzan, and Yael Dubinsky. Agile software testing in a large-scale project. *IEEE software*, 23(4):30–37, 2006.

Sergio Luján-Mora and Firas Masri. Integration of web accessibility into agile methods. In *Proceedings of the 14th International Conference on Enterprise Information Systems (ICEIS 2012)*, pages 123–127, 2012.

Udo Kelter, Marc Monecke, and Markus Schild. Do we need agilesoftware development tools? In *Net. ObjectDays: International Conference on Object-Oriented and Internet-Based Technologies, Concepts, and Applications for a Networked World*, pages 412–430. Springer, 2002.

L Constantine. What do users want? engineering usability into software. *Windows tech journal*, 4(12):30–39, 1995.

Jennifer Ferreira, James Noble, and Robert Biddle. Agile development iterations and ui design. In *Agile Conference (AGILE), 2007*, pages 50–58. IEEE, 2007.

Gerard Meszaros and Janice Aston. Adding usability testing to an agile project. In *Agile Conference, 2006*, pages 6–pp. IEEE, 2006.

Kathleen M Eisenhardt. Building theories from case study research. *Academy of management review*, 14(4):532–550, 1989.

Pamela Baxter and Susan Jack. Qualitative case study methodology: Study design and implementation for novice researchers. *The qualitative report*, 13(4):544–559, 2008.

Rosalind Edwards and Janet Holland. *What is qualitative interviewing?* A&C Black, 2013.

William Albert and Thomas Tullis. *Measuring the user experience: collecting, analyzing, and presenting usability metrics*. Newnes, 2013.

Christopher Power, André Freire, Helen Petrie, and David Swallow. Guidelines are only half of the story: accessibility problems encountered by blind users on the web. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 433–442. ACM, 2012.

Arnold M Lund. Measuring usability with the use questionnaire12. *Usability interface*, 8(2):3–6, 2001.