KTH ROYAL INSTITUTE
OF TECHNOLOGY

Degree Project in Computer Engineering

First cycle, 15 credits

Method for Automated Accessibility Testing of Web Application Components (AAT-WAC)

Proposal for a Method for Automated Accessibility Testing of Web Applications Built Using a Component-based Architecture

AUGUST RONNE

Stockholm, Sweden 2024

Method for Automated Accessibility Testing of Web Application Components (AAT-WAC)

AUGUST RONNE

Degree Project in Computer Engineering

Date: 2024-01-28

Supervisor: Mira Kajko-Mattson

Examiner: Ki Wong Sung

School of Electrical Engineering and Computer Science

Host company: Zenon AB

Swedish title: Metod för automatiserad tillgänglighetstestning av webapplikationskomponenter

Abstract

It would be difficult to imagine the world we live in without the World Wide Web. We depend on it for communication, entertainment, transfer of capital, access to essential services, and many other things. Even though it feels like the Web is everywhere, its usage is still growing, and so is its importance. However, a significant portion of the world's population is made up of people with disabilities, and if the Web and its content is not made accessible to them, they cannot participate in this integral part of modern society. Making sure that the resources we access through the Web are accessible to people with disabilities is a difficult task. Those who create applications for the internet need to test them to identify accessibility issues. Today, much of the content on the Web is divided into units called components. It would be advantageous if there was a method for automated accessibility testing of these components. The problem is that no such method exists.

Components—in this context—refer to the parts that make up a webpage. When you are, for example, viewing a news article online, the heading could be a component, as could the comments section, and so on. A majority of the world's webpages are constructed using this architecture, where a set of reusable components with different functionality make up the page you are visiting. The purpose of this thesis is to create a method for automated accessibility testing of these web application components. The goal is that the created method should be useful for web developers and testers in their work to create a Web that is more accessible for people with disabilities, and therefore contribute in some way to a more accessible society.

The chosen research methodology was qualitative and exploratory, and followed the design science research-paradigm. The methodology consisted of four distinct phases, a literature study phase, a preliminary design phase, an evaluation of the preliminary design phase, and an improved design phase. The literature study phase laid the groundwork for creating a method proposal in the preliminary design stage. This method was then evaluated in the evaluation stage. This evaluation consisted of a partial implementation of the framework, together with interviews with respondents with relevant experience. Using this evaluation, an improved method was created in the improved design phase. The result of this thesis is the Automated Accessibility Testing of Web Application Components Method (AAT-WAC). The evaluations that were conducted proved that AAT-WAC method met all of the stipulated evaluation criteria, and that the method was useful when implemented in a real-world industrial context. The literature study proved that no other methods similar to the AAT-WAC method existed.

Keywords

Web Accessibility, Accessibility Testing, Component Testing, Web Development, Software Testing

Sammanfattning

Det skulle vara svårt att föreställa sig världen vi lever i utan Webben. Vi använder den för kommunikation, underhållning, överföring av kapital, tillgång till livsviktiga tjänster och mycket mer. Även om det kan kännas som att Webben redan är överallt så växer fortfarande dess användning och betydelse. En ansenlig del av världens befolkning är personer med funktionsnedsättningar, och om Webben och dess innehåll inte är tillgängligt för dem kan de inte deltaga i denna oumbärliga del av det moderna samhället.

Att säkerställa att de resurser vi tillgodogör oss genom Webben är tillgängliga för personer med funktionsnedsättningar är en utmanande uppgift. De som skapar applikationer för Webben behöver testa dem för att upptäcka tillgänglighetsproblem. Idag är mycket av innehållet på Webben indelad i enheter som kallas för komponenter. Det skulle vara fördelaktigt om det fanns en metod för automatisk tillgänglighetstestning av dessa komponenter. Problemet är att ingen sådan metod existerar.

Komponenter, i den här kontexten, syftar på de enheter som tillsammans utgör en webbsida. När du exempelvis besöker en nyhetsartikel på Webben så kan en komponent utgöra rubriken, en annan kommentarsektion, och så vidare. En majoritet av världens webbsidor är konstruerade enligt denna arkitektur, kallad en komponent-baserad arkitektur, där en uppsättning återanvändbara komponenter med olika funktionalitet utgör webbsidan du besöker.

Syftet med denna avhandling är att skapa en metod för automatiserad tillgänglighetstestning av webbapplikationskomponenter. Målet är att öka kunskapen om tillgänglighetstestning, och genom detta skapa ett mer tillgängligt samhälle.

Forskningsmetodologin var kvalitativ och följde en Design Science-paradigm. Forskningsmetodologin bestod av fyra distinkta faser, en litteraturstudie-fas, en preliminär design-fas, en utvärderings av preliminär design-fas och en förbättrad design-fas. Litteraturstudien lade grunden för att skapa en preliminär metod i preliminär design-fasen. Denna preliminära metod utvärderades sedan genom att implementera delar av metoden på en verklig webapplikation, samt genom intervjuer med personer med relevant expertis inom webbutveckling. Dessa utvärderingar användes sedan för att skapa den förbättrade metoden i förbättrad design-fasen.

Resultatet av denna avhandling är metoden Automated Accessibility Testing of Web Application Components (AAT-WAC-metoden). Uvärderingarna av AAT-WAC-metoden påvisade att metoden uppfyllde alla utvärderingskriterier som stipulerats, samt att metoden var användbar när den implementerades i en verklig, industriell kontext. Litteraturstudien påvisade att inga metoder som liknar AAT-WAC-metoden existerade.

Nyckelord

Webbtillgänglighet, Digital tillgänglighet, Komponenttestning, Webbutveckling, Programvarutestning

Acknowledgments

I would like to extend my sincere thanks to Zenon AB and all of the people working in their Stockholm office. I have felt nothing but support since my first meeting with them, and I am truly grateful for their help in facilitating a project in which I could learn so much about something I find deeply important. A special thanks to the employees of Zenon who let me interview them about my research, sacrificing valuable time from their busy schedule in order to help me.

I would also like to thank the KTH Royal Institute of Technology for making this thesis possible. All of the teachers I encountered helped in some way in giving me the knowledge needed for this project. A heartfelt thank you to Leif Lindbäck who was my supervisor for my first attempt at writing a thesis, which I had to suspend. Without his support back then I would not have returned to KTH to finish my degree.

Finally, I want to convey my deepest appreciation to my two supervisors. Associate Professor Mira Kajko-Mattson of KTH for her invaluable advice throughout the whole process, while trying to turn my ramblings into something understandable. Isabel Ghourchian of Zenon AB for answering all of my questions without ever making me feel like an inconvenience, and for always making time for me. Without her help this thesis would not have been completed.

Tillägnad min familj. Alltid.

Stockholm, December
August Ronne

Table of contents

List of Figures

List of Tables

List of acronyms and abbreviations

AAT-WAC Method	Automated Accessibility Testing for Web Application Components Method
CSS	Cascading Style Sheets
HTML	HyperText Markup Language
ICT	Information and Communication Technology
The Web	The World Wide Web

1 Introduction

Web accessibility aims to ensure that no person is hindered from accessing and interacting with web applications, sites, technologies, and tools. In order to achieve this accessibility, people with disabilities need to be considered in the design and development process of the content that is available on the web. There are many categories of disabilities that need to be considered. Take visual impairments, for example. How does a website need to be designed so that a person that cannot see can comprehend its content? Solving such questions is one example of increasing web accessibility. Making resources accessible to people with disabilities also benefits other groups, such as those who have had certain capabilities reduced by age, or people with temporary disabilities [1].

Web accessibility is growing in importance for several reasons, and there is a great interest among many different types of organizations in increasing the accessibility of their web applications. The first—and most important—reason is one agreed upon by many people, namely that making an important part of society more accessible is a question of ethics, meaning that improving accessibility is the right thing to do, and that poor accessibility is a form of discrimination [2, p. 7].

While Sweden does not keep an official register of the number of citizens with disabilities, the percentage of people with disabilities can reasonably be put at between 10 to 30 percent of the Swedish population [3]. Worldwide numbers are similarly hard to pinpoint, but a reasonable estimate is around 16 percent [4]. Removing barriers of usage to sometimes essential services to such a large number of people is to improve society. Depending on the location where web-based content is registered, and the purpose of the content, there can also be legal consequences to making such content inaccessible. The considerable amount of people with disabilities also make web accessibility important to companies for business reasons [5, p. 33].

Although there are more than one group working to build a more accessible web, the term *web accessibility*, as it used to today, most often refers to the accessibility guidelines created by the World Wide Web Consortium (W3C) [6]. These guidelines, named the Web Content Accessibility Guidelines (WCAG) are a comprehensive set of non-prescriptive requirements that, if fulfilled, will improve a web application's accessibility.

Despite web accessibility becoming increasingly relevant, it is still sorely lacking. In 2023, the non-profit organization WebAIM found that approximately 98 percent of the world's 1 million most popular websites were not compliant with WCAG 2.1, and all these websites are could thus be deemed inaccessible to people with disabilities [7].

1.1 Background

When a web application is being evaluated by how well it meets a set of accessibility requirements, it is being accessibility tested. Therefore, accessibility testing can be described as a process aimed at figuring out how well a web application can be used by people with disabilities. Accessibility testing encompasses an extensive range of skills, technologies, and disciplines. It may include different technologies, testing software, automated tests, manual tests, testers brought in from outside the organization, and much more. What differentiates accessibility testing from many other similar testing processes in web development is that it must involve a great deal of manual testing in order to meet the criteria it is evaluated by [8, p. 480].

Accessibility testing a web application is no easy task. The WCAG guidelines are hard to work with. The language they are written in is regulatory-like and therefore difficult to understand. The currently most used WCAG standard, the 2.1 version, includes a section named *Understanding WCAG 2.1* that is almost as extensive as the guidelines themselves [2, pp. 2–4]. It is also important to note that the guidelines are non-prescriptive, meaning they do not tell you *how* you should solve different accessibility issues, only that you *should* solve them. Furthermore, even if you are well-acquainted with WCAG, many cases require you to be knowledgeable about assistive technologies, and how people with different disabilities use them. This knowledge can be very hard to acquire. If you also consider that accessibility testing is technologically specific and needs to change if, for example, a certain assistive technology changes, one can understand the difficulty of accessibility testing.

Considering these difficulties, the concepts of automated tests—either imported or self-written—for evaluating accessibility have always been an enticing one. There are many automated accessibility testing services available. These automated services cannot catch every accessibility violation, but they can often make for a good start [8, p. 483].

When it comes to self-written accessibility tests, the options are less well-understood. However, one possible solution, called component testing, has gained traction fairly recently. It is important to understand what is meant by component in this context, as the concept of a component is not only used in many other areas of software development, but also other industries altogether. In the context of web development, component refers to the individual parts that make up a whole web page. When you are viewing a web page, what you are seeing is a representation of code, written by developers. This code can be divided into modular, reusable units with differing functionalities, and these units are called components. How to divide web content into components is up to the developers [9].

If a web application is built using components, it is said to use a *component-based architecture* [10]. When tests are written for these individual components, it gives the test writer more control over what is being tested, which also makes testing more flexible. This is because a website often changes when you interact with it, and these changes are more difficult to simulate when testing a whole web page instead of just testing a single component.

Web content accessibility is never complete, and the efforts to achieve it do not come to an end. A website cannot be said to be definitely accessible. Accessibility is a continual process and will need to be constantly reassessed. Technologies change, organizations change, things break, and new functionality is added to the internet [5, p. 33].

1.2 Problem

The problem addressed by this thesis is that—according to the author's research—there exists no method for automated accessibility testing of web application components. As can be seen in Appendix A – *Problem Verification Sub-phase*, no method similar to the method created for this thesis, namely the Automated Accessibility Testing for Web Application Components (AAT-WAC) method, could be found. The availability of a properly evaluated method would help with the difficulties discussed in Section 1.1.

1.3 Purpose

The purpose of this thesis is to create and evaluate a method for automated accessibility testing of web application components. This method, called the AAT-WAC method, provides guidelines and structure for implementing such testing, while still retaining flexibility and adaptability so that it can be useful in many different contexts.

1.4 Goals

The main goal of this thesis is that the method that was created should help people interested in web content accessibility in creating a more accessible internet. As the most direct sub-goal, web developers and testers should be able to use the method to increase the accessibility of the web applications they are working on. A second significant sub-goal is that the discoveries that were made when creating and evaluating the method should also provide useful knowledge to educators, students, and anyone interested in web content accessibility. Finally, the third sub-goal is that both the method itself and the findings that were made should serve as a basis for future research in the relevant areas.

1.5 Research Methodology

Considering the nature of research purpose and the necessary limitations that would need to be imposed, the research presented in this report was based on exploratory research design in combination with a qualitative research methodology. As a research paradigm, design science was chosen [11].

Qualitative research is an umbrella term encompassing different research methodologies where data can be gathered through, for example, interviews, case studies, and researcher observations. It was well-suited to this research as it focuses on evaluation and analysis that is continually expanded throughout the process [12].

Exploratory research design was chosen as it focuses not on conclusive evidence but on greater understanding of the problem. It also allows for flexibility, adaptability, and being able to lay the groundwork for future studies even though the research findings are not clearly quantifiable [13]. This methodology design resulted in the decision to divide the research process into four distinct phases, a literature study phase, a preliminary design phase, an evaluation of the preliminary design phase, and an improved design phase.

In the literature study phase an extensive literature study was conducted. This literature study revealed that there was no such method as the one created and evaluated in this thesis. It also provided the necessary knowledge for designing and evaluating methods.

In the evaluation of the preliminary design phase, the preliminary method was evaluated using two different techniques. The first part of the evaluation consisted of a partial implementation of the method within Zenon AB, which is the commissioning company. Using the preliminary method, the components of one of Zenon AB's real-world web applications were accessibility tested. This partial implementation was evaluated by its conformance to evaluation criteria defined by the researcher. The second part of the evaluation consisted of interviewing web developers of Zenon AB who had worked on the real-world web application and collecting their feedback on the method as a whole.

1.6 Commissioned Work

Zenon AB is an information technology consulting firm with offices in Linköping and Stockholm, Sweden. Their consultants work in a variety of areas, such as web development, integration, database solutions, embedded software, and much more.

Blixtvakt (https://blixtvakt.com) is a web application developed and owned by Zenon AB. The application tracks lightning strikes on an interactive map by displaying them using graphical symbols. The application has other functionality as well, but information on lightning strikes is its main concern. The application is very successful and is currently the most popular lightning strike-tracking application in Sweden [14].

The Blixtvakt application uses a component-based architecture, but during and after the development of Blixtvakt, no structured accessibility testing of the application's components was used. The application design has taken web accessibility into account, but Zenon AB was interested in finding out what component testing practices focused on accessibility could be useful both for the Blixtvakt web application and for their company.

The work the author and Zenon AB agreed upon was for the author to build a prototype of the Blixtvakt web application and implement a component testing-focused accessibility testing on that prototype. This work could then be used both by the commissioning company to learn more about accessibility testing their application, as well as being used by the researcher to evaluate the testing method.

1.7 Target Audience

The target audience for this thesis is anyone interested in or involved with accessible web development. While the increased knowledge of component testing for accessibility issues itself is meant to be benefit web developers, people that interact with developers, either through work, school, or any other of the many possibilities, are also the target of this thesis. As the research methodology is based on work taking place in a professional setting, this will invariably affect the results. However, it is the intended goal that the results of the thesis are also of interest to hobby developers, students, or teachers.

1.8 Scope and Limitations

This thesis attempts to increase the knowledge of component testing for accessibility issues. If the scope and limitations for such an endeavor are not properly defined, the effort can become absolutely colossal. Therefore, a number of limitations needed to be defined. These limitations were produced by taking into consideration some important factors limiting the potential of the study. Most importantly, the time available to the author, the author's knowledge in the relevant areas, and the resources available to the author needed to be taken into account. These parameters, time, knowledge, resources, are also relevant when it comes to the other stakeholders, Zenon AB, and the KTH Royal Institute of Technology. Considering these factors, the following limitations were defined: (1) *Technologies*, (2) *Blixtvakt Application*, (3) *Web Content Accessibility Guidelines (WCAG)*, (4) *Zenon AB's Internal Infrastructure*, and (5) *Mobile Web Applications*.

- **Technologies:** The evaluation of component testing for accessibility issues focuses on the usefulness of this approach as a concept. There are too many web development frameworks using a component-based architecture, and too many frontend testing frameworks for the author to have time to compare the strengths of different technologies.
- **Blixtvakt Application:** The author does not have the knowledge or the time to fully grasp all the infrastructure of the Blixtvakt application in full. Therefore, a prototype application is developed, that matches the real application exactly when it comes to visual output. If this is achieved, accessibility testing is not affected at all.
- **WCAG:** Completely evaluating the web application according to all the WCAG criteria is too extensive a task for this study. If the accessibility testing implemented using component testing seems promising to Zenon's developers is what is interesting to Zenon AB, not an attempt at a complete WCAG certification of an enterprise application.
- **Zenon AB's Internal Infrastructure:** Zenon AB has their own processes for working with continuous deployment and integration of web applications. The author does not have the knowledge or the time to propose solutions for how to integrate the non-manual tests into the company workflow. If aspects of the accessibility testing implemented by the author are interesting to Zenon AB, they will incorporate these aspects themselves.

- **Mobile Web Applications:** Most resources aimed at helping developers with web accessibility—such as standards, guides, and specifications—focus on web content displayed on a computer screen. This is because web accessibility is much harder to achieve on smaller screens, such as smartphones and other mobile devices. Therefore, many people with disabilities prefer to use computer screens, or are forced to. Because of this fact, this thesis only evaluates the accessibility of Blixtvakt as it is presented on larger screens. However, web content accessibility on mobile devices is one of the most interesting areas of research, so if the resources available to the researcher were greater, this definitely should have been included in the research process.

1.9 Benefits, Ethics, and Sustainability

This section discusses the beneficial, ethical, and sustainability effects of the research conducted in this study. Section 1.9.1 *Social Benefits* describes the potential negative and positive social effects of the thesis. Section 1.9.2 *Ethics* describes the aspects that needed to be considered from an ethical standpoint when conducting the research. Last, section 1.9.3 *Sustainability* describes the conceivable impact on sustainability that this research might have.

1.9.1 Social Benefits

The main social effect that this research has the possibility to achieve is to encourage organizations or individual developers to take the first steps in improving accessibility of some web application. The benefits of this social effect are plenty, but they can summarized as increasing web accessibility, and thus increasing how accessible society is. A more accessible society is beneficial to all of us as it can increase the enjoyment and ease of participating in it for both ourselves and people that are important to us.

The social effect can also have negative consequences. If the discussion and analysis of the weaknesses of component testing for accessibility issues is lacking, readers could take the wrong knowledge away from this thesis. This could result component testing being used in the wrong way, and therefore hindering a better solution being used, or making developers feel that accessibility testing is too difficult and not worth the effort.

1.9.2 Ethics

Throughout the degree project the IEEE code of ethics was used to attempt to make sure all work was conducted in an ethical manner [15]. This included not only the research itself, but also communication with Zenon AB and the KTH Royal Institute of Technology.

The commissioned work at the center of this thesis is based on software belonging to Zenon AB being shared with the researcher. The researcher and the software owner have signed a contract regarding what the researcher is allowed to use the software for, and what parts of the research can be shared with others. Disregarding the obvious legal ramifications, it is important not to breach this contract for ethical reasons. The research also necessitated continual transfer of company information from Zenon AB to the researcher, and the researcher needed to use personal judgement to handle and use that information in an ethical manner.

The thesis is based on a qualitative research methodology, and the methodology in question was heavily reliant on external sources, such as past research and relevant literature. It is therefore an important ethical consideration to make sure these sources are credited, which is done by the researcher making the best effort to reference these sources throughout the thesis, in combination with this referencing being evaluated by the thesis' examiner and supervisor. Other important ethical considerations required when conducting qualitative research were also adhered to [16].

1.9.3 Sustainability

The main sustainability benefit of the results of the research is to contribute to a greater understanding of accessibility testing web applications, and in doing so helping organizations and developers to implement accessibility testing more effectively during development of their web applications. In the long run, this will save on resources as a web application lacking accessibility testing will need to be addressed at a later stage, where such modifications are more cumbersome.

1.10 Structure of the thesis

The thesis has been structured into the following chapters, which are presented here together with a short summary of their contents:

- *Chapter 2 – Extended Background:* This chapter presents an overview of the theoretical background required to understand the thesis, along with describing the main technologies used in the research. This chapter is a foundation for the thesis research methodology.
- *Chapter 3 – Research Methodology:* This chapter presents the research methodology that was used. This includes validity issues of the research process, the theoretical motivations for developing the chosen research process, as well as the parts making up the research process itself.
- *Chapter 4 – Evaluation Models:* This chapter presents the evaluation models that were used when evaluating the preliminary and improved AAT-WAC methods.
- *Chapter 5 – Overview of the Preliminary AAT-WAC Method:* This chapter presents the preliminary AAT-WAC method that was created as a result of conducting the *Preliminary Design* Research Phase.
- *Chapter 6 – Partial Implementation and Evaluation of the Preliminary AAT-WAC Method within Zenon AB :* This chapter presents the results of implementing parts of the AAT-WAC method to implement automated accessibility testing of Zenon AB's web application *Blixtvakt*
- *Chapter 7 – Evaluation of the Preliminary AAT-WAC Method Using Interviews:* This chapter presents the results of evaluating the preliminary AAT-WAC method by interviewing three Zenon AB employees with relevant expertise in web development
- *Chapter 8 – Improved AAT-WAC Method:* This chapter presents the results of using the feedback gathering in Chapter 6 and Chapter 7 to create an improved AAT-WAC method. The chapter also includes an overview of the evaluation of the improved method.
- *Chapter 9 – Conclusions and Future Work:* In this chapter the conclusions drawn from the research results are presented, alongside suggestions of future work.

2 Web Content Accessibility and Web Application Testing

This chapter presents the theoretical and practical background that is required to understand the rest of this thesis. As the chapter addresses many different areas, the structure of the chapter is first explained using a scenario. This scenario, where a user follows the AAT-WAC method, is presented in Section 2.1. After reading this section, the structure of the rest of the chapter should be more understandable.

Following the introductory scenario section, Section 2.2 outlines what is meant by web content accessibility evaluation, presents the dominant accessibility standard by which web content is evaluated, as well as describing ARIA and introducing the ARIA APG. In Section 2.3, the reasons for the dominance of component-based architecture in modern web development are explained by going through the JavaScript programming language and some of the most popular web application frameworks. Section 2.4 presents automated testing together with some of its most important benefits, the different types that automated tests are often divided into and describes the automated test type called component testing. Last, Section 2.5 presents the major related works that were chosen to serve as the foundation for the AAT-WAC method.

2.1 Explanatory AAT-WAC Method Usage Scenario

To illustrate the areas that need explaining, a scenario based on using the AAT-WAC method will be used. In this scenario, the web application you developed needs automated accessibility testing. The AAT-WAC method instructs you to make sure that the application uses a component-based architecture, to read up on accessibility standards, and choose your testing tools and technologies. When these steps are completed, the method states that you should use accessibility requirements from the *Accessibility Rich Internet Applications* (ARIA) *Authoring Practices Guide* (APG) to write accessibility component tests for each of the application's components.

Reading through this scenario reveals all major areas that this chapter needs to explain in order for the reader to understand the thesis: (1) *Web Content Accessibility Standards*, (2) *Component-based Architecture*, (3) *Tools and Techniques for Testing*, (4) *What is meant by component testing*, and (5) *What is the ARIA APG?*

2.2 Web Content Accessibility

This section describes the areas of web content accessibility relevant to this thesis. Section 2.1.1 explains what is meant by web content accessibility evaluation by describing concepts such as web content and conformance evaluation. Section 2.1.2 describes the structure and contents of the most used accessibility standard today, WCAG. Section 2.1.3 presents the suite of web content accessibility standards known as ARIA. Then, Section 2.1.4 explains the most important differences between WCAG and ARIA APG. Finally, Subsection 2.1.5 gives a brief description of the most common testing methods used when evaluating web content accessibility by some standard, together with a limited presentation of some available methodologies.

2.2.1 Web Content Accessibility Evaluation

When evaluating web accessibility, the actor or actors performing the evaluation are attempting to determine how well some web content conforms to a set of chosen accessibility standards [17]. Web accessibility evaluation can also be called web accessibility testing, or auditing, but since W3C uses evaluation in its official documentation, this term is also used in this thesis.

In order to understand the description above, the meaning of web content needs to be explained. The World Wide Web, or the Web, is not the same thing as the internet. The Web is a set of open standards that define the formats and protocols required for a system of public webpages to be accessible through use of the internet. Web content, in turn, is anything provided by the web that its users can perceive or interact with. This includes, but is not limited to, text, images, videos, and sounds. Importantly, web content does not only refer to what the user perceives, but also the code used to define and style the content [8, pp. 225–227].

Furthermore, it is important to understand is that web accessibility evaluation if a type of conformance evaluation. This means that the web content is being evaluated by how it confirms to a set of accessibility standards, and that these set of standards must be decided on before the evaluation begins [17]. Today, the web content accessibility standard that is completely dominant is WCAG 2.1, which is developed and maintained by the W3C Web Accessibility Initiative (WAI). The current WCAG version is approved as an International Organization for Standardization (ISO) standard [8, pp. 231–232].

2.2.2 WCAG

WCAG is an accessibility standard consisting of a set of guidelines aimed at making web content more accessible. Conformance to WCAG is divided into three levels, *A, AA,* and *AAA*, where level *A* is the least strict. The recommendations and requirements found in WCAG attempts to take a wide variety of disabilities into consideration, but willingly admits that not every user need of people with these disabilities can be accommodated. Since several different technologies—such as desktops, laptops, tablets, and mobile devices—can deliver web content to users, the guidelines take all these technologies into account [18, pp. 27–29].

The structure of WCAG is called layers of guidance. There are four layers, and they are defined in the following, hierarchical fashion :

- **Principles:** There are four principles which form the foundation on which the rest of WCAG is built, and they constitute the top layer of guidance. These principles are listed below, together with their definition, as published by WAI [6]:
    1. *Perceivable:* Information and user interface components must be presentable to users in ways they can perceive.
    2. *Operable:* User interface components and navigation must be operable.
    3. *Understandable:* Information and the operation of user interfaces must be understandable.
    4. *Robust:* Content must be robust enough that it can be interpreted reliably by a wide variety of user agents, including assistive technologies.
- **Guidelines:** Belonging to each principle is a set of guidelines. These guidelines are the second layer of guidance. Crucially, the guidelines are not testable, but rather provide goals that developers should strive to achieve. The current version of WCAG, 2.1, contains 13 guidelines in total [6] .
- **Success Criteria:** For each guideline, there is a set of success criteria. These success criteria constitute the third layer of guidance. These criteria outline exactly what the web content needs to achieve in order to conform to the standard. These success criteria are testable, but importantly, non-prescriptive. This means that evaluators should be able to check that a success criteria has been confirmed to, but that the criteria should be technology neutral. The success criteria tell you what you solve, not how you should

solve it. Each success criteria belongs to one of the three conformance levels, *A, AA,* and *AAA* [6].

- **Sufficient and Advisory Techniques:** Lastly, the fourth layer of guidance consists of sufficient and advisory techniques, where each technique belongs to a success criteria. These techniques consist of code examples, tests, and other resources. The techniques are informative, and not required. Evaluating conformance to WCAG is done by the success criteria, while the techniques are meant to help with understanding and solving them [6].

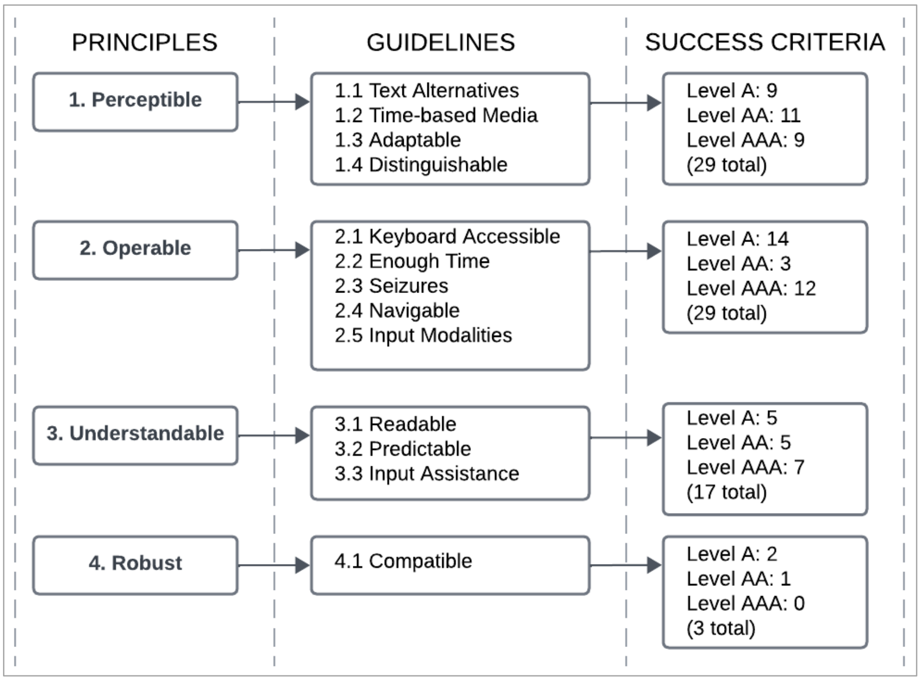

Figure 2-1: Diagram of the structure of WCAG 2.1

The current version of the standard, WCAG 2.1, was released in 2018. Figure 2-1 shows an overview of the top three layers of guidance of WCAG 2.1. WCAG versions are fully backwards compatible with each other, and as such web content that conforms to WCAG 2.1 confirms to version 2.0 as well. The WCAG conformance level that most organizations seek to achieve is *AA*. It is worth noting that level *AAA* cannot always be met, as some of its success criteria are not relevant to all web content [18, pp. 27–29].

Lastly, the WCAG 2.1 standard includes a number of supporting documents meant to help evaluators on how to understand and conform to WCAG 2.1. Two of the most important of these supporting documents are [6]:

- **How to Meet WCAG 2.1:** A quick reference document aimed at helping developers evaluate their web content. Can be customized to quickly find relevant success criteria and their corresponding techniques.
- **Understanding WCAG 2.1:** A guide to understanding the standard. Includes explanatory sections on every principle, guideline, and success criteria.

2.2.3 Accessibility Initiative Accessible Rich Applications (ARIA)

This section presents ARIA, which is a suite of web content accessibility standards. First, section 2.1.3.1 provides an introduction to ARIA, its reason for existence, and the organization responsible for its creation and maintenance. Second, section 2.1.3.2 describes the ARIA Authoring Practices Guide (ARIA APG), which is a guide that plays a fundamental role in the AAT-WAC method.

2.2.3.1 Introduction to ARIA

ARIA provides a framework for making web content more accessible by combining a suite of different web content accessibility standards. Just as with WCAG, ARIA is maintained by WAI. It is clearer in its instructions on how to solve web content accessibility issues than WCAG, as it is specifically aimed at providing help with dynamic web content and advanced user interface entities developed using JavaScript, HTML, and related technologies [19]. WCAG is easier to understand than ARIA for people without experience in web development or web technologies. However, since ARIA presupposes a higher degree of familiarity with the subject than WCAG, web developers often find ARIA easier to work with than WCAG [20].

It is important to remember that in the end, ARIA is still based on WCAG. To illustrate this with a concrete example, ARIA provides more hands-on directions on how the web content navigation using a keyboard should be organized. However, when the ARIA solutions have been implemented, WCAG still needs to be consulted to check that all the WCAG requirements related to keyboard navigation have been met.

2.2.3.2 ARIA Authoring Practices Guide (ARIA APG)

The ARIA APG is a guide aimed at providing web application developers and testers with guidance in developing interactive web content [21]. WCAG's requirements are non-prescriptive—meaning that they do not provide an answer for how the requirement should be met—while ARIA APG provides a set of techniques aimed at meeting the requirement. The ARIA APG has synthesized solutions and requirements from many of these techniques in order to create a comprehensive guidance for how to develop interactive web content so that it is accessible.

It is the ARIA APG's stated intention to help web developers and testers. The guide is not understandable to users without the required experience in web development. Unlike WCAG and ARIA, the ARIA APG is an informative standard, not a normative one. This means that conformance evaluation to the ARIA APG cannot be conducted [21]. When the ARIA APG is followed during web content development, many of the most important requirements in WCAG are automatically met, but accessibility evaluation still has to be carried out in order to identify *exactly* which WCAG requirements have been met, and which have not been met.

2.2.4 Differences Between WCAG and the ARIA APG

It is of utmost importance that the differences between a standard like WCAG and a set of guidelines like the ARIA APG is explained if the results of this thesis are to be presented in a responsible way. WCAG is a normative standard aimed at providing a way to perform a comprehensive and definite accessibility evaluation of the chosen set of web content. The WCAG requirements are intentionally broad, as they must be usable by all sorts of technologies and in many different environments [6]. The ARIA APG, on the other hand, is not a standard, nor even a specification. It is a set of guidelines that provides support for making a very specific type of web content more accessible. When using the ARIA APG, web developers have a huge responsibility in making sure that the solutions they implement by following the guidelines do not *damage* accessibility instead of increasing it. Furthermore, the ARIA APG is *not* the same thing as ARIA. ARIA is a specification, which provides a framework for making web components more accessible, which means that web content can be evaluated in order to determine its conformance to ARIA, something which can not be done with the ARIA APG [22].

2.2.5 Web Content Accessibility Evaluation Methods

Web accessibility evaluation is comprised of an extensive range of disciplines and skills. Evaluators need comprehensive knowledge of both technical and non-technical aspects related to web technologies and web accessibility. Because of this, evaluation requires a diverse set of methods. Although there is more than one way to categorize these methods, they are usually divided into three categories: (1) *automated accessibility testing*, (2) *manual accessibility testing*, and (3) *user accessibility testing* [8, p. 480]:

- **Automated Accessibility Testing:** With automated testing, separate software—running either locally or online—is used to execute tests on the web content. The tests can either be imported or written by the evaluators themselves.
- **Manual Accessibility Testing:** Also called manual inspection. This method consists of evaluators knowledgeable in web accessibility manually testing the web content in order to evaluate if it conforms to the chosen accessibility standard.
- **User Accessibility Testing:** This method consists of informal usability tests of the web content. The tests are often, but not always, conducted by people with disabilities. User testers supply feedback to the evaluators through a process that has been defined by the evaluators.

2.3 Web Applications and Component-based Architecture

This section attempts to explain what component-based architecture means in the context of web applications, and why this architectural solution is so popular on the modern Web. In Section 2.2.1 the dominance of the JavaScript programming language in modern web development, together with some of the most popular JavaScript web application frameworks, is presented. Second, Section 2.2.2 describes the characteristics of the component-based architecture for web applications, and some of its advantages.

2.3.1 JavaScript and Web Application Frameworks

Deciding on the programming language to use for a web application is an important architectural decision, as the architecture describes not only the design patterns, but also the techniques used when building an application. Today, JavaScript is the preeminent programming language in web development. It is estimated that 98.8% of websites use JavaScript as the client-side language running in the browser [23].

Table 2-1: Definition of Software Framework

Software Framework: Reusable software environment that enables a standard process of building and deploying software applications. The software in the framework provides generic functionality that can be selectively altered or removed [24].

The popularity of JavaScript has led to the need for JavaScript web application frameworks. Web application frameworks are software frameworks that help with application development. The term software framework is often misunderstood, and therefore a definition is found in Table 2-1. The most popular JavaScript web applications frameworks are Angular, React, and Vue.js. While there are many important differences between these frameworks—they, together with many other web application frameworks—share one important characteristic, they employ what is called a *component-based architecture* [25].

2.3.2 Component-based Architecture

When you are viewing and interacting with web content, that content has been created by several kinds of code. HTML to structure it, CSS to style it, and, for example, JavaScript to give it interactive behavior. When this code—and the content it defines—is divided into modular, independent, and reusable components, the web application has organized the web content using a component-based architecture [26, pp. 3–5].

In this thesis, the usage of the term component refers to the web content units described above. However, the term component can describe many different things depending on which software architecture context it is used in. In some systems, a whole library, or a single function, may be regarded as a component, depending on what the developer has decided [27].

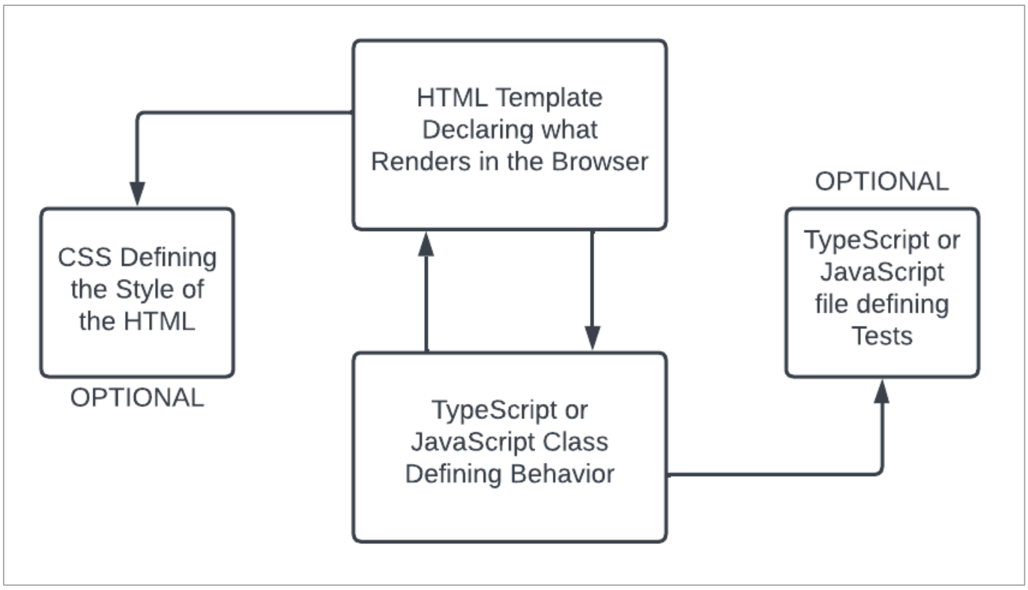

Figure 2-2: Diagram of the component architecture used by the Angular web application framework

Presently, a majority of the most popular web application frameworks make heavy use of components [25]. This is because the components independence and reusability are of great use in web development. The component independence means it can be mounted on its own, and therefore tested in its entirety without running the whole web application. Its reusability means that it can be used in different places in the web application without needing to rewrite its functionality. Figure 2-2 shows the component architecture used by the Angular web application framework. The component architecture used by the React and Vue.js frameworks is fairly similar to the component architecture Angular employs.

2.4 Testing Web Applications

The theory and practices of testing web applications is an extensive subject, and due to limited resources only the parts most relevant to this thesis are discussed here. Section 2.3.1 explains what automated testing is and presents some of its benefits. Section 2.3.2 describes one popular categorization of automated tests, wherein they are categorized into different types. Last, Section

2.3.3 illustrates how component tests are different from other test types and points out why there is a growing interest in using component tests when testing web applications.

2.4.1 Automated Testing

Manually testing complex software systems is repetitive, time-consuming, and likely to involve human errors. If the system is complex enough, it may also be impossible to achieve with the resources at hand. This introduces the need for automated testing. With automated testing, software is responsible for executing the tests. The software responsible for executing the tests can be built specifically for a certain project or system, but most often established test automation software frameworks are used instead [28, p. 10].

The benefits of automated testing include reducing mistakes in highly repetitive testing processes, increasing test coverage, increasing test productivity, and reducing testing costs. Automated testing cannot fully replace manual testing of web applications, but the resources saved make it easier to perform the manual testing that is necessary [28, pp. 11–15].

In order to effectively implement automated testing, a test automation software framework should be used. These software frameworks help testers with standardizing tests, reusing code, techniques for data handling, and reducing the need for human intervention [29]. In 2022, the most popular test automation software frameworks for JavaScript web applications were Jest, Storybook, and Cypress [30].

2.4.2 Types of Automated Tests

When implementing automated testing of a web application, the tests created can be categorized into different types. There are many different test categorizations available, which leads to the definitions of different test types not always being universally agreed upon. However, the categorization presented in this section is widely used, and its most important characteristics are shared with other popular categorizations.

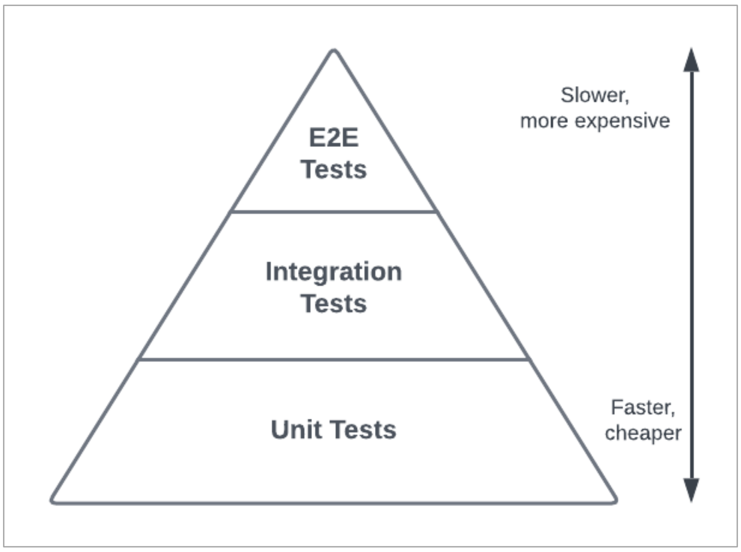

Figure 2-3: Test pyramid with three different types of automated tests (inspired by Figure 1.2 on page 9 of [31])

This categorization, which is summarized in Figure 2-3, divides automated web application tests into three types, unit tests, integration tests, and end-to-end (E2E) tests. These tests serve different purposes, and a brief description of each test type is presented here:

- **Unit Tests:** A unit is the smallest testable part of the software being tested, and the unit tests the basic functionality of such parts. Only one responsibility of the source code should be tested by a unit test. Typical examples of what can be classified as a unit in a web application is a single function, class, type, or method. Unit tests are meant to be fast and reliable.
- **Integration Tests:** Tests whether independently developed modules of the software still work when they interact with or are connected to each other. The scope of integration tests can vary widely as the simplest cases test the integration of two parts of the system, while some tests need to involve many more parts. A typical example of a web application integration test is testing the interaction between a part of the system that presents data, and another part that handles connecting to the database.
- **E2E Tests:** These tests verify the functionality of the web application as a whole by simulating real scenarios where users interact with the application. An example of an E2E test is attempting to log in to a web application with correct credentials, and verifying that the user is logged in. In such a test, multiple parts of the web application are involved, but the E2E test is not concerned with how these parts interact, only that the input data leads to the correct result. E2E tests are typically the slowest of tests listed here and may miss internal problems in the application code as they are only concerned with the end results.

A fourth type of automated web application test is growing in popularity, this test type is called *component test*. In component testing, the test scope is limited to one single component, as defined by the tester. The tests deliberately ignore the parts of the web application outside the chosen component, and only test the functionality that the component should implement.

2.4.3 Why the Need for Component Tests

When speaking on major trends in web application testing, Andrew Knight, the principal architect at the test automation company Cycle Labs, noted that “Component testing is on the rise, because components *themselves* are on the rise” [32]. With the most popular web application frameworks employing a component-based architecture, and many organizations building their own shareable component libraries, it is only natural that there would be a need for tools helping with automated testing of such components.

Component testing is different from unit testing. Unit tests interact directly with code, calling a function or a method and verifying the outcome. A component is inherently visual and needs to be rendered in a browser for complete testing of its functionality. Components often have multiple behaviors, may need to connect to external resources, but are still able to be tested in isolation from other components.

This increased need of tools for component testing has meant that test automation software frameworks have begun implementing support for component testing. For example, the popular test automation software framework Cypress released the first version of their framework with support for component testing in June of 2022 [33]. This increase in support for component testing in popular test automation tools is important as developers previously needed to cobble together their own component testing software frameworks, which was very inefficient.

2.5 Related Works

This section presents the major related works that were used to create the preliminary AAT-WAC method. While many more sources were consulted in order to form a comprehensive knowledge base, the related works described here form the backbone of the methods that were created. First, section

2.4.1 describes the web application strategy that formed as the foundation for the different phases and sub-phases of the preliminary AAT-WAC method. Then, section 2.4.2 details the WAI-ARIA APG patterns, a specification that is used by the method in order to produce accessibility test cases. Last, Section 2.4.3 presents the third major related work, which is the most popular web content accessibility evaluation methodology.

2.5.1 Kinsbruner Web Application Testing Strategy

The first major related work important to the background of this thesis is E. Kinsbruner's web application testing strategy [34]. Several publications pertaining to methods for web development test planning and testing strategies were read. Through this reading of relevant publications, it became clear that many testing strategies and methods shared important qualities. The decision was made to use Kinsbruner's strategy as it solely focused on testing the parts of web applications responsible for content visible to web users, together with the fact that the strategy is adaptable and takes the test team's attributes into account, which are important qualities to the AAT-WAC method.

The testing strategy cannot be presented in its entirety here, but it is based around six stages, called key pillars. These stages, as defined by Kinsbruner, are presented here [34, Ch. 6]:

1. **Know your target users:** Identify and analyze the target users relevant to the testing you are implementing. What do the attributes of the target users affect when it comes to planning the tests. This stage can help define the scope of the testing plan, make testing activities more efficient, and inform testers of additional knowledge they need to acquire before continuing with the strategy stages.
2. **Building a test plan:** In this stage a test plan that covers web application workflows, target user interactions, dependencies, and scope is created. The plan should also cover the relevant test types and define important limitations of the rest of the testing strategy stages.
3. **Preparing tool stack and environments:** This stage consists of identifying and making sure the testing team has access to the needed tools and environments to achieve the goals defined in the test plan. These tools include, but are not limited to, test automation software frameworks, development environments, deployment environments, and relevant documentation.
4. **Set quality criteria and objectives:** Define the necessary metrics so the testing team knows when one iteration of the testing strategy has been completed in full. These metrics may include test coverage, categorization of discovered defects, functional and non-functional criteria, and more.
5. **Build a timeline and schedule:** Construct a timeline and schedule based on the available resources in the form of time, testers, tools, environments, and the previous testing strategy stages.
6. **Execute, monitor, measure, and document:** Execute implementation of the testing using the test plan together with the chosen tool stack and environments. Monitor and measure the progress by documenting what is done and then evaluate what has been documented using the metrics defined in the *set quality criteria and objectives* stage.

2.5.2 ARIA APG Patterns

The ARIA APG, which is presented in section 2.1.3.2 earlier in this chapter, provides guidance for web developers and testers in making web content more accessible. While the guide contains many specifications and tools that were used to gain the required theoretical knowledge in order to write this thesis, it is the ARIA APG patterns that were chosen as a major related work from which the AAT-WAC method was created.

The ARIA APG patterns consist of a set of design patterns that provide clear, concrete accessibility requirements for the different kinds of categories that web content can be divided into [35]. If a web application is built using a component-based architecture, most of the components can be matched to one or more of these design patterns.

To provide a concrete example, the *Menu Button* ARIA APG pattern can be used. This design pattern is applicable to buttons that open a menu when clicked. A menu is understandable to a user navigating using keyboard by menu items being focused. Focus can be determined either visually or by an assistive technology reading the focus programmatically. Success criterion 2.4.3 – *Focus Order* of WCAG, states that [6]:

> *"If a Web page can be navigated sequentially and the navigation sequences affect meaning or operation, focusable components receive focus in an order that preserves meaning and operability"*

This requirement is too ambiguous and vague for it to be usable to formulate a test case for an automated test. However, the ARIA APG Menu Button pattern supplies us with specific, detailed requirements for how focus state and change of focus should be handled in a menu and menu button. The ARIA APG pattern requirements *can* be used to formulate test cases. If the ARIA APG patterns are followed when developing the handling of focus for all the components of a web application, they will automatically conform to WCAG requirement 2.4.3.

Because the ARIA APG patterns provide such specific requirements for web content, it was chosen as one of the foundational related works for the AAT-WAC method. The method is only concerned with automated accessibility testing, and most web content accessibility standards provide requirements that are far too abstract or ambiguous to use for writing automated tests.

2.5.3 WCAG Evaluation Methodology (WCAG-EM)

When reading relevant works on structuring the process of web content accessibility evaluation, it became apparent that most, if not all, such works defined methodologies. This thesis presents the creation of a method, but no method for web content accessibility evaluation using component testing could be found. Therefore, parts of the methodology WCAG-EM were chosen as a major related work forming a basis for the created method.

When it comes to methodologies for web accessibility evaluations, there are plenty of options. These methodologies vary greatly in meticulousness, comprehensiveness, and purview. Some of the most important methodologies are the Website Accessibility Conformance Evaluation Methodology (WCAG-EM) by WAI [36], the BITV-test by DIAS GmbH [37], and the TECED Accessibility Evaluation Methodology by TecEd [38]. While—as mentioned—methodologies differ in comprehensiveness, almost all of the most important methodologies involve all of the methods listed in Section 2.1.4 – *Web Content Accessibility Evaluation Methods*, automated testing, manual testing, and user testing.

WCAG-EM was chosen as it is the most popular methodology and has been used as a basis for many of the methodologies created by other organizations worldwide. WCAG-EM cannot be presented in its entirety here, instead the five stages, called main steps, are listed here [39]:

1. **Define the scope of the evaluation:** decide what parts of the web application are included in the evaluation, the overall goals of the evaluation, as well as the WCAG (A, AA, or AA) conformance levels you aim to achieve.
2. **Explore the website:** Identify the types of web content that will be evaluated. This includes functionality, design, used technologies, and more.
3. **Select a sample:** Decide on a part of the web content that is to be evaluated.
4. **Evaluate the sample:** Determine the samples conformance to the current version of WCAG and document the evaluation steps.
5. **Report the evaluation findings:** Aggregate the results of evaluating the sample, combine with evaluation statements, and determine conformance level. When this stage is completed, return to stage 3.

3 Research Methodology

This chapter describes the research methodology that was used for this thesis. First, Section 3.1 – provides a general summary of the research strategy and its different parts. Section 3.2 provides an explanation of research methods used, alongside the motivation for choosing them. Section 3.3 describes all the phases that the research was divided into. Section 3.4 presents the research instruments that needed to be employed in order to gather data and perform the evaluations. Section 3.5 lists the threats to the validity of the research and explains their significance. Last, Section 3.6 describes the ethical considerations that were pertinent to the chosen research strategy.

3.1 Overview of Research Strategy

The purpose of the work described in this thesis is to create and evaluate a method for automated accessibility testing of web application components. Therefore, the chosen research strategy needs to be appropriate for structuring this kind of work. As a method is a typical example of what the Design Science Research (DSR) paradigm calls an *artifact*, in combination with DSR being well-suited for research in the field of information science, it was decided that the chosen research strategy be based on DSR [40].

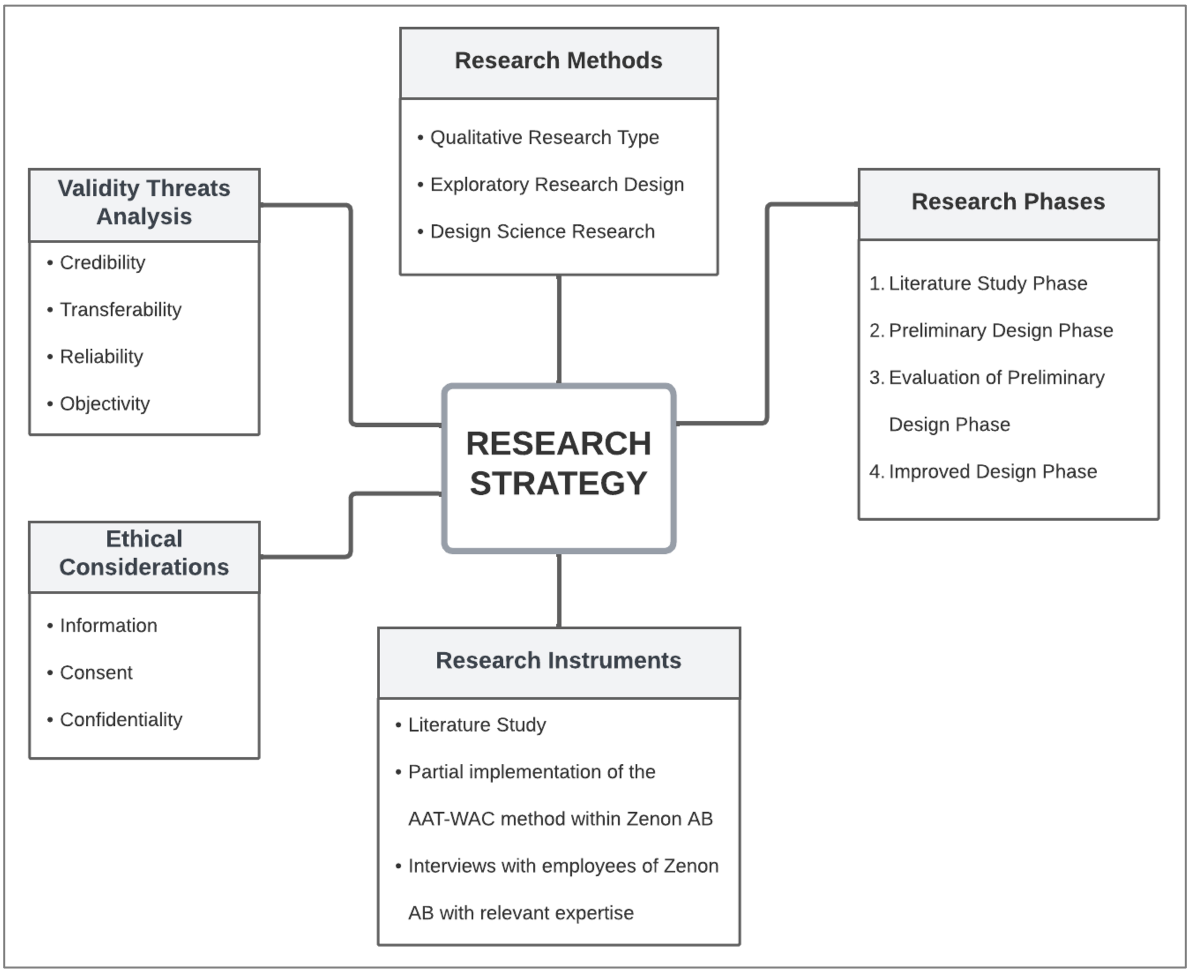

Figure 3-1: Overview of the research strategy

A diagram providing an overview of the research strategy is shown in Figure 3-1. As the figure illustrates, the research strategy consists of five named parts. These five parts of the research strategy are: (1) *Research Methods*, (2) *Research Phases*, (3) *Research Instruments*, (4) *Validity Threats Analysis*, and (5) *Ethical Considerations*. Part 2 of the research strategy, *Research Phases*, was defined using an existing DSR process model as its foundation. It is not an exact copy of a published DSR process model, however, it was heavily inspired by one. The published DSR process model that served as the basis for the *Research Phases* part is described in Section *3.3 – Research Phases* of this chapter.

3.2 Research Methods

This section presents the research methods that were used, as well providing the motivation for *why* they were used. Section 3.2.1 explains why qualitative research was chosen as the research type. Section 3.2.2 explains why the research undertaken for this thesis is exploratory in nature and motivates why this is suitable. Last, Section 3.2.3 describes DSR, why DSR is well-suited for solving the problem of the thesis and presents the DSR process model that served as a foundation for the research phases of the chosen research strategy.

3.2.1 Qualitative Research

Creating a method for accessibility testing web application components, as well as evaluating said method, is work that by its nature contains problem areas that are hard to define, and that focuses on gaining greater understanding of some problem. It is clear that the problem is interpretative in nature. Therefore, qualitative research is the research type that is most well-suited for structuring and guiding this work [41].

Qualitative research is an umbrella term including many different methods and methodologies, where data is collected through such techniques as interviews, surveys, case studies, and researcher observations. Qualitative research aims to build a comprehensive understanding of phenomena through a researcher-defined exploratory process, where understanding and analysis of the studies subject is continually expanded throughout the research process. Qualitative research also permits the researcher themself to be a source of data collection, for example through defining a set of evaluation criteria, or an evaluation model, and then performing a conformance evaluation. This is relevant to this thesis, as the created method was evaluated using an evaluation model based on a partial implementation of the method on a real-world web application [41].

Other qualitative research methods that were used in order to fulfill the purpose of this thesis were interviews and a literature study. The interviews were conducted as a part of the evaluation of the preliminary AAT-WAC method. The literature study provided the researcher with a theoretical and practical background for the thesis.

Qualitative research stands in contrast to quantitative research, which is focused on measurement and uses numerical data as the basis of its results and conclusions. Quantitative research must be controlled, replicable, and able to be used to predict events. It is also important that a quantitative research process is objective, formal, deductive, and systematic. Considering all the qualities mentioned, it is clear that the problem and purpose of this thesis is poorly suited to the quantitative research type [42].

3.2.2 Exploratory Research

The research strategy used in this thesis follows an exploratory research design. Exploratory research does not propose to supply a definite, conclusive answer to a set of research questions, but instead aims to explore a problem which cannot be clearly and precisely defined. The results of exploratory research are not conclusive evidence, but rather a greater understanding of the problem and its contexts. Conducting exploratory research allows for a range of causes and alternative options in the results. Among the advantages of exploratory research is greater flexibility and adaptability in laying the groundwork for future research. However, like qualitative research, exploratory research is heavily vulnerable to bias, and findings cannot often be generalized with a satisfactory degree of certainty [42].

Alternatively, a research strategy can follow a conclusive research design. With conclusive research design, objectives, problems, and data requirements need to be precisely defined. The goal is to produce a conclusive answer to a specific question. Keeping the purpose of this thesis in mind and considering the qualities of exploratory research versus those of conclusive research, it is clear that an exploratory approach is the better choice for this thesis [42].

3.2.3 DSR Paradigm

DSR is a problem-solving paradigm that aims to increase the knowledge of some area through the creation of innovative artifacts. DSR can be thought of as a template meant to define research methods and strategies. DSR focuses on the design and evaluation of the artifact, with the explicit goal of being able to improve the functional performance of the artifact. The artifacts that are created as a result of DSR—the AAT-WAC method in this case–are meant to improve the environment in which they are instantiated or used. The result of DSR is not only the created artifact, but also the design knowledge acquired during the research [11].

The most integral part of the research strategy of this thesis is the research phases, and DSR served as the foundation for these phases. In order to define the research phases, several different DSR process models were consulted during the thesis literature study. After the literature study had been conducted, the choice was made to base the research phases on the DSR process model created by Vom Brocke, Hevner, and Maedche [11, Ch. 1.3]. Their DSR process model was in turn inspired by perhaps the most referenced DSR process model available, the one proposed by Peffers et al. in 2007 [43].

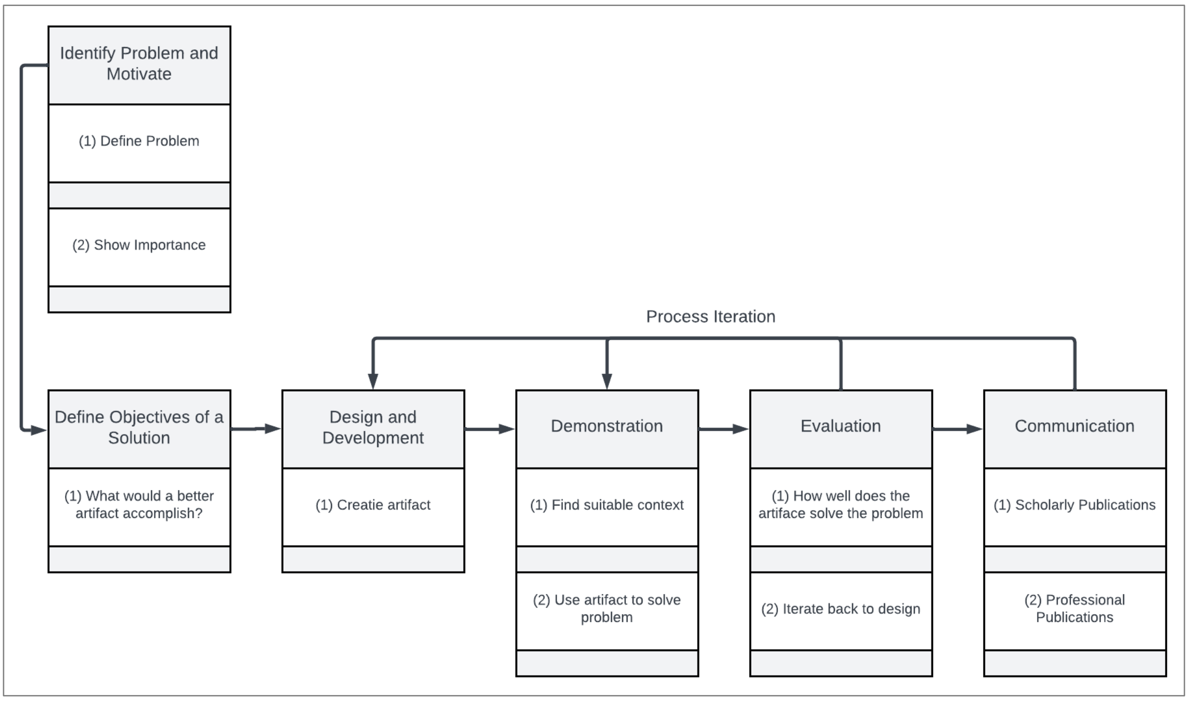

Figure 3-2: DSR process model created by Vom Brocke, Hevner, and Maedche (inspired by Figure 2 on page 4 of [40])

Figure 3-2 shows an overview of the process model created by Vom Brocke, Hevner, and Maedche. The DSR process model chosen as a foundation for the research phases consists of six steps, called *activities* in this specific process model. These six activities served as the foundation for the four research phases used in the chosen research strategy. These six activities, as well as the research phase they correspond to, are presented here:

- **Activity #1 - Problem identification and motivation:** This activity defines the research problem and motivates the benefits of a solution. This activity corresponds to the research phase named *Literature Study*. The problem identified through the literature study research phase was that there is no method for automated accessibility testing of web application components.
- **Activity #2 - Define the objectives for a solution:** During this activity the problem definition and knowledge of the problem area is used to infer possible objectives of the proposed solution. These objectives can be quantitative or qualitative. This activity corresponds to the *Preliminary Design* research phase.
- **Activity #3 - Design and development:** An artifact—in this case a method—is designed and developed during this activity. This activity includes deciding on the desired functionality of the artifact, defining its architecture, and creating the actual artifact. This activity was translated into two different research phases. First, it formed the foundation of the *Preliminary Design* phase together with activity 2, which is described above. Second, due to resource constraints, the activity also corresponds to the *Improved Design* research phase, where an improved method was created after evaluating the preliminary method.
- **Activity #4 - Demonstration:** During this activity the activity is used to solve one or more instances of the research problem, and in doing so demonstrate its use. This activity corresponds to the *Evaluation of the Preliminary Method* research stage, specifically using the preliminary AAT-WAC method to implement basic accessibility testing of the components of a real-world web application. Due to resource constraints this activity and its corresponding research phase is not conducted again after the final research stage, *Improved Design*.
- **Activity #5 - Evaluation:** Here the artifact is evaluated by how it supports a solution to the research problem. This evaluation can take many forms, depending on the context of the problem and research. This activity corresponds to the *Evaluation of Preliminary Method* research phase.
- **Activity #6 - Communication:** This activity consists of communicating pertinent aspects of the designed artifact to the relevant stakeholders. This activity corresponds to two different research phases. Firstly, the activity is carried out at the end of the *Preliminary Design* research phase, and then again at the end of the *Improved Design* research phase.

As can also be seen in Figure 3-2, activities 3-6 of the chosen DSR process model are iterative. This means that they are continually executed in the pre-defined order until the researcher deems the results satisfactory. The number of iterations can also be limited by a lack of resources, such as time or funding.

3.3 Research Phases

This section describes the four research phases that were the result of adapting an existing DSR process model for this research strategy. The four phases and their corresponding sub-phases are illustrated in Figure 3-3. As can be seen in the figure, not all sub-phases were carried out sequentially, some had to be conducted in parallel due to the nature of the commissioned work.

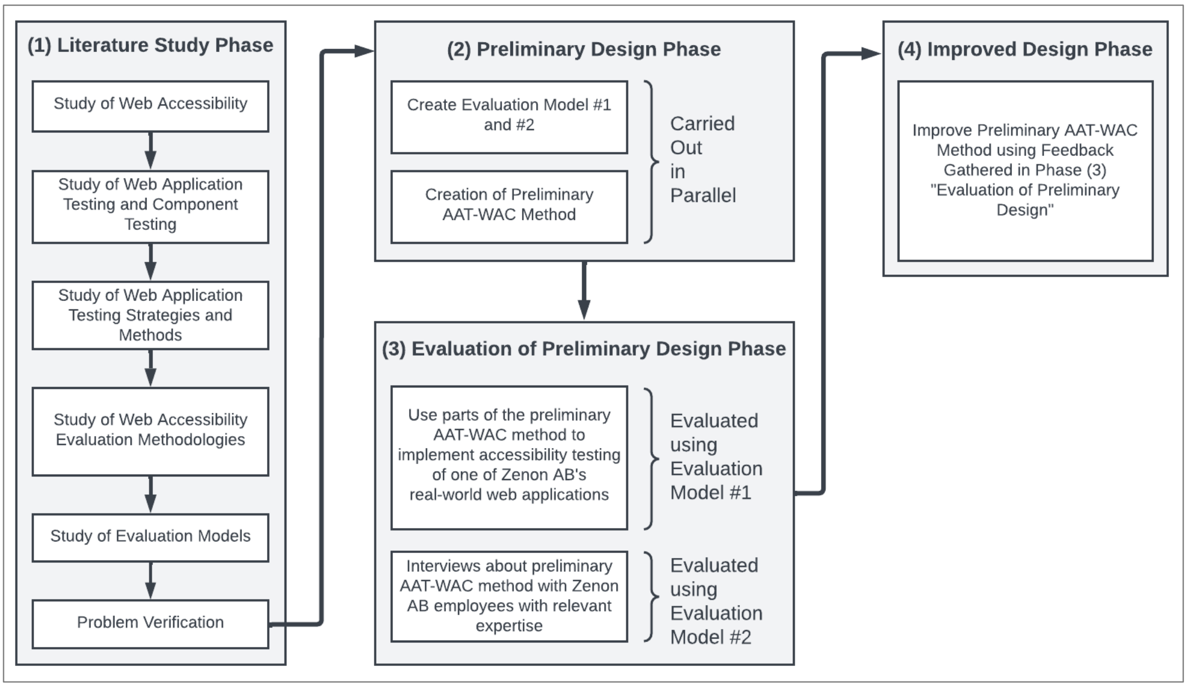

Figure 3-3: **The four research phases of the research strategy**

Section 3.3.1 presents the literature study research phase and its six sub-phases. Section 3.3.2 describes the *Preliminary Design* research phase where the preliminary AAT-WAC method and the evaluation models were created. Section 3.3.3 presents the *Evaluation of Preliminary Design* research phase where parts of the preliminary method was used to implement basic accessibility testing of the components of a real-world web application, as well as describing the interviews that were carried out with three employees of Zenon AB to evaluate the preliminary method. Last, Section 3.3.4 details the *Improved Design* research phase where the data gathered from the evaluations in the previous research phase was used to make improvements to the AAT-WAC method.

3.3.1 Literature Study Phase

The literature study research phase was divided into six sub-phases, of which the first five are: (1) *Study of Web Accessibility*, (2) *Study of Web Application Testing and Component Testing*, (3) *Study of Web Application Testing Strategies and Methods,* (4) *Study of Web Accessibility Evaluation Methodologies*, and (6) *Study of Evaluation Models*. Last, the final sub-phase of the literature study consisted of verifying that the thesis problem identified the first six sub-phases, namely “There is no method for automated accessibility testing of web application components”, actually is true. This sub-phase is called the *Problem Verification Sub-phase*.

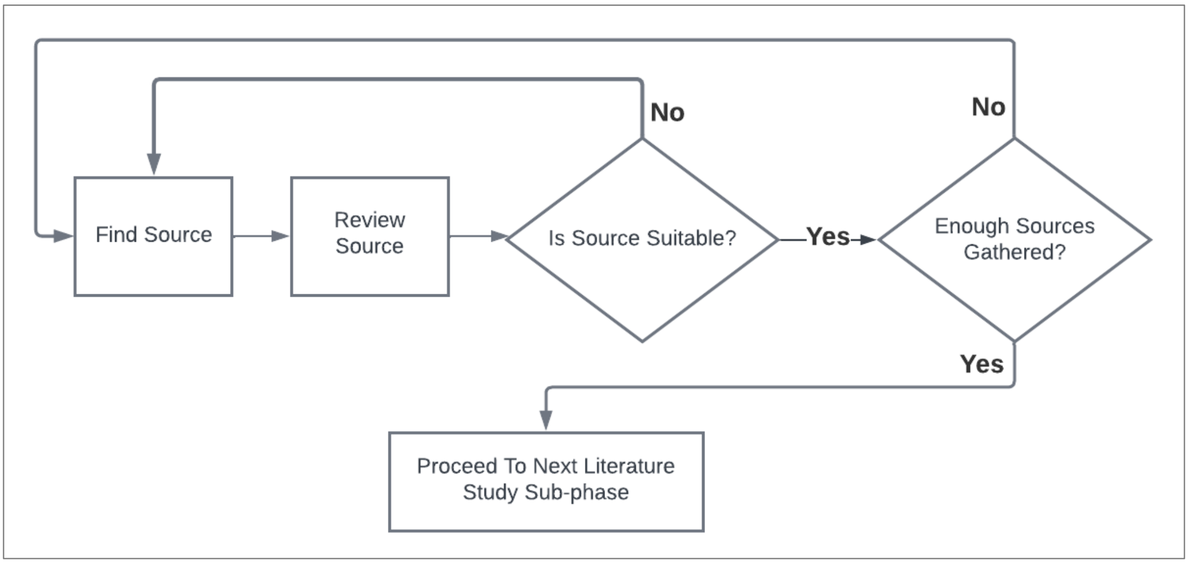

Figure 3-4: Workflow for each sub-phase of the literature study research phase

The workflow that was followed for each of the five sub-phases is illustrated in the diagram of Figure 3-4. As the diagram shows, for each sub-phase the phase was initiated by identifying a collection of possible sources from research databases, followed by reviewing the sources to determine their suitability, and then lastly deciding if enough sources had been gathered to proceed to the next sub-phase. For all the listed sub-phases, the following research databases were used: IEEE Explore [44], Web of Science [45], Scopus [46], and Google Scholar [47]. These databases were chosen as they are respected and well-known, as well as being freely accessible to the researcher thanks to their university, the KTH Royal Institute of Technology.

Listed below is a brief presentation of each of the five sub-phases of the literature study phase, together with the challenges of each phase and what was learned:

- **Study of Web Accessibility:** During this phase a holistic understanding of modern web accessibility, web accessibility evaluation, and web accessibility standards needed to be gained. This led to a sub-phase with an extensive range of sources. What differentiated this sub-phase from the others was that many of the most important sources were readily available as web content, as the most important web accessibility organizations have published their material to the public. The greatest challenge of the sub-phase was forming a general understanding of such a sprawling subject in a fairly short amount of time. The most important, major works that formed the basis of this sub-phase are [1], [2], [5], [6], [8], [18], [48].
- **Study of Web Application Testing and Component Testing:** This sub-phase included consulting both published works found through research databases, but also technical documentation of many of today's most popular web application testing tools and resources. The scope of this sub-phase was defined by the work that the company, Zenon AB, had commissioned. All the web application testing technologies that were investigated needed to be usable when testing Zenon AB's real-world web application that formed the foundation of the *Evaluation of Preliminary Method* research phase. This sub-phase resulted in the following sources being identified and used to gain an understanding of the relevant subjects [28], [34], [49]–[51].
- **Study of Web Application Testing Strategies and Methods:** This sub-phase proved challenging because of the jumbled, incoherent, and sometimes straight up faulty

usage of the term *method* in software engineering and its related fields. When determining the suitability of the sources found, it needed to be determined that they were actually describing methods or strategies, and not methodologies or process models. This sub-phase resulted in one major work being chosen as the primary source for web application testing methods: [34].

- **Study of Web Accessibility Evaluation Methodologies:** This sub-phase consisted of identifying sources on structuring and guiding the work of performing accessibility evaluations of web content. The sub-phase revealed that almost all major works in this area were methodologies, and not methods or strategies. As no relevant method could be found, the most well-known methodology was chosen as the major source instead [39].
- **Study of Evaluation Models:** During this sub-phase sources pertaining to evaluation models were identified and reviewed for suitability. This sub-phase proved challenging as even though several published evaluation models were found, they related to contexts that were not fully applicable to the context of the thesis. Specifically, evaluation models related to evaluating web content accessibility testing methods, or web content accessibility testing method development, could not be found. Therefore, the sources chosen were major works focused on evaluation models from an over-arching perspective [52], [53].

Finally, the literature study phase was concluded by working through the **Problem Verification Sub-phase**. This sub-phase was needed in order to verify that the researcher's premise—that there were no methods for automated accessibility testing of web application components—was actually true. This premise was arrived at while working through the first five sub-phases of the literature study. The verification consisted of searching several major research databases for a combination of keywords and determining that no such method could be found. Fully presenting the database query strings, which research databases that were used, and the search results cannot be done in this section. Instead, a presentation of this work can be found in full in Appendix A.

3.3.2 Preliminary Design Phase

The preliminary design research phase consisted of two sub-phases that were carried out roughly in parallel, (1) *Creation of Evaluation Models Sub-phase*, and (2) *Creation of Preliminary AAT-WAC Method Sub-phase*. The evaluation models created during the first sub-phase were to be used to evaluate the AAT-WAC method in the research phase *Evaluation of Preliminary Method*.

The evaluation models are presented in full in Chapter 4 – *Evaluation Models*, but a brief summary of their general structure is included here. The evaluation models are based on evaluation criteria suitable for evaluating artifacts such as methods, frameworks, and process models. The evaluation criteria used are the same for both evaluation models, with some small differences. For example, the evaluation model for the interviews includes the *Interviewee Credibility* evaluation criterion, which is not included in the evaluation model for the partial implementation of the preliminary method.

The creation of the preliminary AAT-WAC method was mainly based on the knowledge gathered during the literature study research phase, but knowledge gained from creating the evaluation models also contributed. The preliminary method was also created using information gathered from analyzing the real-world web application that Zenon AB—the company that had commissioned the research—had supplied the researcher with. While a satisfactory knowledgebase had been gathered from the literature study, access to a real-world web application on which the method was to be applied in the future was an important resource.

3.3.3 Evaluation of Preliminary Method

This research phase consisted of evaluating the preliminary AAT-WAC method by the evaluation models created in the previous research phase, *Preliminary Design*. First, the preliminary method is used to implement automated accessibility testing of the components making up Zenon AB's real-world web application *Blixtvakt*. This implementation is then evaluated by the researcher themself, as well as serving as the basis for conducting interviews with three of Zenon AB's employees with knowledge and experience relevant to the research area. The structure of these evaluation interviews is more thoroughly explained in Chapter 4 – *Evaluation Models*.

The interview questionnaire that formed the foundation of this evaluation model was answered by three of Zenon AB's employees. The sampling method described in Section 3.5 resulted in these three employees being chosen as they possessed the required experience in web development. All three interviewees worked at Zenon AB's Stockholm office, and therefore the presentation stage of the evaluation model could be carried out in person.

3.3.4 Improved Design Phase

In the *Improved Design* research phase, the evaluations conducted in the previous research phase were used as a foundation for improving the preliminary AAT-WAC method. The feedback collected from the evaluation interviews, together with knowledge the researcher themselves acquired while using the method to implement automated accessibility testing of the components of the real-world web application, were both used to make decisions about the improvements that needed to be made.

It is desirable to perform research phases of this nature in several iterations, but due to resource constraints only one iteration could be completed during the span of the research. However, the research phase was designed so that it would allow for multiple iterations to be performed. Also resulting from the resource constraint mentioned above, the improved method is only evaluated from the researcher's perspective once, and no evaluations using interview questionnaires or implementing the method in an industrial setting were conducted.

3.4 Research Instruments

When following the chosen research strategy, three qualitative research instruments needed to be used, (1) *Literature Study*, (2) *Partial Implementation of Preliminary Method (commissioned work),* and (3) *Interview Questionnaire*. These three instruments are presented here in brief:

- **Literature Study:** The work carried out by choosing to use this research instrument is more thoroughly presented in Section *3.3.1 – Literature Study Phase*. The literature study not only supplied the researcher with the knowledge to conduct the research presented in the thesis, but also revealed the problem that is the foundation of the whole thesis.
- **Partial Implementation of Preliminary Method (commissioned work):** Using the preliminary AAT-WAC method to implement accessibility testing of the components of Zenon AB's Blixtvakt web application not only served as the basis for the interviews, but were also part of the work commissioned by Zenon AB. The partial implementation is also evaluated from the researcher's perspective. The evaluation model defining this evaluation is further explained in Section 4.1.

- **Interview Questionnaire:** Three of Zenon AB's employees with experience in web development were interviewed. Because of the research being qualitative and exploratory, the interviews were designed to be focused on gaining a greater understanding of the problem areas, and not force interviewees to supply definitive answers on interpretative questions. These interviews were structured according to the evaluation model presented in Section 4.2.

3.5 Sampling Method

Two sampling methods needed to be defined for the chosen research strategy. First, a sampling method for partial implementation of the preliminary AAT-WAC method needed to be defined, and secondly, a sampling method for choosing suitable interviewees was also needed. The considerations when defining these two sampling methods are presented in this section.

The sampling method for the partial implementation of the preliminary AAT-WAC method was considered first. The basis for this sampling method had already been decided by the commissioned work of Zenon AB. No matter what kind of accessibility testing solution the researcher wanted to investigate, it had already been decided that it should be tested on Zenon AB's real-world web application Blixtvakt. The decision for this sampling method was therefore if more real-world applications of Zenon's should be included, or if the researcher should develop a prototype web application purely for evaluating the method. This solution would have led to the preliminary AAT-WAC method being partially implemented on both the prototype application and Blixtvakt. In the end the choice was made to have Blixtvakt as the only web application in the sample due to a lack of resources. The sampling technique used to select web applications to include in the evaluation model therefore can be described as convenience sampling [54].

When it came to sampling methods for choosing suitable interviewees, the commissioning company was of great help. The sampling technique used to identify possible interviewees combined purposeful sampling with convenience sampling [54]. For the purposeful sampling, Zenon AB aided the researcher in identifying employees with relevant knowledge in web application development. Previous experience in web application testing is not needed, as the method should be usable by web developers with little to no experience in the subject. The search for interviewees could have broadened to include web developers working for other companies, but due to a lack of resources—most critically time—the decision was made to only include Zenon AB employees at their Stockholm office, which means that convenience sampling was also used.

3.6 Validity

Validity is used to evaluate the soundness and appropriateness of the chosen research method, or methods. The criteria used to evaluate the validity are (1) *internal validity,* (2) *external validity,* (3) *dependability*, and (4) *conformability*. However, these criteria are used to evaluate quantitative research, which differs from qualitative research. They therefore cannot be applied to the qualitative research in this thesis [55]. Instead, validity criteria adapted to qualitative research needed to be employed. These qualitative validity criteria and the quantitative criteria they correspond to are listed here [56]:

- ***Credibility*** (corresponding to ***Internal Validity***): Evaluates whether the findings resulting from the research method are credible, meaning that it attempts to determine the degree of confidence in the truthfulness of the findings. In qualitative research where the findings are based on interviews, establishing the credibility of the interviewees increases the credibility of the research results. Therefore, *Interviewee Credibility* was added as a criterion to the evaluation model.

- **Transferability** (corresponding to **External Validity**): Evaluates the degree to which the research findings are generalizable to other contexts. Qualitative research findings are notoriously difficult to generalize. In order to improve transferability interviewees were asked to consider how well they thought the AAT-WAC method would work in other contexts than the one it had been partially implemented in.
- **Reliability** (corresponding to **Dependability**): Evaluates the repeatability of the research process. If research is reliable, its findings should be consistently repeatable if the research is repeated in another context. A lack of reliability is a well-known weakness of qualitative research, which can be mitigated by repeating the research process in different contexts. A lack of resources made this unfeasible, and thus the reliability of the research documented in this thesis remained low throughout the process.
- **Objectivity** (corresponding to **Conformability**): Evaluates the neutrality of the researcher, meaning that their biases did not affect the results of the research process. While several objectivity threats could not be mitigated in this study, a conscious effort was made to make the interview design process more objective. This was done by designing the interviews based on established research, instead of a common qualitative objectivity threat where interviews designed based only on the researcher’s own knowledge, which render them vulnerable to bias.

3.7 Ethical Considerations

Many ethical consideration, requirements, and guidelines for qualitative research have been published. The literature study resulted in three such ethical requirements being deemed relevant to the chosen research strategy [57]. These three requirements are: (1) *Information*, (2) *Consent*, and (3) *Confidentiality*. These requirements are briefly presented here:

- **Information:** The interviewee must be informed of their rights during the interview, how their input will be handled, and what the purpose of the research is. In order to meet this requirement interviewees were informed that they had voluntarily agreed to the interview and could cancel it at any time. They were also presented with the purpose of the research, and how their input would be used in the research.
- **Consent:** Interviewee participation must be based on continual consent. This is ensured by making it clear that participation is voluntary when contacting potential interviewees and asking them if they wish to participate, together with communicating that participation can be terminated at any point.
- **Confidentiality:** Ensures that the interviewee knows beforehand what information from the interviewing process will be published, and that the researcher respects the information sharing decisions agreed on before the interview. If the interviewee does not agree with what will be shared, the interview either needs to be redesigned or not carried out at all.

4 Evaluation Models

This chapter presents the evaluation models that were the results from the *Creation of Evaluation Models* sub-phase of the *Preliminary Design* research phase. This research phase is described in Chapter 3, Section 3.3.2. First, Section 4.1, describes the evaluation model that was used to evaluate the partial implementation of the preliminary AAT-WAC method within Zenon AB. Section 4.2 describes the evaluation model used when evaluating the preliminary AAT-WAC method by interviewing employees of Zenon AB.

4.1 Evaluation Model for Partial Implementation of the preliminary AAT-WAC Method within Zenon AB

The two main approaches for evaluating a newly created method are to either implement it in an industrial setting, and then compare the results to that of published methods, or to interview respondents with relevant expertise [58]. However, as the *Problem Verification* sub-phase presented in section 3.3.1 revealed, there are no published methods similar to the preliminary AAT-WAC method. Therefore, the preliminary method cannot be evaluated by such a comparison.

In their article on evaluating process models from 2002, Sedera, Rosemann, and Gable propose that process models should be evaluated both from the user and modeler perspectives [58]. In this case the modeler is the researcher, and henceforth *researcher* is used instead of *modeler* when referring to the creator of the method. While AAT-WAC is a method, the nature of a method for web application testing was deemed similar enough to that of process models that their proposal was considered relevant. Following this decision, an evaluation model for the partial implementation of the preliminary method within Zenon AB was created.

The reason for conducting only a partial implementation of the preliminary method was a lack of resources, most pressingly time, but also prerequisite knowledge. The partial implementation of the preliminary method was evaluated using the following evaluation criteria:

1. Semantic Correctness
2. Syntactic Correctness
3. Usefulness
4. Method Adaptability and Flexibility
5. Appropriateness of the ARIA APG Design Patterns as Requirement Source for Tests

By using the evaluation criteria listed above, an evaluation model for the partial implementation of the preliminary AAT-WAC method within Zenon AB was created. This evaluation model can be viewed in Table 4-1, which is included below. As the table shows, the questions are more open-ended than yes/no questions. The questions were answered by the interviewees using a survey tool, but before the survey was sent to the interviewees each question was presented and explained during the Presentation Sub-phase. During this presentation the interviewees could ask for more information on questions they did not understand. The interviewees were also informed that they could contact the researcher at any time during their answering process of anything about the questionnaire that needed further explaining.

Table 4-1: Evaluation model for the partial implementation of the preliminary method within Zenon AB

Evaluation Criteria	Question Number	Question
Semantic Correctness	1.1	Is the meaning of the phases and sub-phases of the method clear?
	1.2	Can the phases and sub-phases of the method be followed?
	1.3	Are there parts of the method that are not easily understandable? If so, which parts?
Syntactic Correctness	2.1	Are there method phases or sub-phases that you feel are redundant?
	2.2	Are there method phases or sub-phases that you feel are partially redundant, or should be modified?
	2.3	Are there any aspects of implementing automated accessibility testing of web application components that you feel the method does not address? If yes, do you have ideas for stages that could be added?
	2.4	Is the sequence of method sub-phases appropriate?
Usefulness	3.1	Do you think this method could be useful for implementing automated accessibility testing of web application components?
	3.2	If you do not think the method as a whole is useful, are there parts of it that could be?
	3.3	What weaknesses do you identify when it comes to the method's usability?
Method Adaptability and Flexibility	4.1	Is the method flexible enough to be usable in other contexts? (such as a React app instead of an Angular app, and using a different testing framework)
	4.2	Do you think the method is documented in such a way that it allows for making adaptations?
Appropriateness of the ARIA APG Design Patterns as a Requirements Source for Tests	5.1	Can each component of the web application be matched to one or more ARIA APG patterns?
	5.2	Can the chosen ARIA APG patterns be used to find accessibility requirements?
	5.3	If accessibility requirements can be found, are they concrete enough to turn into test cases for automated tests?

Evaluation Criteria 1 to 5 are explained in greater detail in Section 4.4-4.8, respectively. As for evaluation criterion (5) *Appropriateness of the ARIA APG Design Patterns as Requirement Source for Tests*, this criterion was added as the ARIA APG design patterns collection is such an integral part of the preliminary AAT-WAC method. This criterion was not included in the interview questionnaire as the interviewees had no opportunity to implement the method themselves, and therefore lacked the required knowledge in the area.

4.2 Evaluation Model for Interview Questionnaire

This chapter presents the second evaluation model that was the result of the *Creation of Evaluation Model* sub-phase of the *Preliminary Design* research phase. This research phase is described in Chapter 3, sub-section 3.3.2 – *Preliminary Design Phase*. The evaluation model defines the evaluation criteria that were used to create the interview questionnaire. The evaluation model also defines how the preliminary AAT-WAC method should be demonstrated to the chosen interviewees in order for them to gain the required understanding of the method.

First, Section 4.1 provides a brief presentation of the interviewees. Then, Section 4.2, describes how the preliminary method was presented to the chosen interviewees, and motivates why it was demonstrated in this way. Thereafter, Section 4.2-4.6 describes the chosen evaluation criteria: (1) *Interviewee Appropriateness*, (2) *Semantic Correctness*, (3) *Syntactic Correctness*, (4), *Usefulness*, and (5) *Method Adaptability and Flexibility*, respectively.

Using the above evaluation criteria resulted in an interview questionnaire that constituted the second part of the evaluation model, with the preliminary method demonstration being the first part. This interview questionnaire is presented in Table 4-1, included below.

Table 4-2: Evaluation model interview questionnaire

Evaluation Criteria	Question Number	Question
Interviewee Credibility	**1.1**	What is your profession?
	1.2	What is your role?
	1.3	How long has this been your role?
	1.4	Have you had previous roles relevant to this area?
	1.5	What is your experience in web development?
	1.6	What is your experience in web accessibility?
	1.7	What is your experience in accessibility testing web applications?
Semantic Correctness	**2.1**	Is the meaning of the stages and guidelines of the method clear?
	2.2	Can the stages and guidelines of the method be followed?
	2.3	Are there parts of the method that are not easily understandable? If so, which parts?
Syntactic Correctness	**3.1**	Are there method stages or guidelines that you feel are redundant?
	3.2	Are there method stages or guidelines that you feel are partially redundant, or should be modified?
	3.3	Are there any aspects of implementing automated accessibility testing of web application components that you feel the method does not address? If yes, do you have ideas for phases or sub-phases that could be added?
	3.4	Is the sequence of method phases and sub-phases appropriate?

Table 4-3: Evaluation model interview questionnaire

Usefulness	**4.1**	Do you think this method could be useful for implementing automated accessibility testing of web application components?
	4.2	If you do not think the method as a whole is useful, are there parts of it that could be?
Method Adaptability & Flexibility	**5.1**	Is the method flexible enough to be usable in other contexts? (such as a React app instead of an Angular app, and using a different testing software framework)
	5.2	Do you think the method is documented in such a way that it allows for making adaptations?

As the questionnaire in Table 4-2 shows, the evaluation criteria are checked using the same questions as those in the evaluation model presented in Section 4.1. It is important to note that the interviewees' answers to many of these questions should be considered as more reliable than the researcher's answers. The interviewees have experience from real-world web development projects in an industrial setting, something which the researcher lacks.

4.2.1 Demonstration of the Preliminary AAT-WAC Method

In order to evaluate the preliminary AAT-WAC method using interviews, the method needed to be presented to the interviewees in such a way that they gain the required knowledge of the method. This presentation needs to account for several factors, such as time and the interviewees' potential involvement in the creation process of the preliminary method. Considering these aspects, the decision was made to not only present documentation defining the preliminary method, but to also use the preliminary method to implement automated accessibility testing of the components of a real-world web application, and to then include a presentation of that implementation.

The real-world web application *Blixtvakt* was chosen to illustrate the usage of the preliminary AAT-WAC method. This application was developed by Zenon AB, and the chosen interviewees all had previous knowledge of it, though that knowledge varied in its comprehensiveness. Using the preliminary method, accessibility testing of Blixtvakt's components was implemented. The new prototype web application—with accessibility tests included—was shared with the interviewees together with documentation of what work had been done to it by following the method. The interviewees could then view the results of implementing parts of the method themselves, and in doing so gain a better understanding of the preliminary AAT-WAC method.

4.2.2 Interviewee Appropriateness

As the first part of these interviews, the appropriateness of the chosen interviewee needed to be determined, as this appropriateness affects the rest of the interview. If the interviewee is appropriate, it means they have the required knowledge of the subject in question so that the feedback they provide can be deemed believable and trustworthy. If this appropriateness is properly evaluated, the data gathered from exploratory interviews is not credible. Therefore, this evaluation criteria needed to be included.

In this case, a method for implementing automated testing of web application components was being evaluated. However, this method was meant to be usable by web developers inexperienced in accessibility testing, or inexperienced in automated testing of web applications, or both. The method was also meant to be usable by web developers working alone. This meant that the most important factor for appropriateness was the interviewees experience in web development. As long as the interviewee had worked in web development, differing experiences in other aspects related to the method could actually provide better interview results. Considering this, the evaluation criterion was evaluated using the following questions:

- **1.1:** What is your profession?
- **1.2:** What is your role?
- **1.3:** How long has this been your role?
- **1.4:** Have you had previous roles relevant to this area?
- **1.5:** What is your experience in web development?
- **1.6:** What is your experience in web accessibility?
- **1.7:** What is your experience in accessibility testing web applications?

4.2.3 Semantic Correctness

For the created method to be understandable to its users, it needs to be semantically correct. Semantic correctness refers to the truthfulness of what is being conveyed, meaning that things are called what they actually are. A method with a high degree of semantic correctness is easier for its users to follow and understand compared to a method with lower semantic correctness. The semantic correctness evaluation criterion of the preliminary AAT-WAC method was evaluated by the following questions:

- **2.1:** Is the meaning of the stages and guidelines of the method clear?
- **2.2:** Can the stages and guidelines of the method be followed?
- **2.3:** Are there parts of the method that are not easily understandable? If so, which parts?

4.2.4 Syntactic Correctness

Syntactic correctness—in the context of a method—refers to the correctness of the order and inclusion of the different method phases and their instructions. This means that if the method is syntactically correct, the different phases of the method are rightfully included, but also that the order they are placed in is appropriate. The inclusion of this evaluation criterion was one of the major factors of deciding to also include an implementation of the preliminary method on a real-world web application in the demonstration, as the questions would be hard to answer without seeing the method in action. This evaluation criterion was evaluated using the following questions:

- **3.1:** Are there method phases or sub-phases that you feel are redundant?
- **3.2:** Are there method phases or sub-phases that you feel are partially redundant, or should be modified?
- **3.3:** Are there any aspects of implementing automated accessibility testing of web application components that you feel the method does not address? If yes, do you have ideas for phases or sub-phases that could be added?
- **3.4:** Is the sequence of method phases and sub-phases appropriate?

4.2.5 Usefulness

The usefulness evaluation criterion is more self-explanatory than the previous criterion. Usefulness—in this context—refers to how usable the interviewee finds the method that is being presented to them.

As with all evaluation criteria evaluated using interviews in qualitative research, the interviewees personal understanding of the question and its contents will affect the answer. To combat this, the questions evaluating this evaluation criteria attempted to provide more context for the interviewee. The questions are as follows:

- **4.1:** Do you think this method could be useful for implementing automated accessibility testing of web application components?
- **4.2:** If you do not think the method as a whole is useful, are there parts of it that could be?

4.2.6 Method Adaptability and Flexibility

The preliminary AAT-WAC method is demonstrated to the interviewee in a very specific context. In order for the method to be usable in other contexts it needs to be adaptable and flexible. This evaluation criterion is answered by gathering feedback from the interviewee about their opinions on using the preliminary method in another context. Methods are, by their nature, more flexible than methodologies or process models, so the questions need to account for this fact. The evaluation criterion was evaluated using the following questions:

- **5.1:** Is the method flexible enough to be usable in other contexts? (such as a React app instead of an Angular app, and using a different testing software framework)
- **5.2:** Do you think the method allows for adaptations?

5 Overview of the Preliminary AAT-WAC Method

This chapter presents an overview of the preliminary AAT-WAC method, which is the result of working through the *Preliminary Design* research phase described in Section 3.3.2. The reason for solely presenting an overview instead of the method documentation in full is that this would take up too much space and render the thesis unreadable. Instead, the preliminary method and its phases and sub-phases are described in full in Appendix B. The preliminary design presented in this chapter provides a method for implementing automated accessibility testing of web application components.

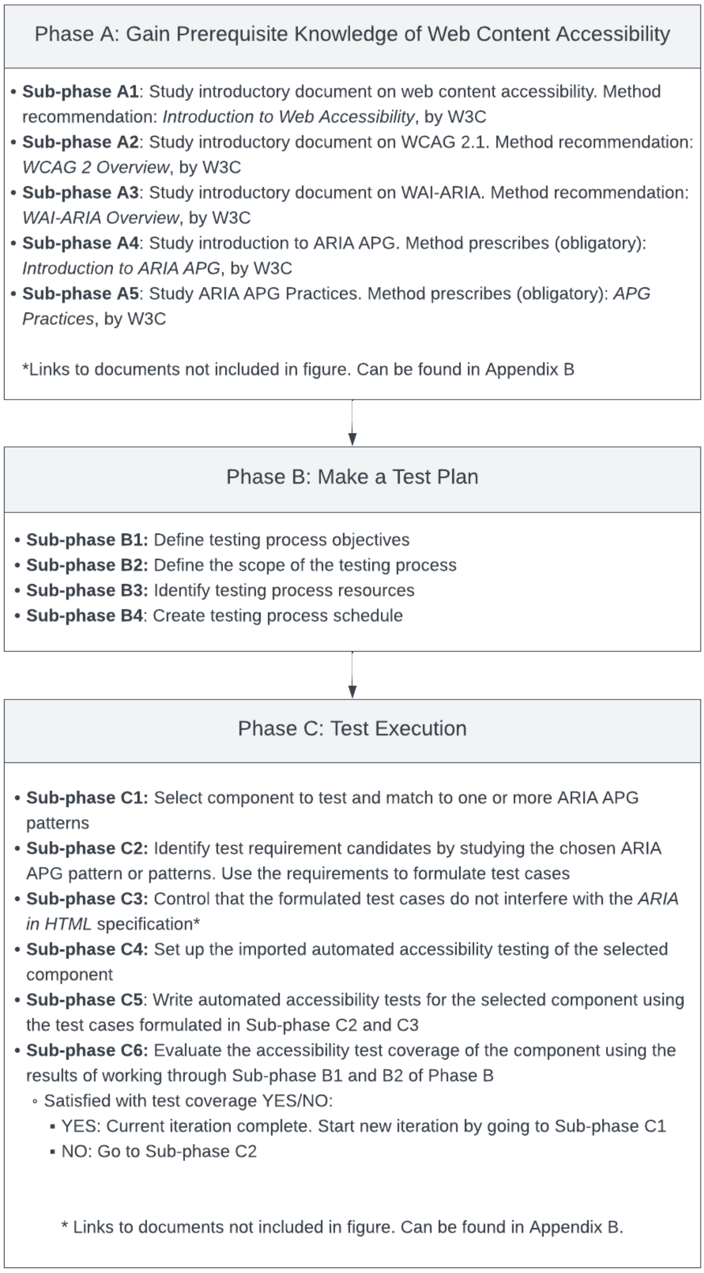

Figure 5-1: Diagram illustrating the phases and sub-phases of the preliminary AAT-WAC method

Figure 5-1, which is included above, shows an overview of the phases and sub-phases of the method. As the figure shows, the *Preliminary Design* research phase resulted in a method that was divided into three main phases: (1) *Phase A – Gain Prerequisite Knowledge of Web Content Accessibility*, (2) *Phase B – Make a Test Plan*, and (3) *Phase C – Test Execution*. These three main phases are made up of several sub-phases.

Phase A is aimed at providing guidelines for the method user in order to acquire the prerequisite knowledge of web content accessibility that is needed in order to follow the next two phases of the method. This is achieved by including recommendations for web content accessibility documents that the method user should study. As few documents as possible are included, and every document is introductory and holistic in nature, meaning that method users without previous knowledge should be able to both understand the document and gain an understanding that is satisfactorily comprehensive.

Phase B provides guidelines for formulating a test plan for the component tests that are to be written. It is heavily based on Kinsbruner's testing strategy described in Section 2.5.1. The sub-phases should be done in order, but each sub-phase allows for adaptation. First, in sub-phase B1, the objectives of the testing process that is to be undertaken are defined, followed by sub-phase B2 where the scope is determined, which consists of—but is not limited to—testing areas, features, and boundaries. The second half of the phase starts with sub-phase B3, where the work of identifying the testing resources needed to carry out the testing process—such as personnel, skillsets, tools, and infrastructure—is performed. As an illustrative example, the testing software framework that is to be used should be chosen in this sub-phase. Last, in sub-phase B4, a testing schedule outlining aspects such as activities, timelines, and more, is defined.

Phase C provides guidelines for executing the automated testing. This includes writing the tests, running them, as well as documenting and evaluating the results and test coverage. Phase C, unlike the two previous phases, is an iterative phase. Each iteration implements automated accessibility testing for one component, and the phase is repeated until all components that should be tested have been so. Phase C is heavily dependent on the test plan created in Phase B, and if no test plan has been created, Phase C cannot be properly followed. The foundation for Phase C is the ARIA APG design patterns, which are presented in Section 2.5.2. At the start of the iteration, the selected component is matched to one or more of the ARIA APG patterns. If no such match can be made, the selected component cannot be tested using the guidelines in Phase C.

6 Partial Implementation and Evaluation of the Preliminary AAT-WAC Method within Zenon AB

This chapter describes the partial implementation and subsequent evaluation of the preliminary AAT-WAC method within an industrial setting. Partial implementation refers to the fact that only parts of the method were implemented. First, Section 6.1 presents the context in which the method was partially implemented, as well as detailing exactly what parts of the method were included. Second, Section 6.2 details the results of the researcher evaluating the partial implementation using the evaluation model presented in Section 4.1.

6.1 Context

The basis for the partial implementation of the method was the real-world web application *Blixtvakt* which was developed by Zenon AB. The researcher and Zenon AB decided that Blixtvakt was an appropriate choice of web application through discussing different alternatives. While some consideration to web content accessibility had been made during development, the web application was still lacking when it came to accessibility. While the AAT-WAC method is meant to be used during development, the nature of the commissioned work was such that the method could only be implemented on web applications that had already been developed.

Table 6-1: Altered version of Phase C of the preliminary AAT-WAC method used for the partial implementation

Phase C: Test Execution (Altered Version for Partial Implementation)
• **Sub-phase C1:** Select component to test and match it to one or more ARIA APG patterns • **Sub-phase C2:** Define testing process objective, scope, and select testing tools • **Sub-phase C3:** Identify test requirement candidates by studying the ARIA APG pattern or patterns, and use these requirements to formulate test cases • **Sub-phase C4:** Control that the proposed test cases do not interfere with the ARIA in HTML specification* • **Sub-phase C5:** Set up the imported, automated accessibility testing tools for the selected component • **Sub-phase C6:** Write automated accessibility tests for the component using the test cases identified in sub-phases C2 and C3 • **Sub-phase C7:** Evaluate the test coverage of the component using the objective and scope definitions From Sub-phase C2. Satisfied with test coverage *YES/NO*: *YES*: Go to Sub-phase C1 *NO*: Go to Sub-phase C3 **Links to documents not included in table, can be found in Appendix B*

In order to implement automated accessibility testing of the web application's components, Phase C – *Test Execution* was followed. Stemming from a lack of resources— most pressingly time—the decision was made that the partial implementation would only use this phase of the method. Phase C is an iterative phase, consisting of six sub-phases. The phase is repeated until every component making up the web application includes a set of automated accessibility tests, both imported and self-written.

Because of the fact that Sub-phases C5, C6, and C7 are dependent on work carried out in Phase B – *Make a Test Plan*, a slightly altered version of Phase C had to be used for the partial implementation. This version of Phase C with an altered Sub-phase C6 is presented above in Table 6.1. As the table shows, the partial implementation consists of seven sub-phases, and the iterative nature of Phase C remains.

6.2 Results of Partial Implementation of the Preliminary AAT-WAC Method on the Blixtvakt Web Application

This section presents the results of the process of working through the altered version of Phase C of the preliminary AAT-WAC method, visible in Table 6-1, in order to implement automated accessibility testing for each of the components making up the real-world web application *Blixtvakt*. The results of conducting the Sub-phases *C1* to *C7* are presented in Section 6.2.1-6.2.7, respectively.

6.2.1 Sub-phase C1

Iterating through Phase C until the whole web application had been tested led to nine different components being identified and selected for testing. Each of these nine components could be matched to one or more ARIA APG patterns.

Table 6-2: Results of matching each of Blixtvakt's components to one or more ARIA APG patterns

Component Number	ARIA APG Patterns Deemed an Exact Match	ARIA APG Patterns Deemed a Partial Match
1	-	Meter Pattern
2	Slider Pattern	• Button Pattern • Dialog Pattern • Disclosure (Show/Hide) Pattern
3	Menu Button Pattern	-
4	Menu Pattern	-
5	-	• Dialog (Modal) Pattern • Link Pattern
6	-	Dialog (Modal) Pattern
7	-	Dialog (Modal) Pattern
8	• Menu Button Pattern • Menu Pattern	Dialog (Modal) Pattern

The results of this pattern matching process can be viewed in Table 6-2, which is included in this section. As can be seen in the table, the components have had their names generalized to a number, as including their programmatic names from the application codebase would just cause confusion. As the table also illustrates, three of the eight components were deemed a partial or full match to more than one ARIA APG pattern.

6.2.2 Sub-phase C2

The objective of the testing process, that is its specific goals and outcomes, were identified as implementing automated accessibility testing for each of the identified components. Furthermore, running the tests, as well as documenting and interpreting their results should be presented in such a way that the commissioning company can understand the process. If these goals are reached, the outcome is a significant increase in the understanding of the accessibility issues of each component.

The scope of the testing process was defined as identifying as many accessibility requirements as possible from the chosen ARIA APG patterns, and that each of these requirements should be converted to test cases that could then further be converted into automated tests.

Testing tools and technologies were also selected. This included—but was not limited to—selecting a software framework in which to write the automated tests, an external tool for running imported, automated accessibility tests, and a tool for storing and presenting the test results. As an example, the Cypress software framework was chosen as the tool in which the tests were written.

6.2.3 Sub-phase C3

Sub-phase C3 - *Identify Test Requirement Candidates by Studying the ARIA APG Pattern or Patterns and Use These Requirements to Formulate Test Cases*, resulted in a large amount of test requirement candidates gathered for each of the web application's components. Each of the ARIA APG patterns that the components were matched to yielded requirements that could be used. Documenting the gathering of requirements proved difficult as the researcher's increasing knowledge of the subject, as well as the testing tools, led to the number of gathered requirements constantly changing. All of the gathered requirements could be converted into functioning test cases.

6.2.4 Sub-phase C4

Sub-phase C4 - *Control that the Proposed Test Cases Do Not Interfere with the ARIA in HTML Specification*, resulted in very little work, as the architecture of the chosen web application Blixtvakt meant that very few requirements from the *ARIA in HTML* specification were applicable to the testing process. This is because Blixtvakt uses technological solutions that mean that test cases based on ARIA APG accessibility requirements do not interfere with their HTML solutions. The architecture of the Blixtvakt application causes many other accessibility issues, but none that were relevant to this sub-phase.

6.2.5 Sub-phase C5

The work conducted in Sub-phase C5 - *Set Up the Imported, Automated Accessibility Testing Tools for the Selected Component*, consisted of adding the imported accessibility tests to the component test suite of each component, as well as determining how to display and document the results of the tests. The decision was made to include the results of the imported tests with the results of the self-written tests, as well as documenting them together as well. This meant configuring the imported tests so that their results were displayed within the Cypress software framework, where the self-written tests are automatically displayed and documented. The tests also had to be configured so that the results were saved inside the Cypress test documentation, which enables the results to be fetched for future use.

6.2.6 Sub-phase C6

During Sub-phase C6 - *Write Automated Accessibility Tests for the Component Using the Test Cases Identified in Sub-phases C3 and C4*, self-written automated component tests based on the gathered test cases were written. The testing process scope defined in Sub-phase C2 was not met as the researcher was not proficient enough in Cypress to convert every gathered test case into a set of automated tests within the allotted time. While enough automated tests were written for each component for the researcher and Zenon AB to feel that the knowledge of the application's accessibility deficiencies had been improved, there still remained much possible work to be done for this sub-phase to be considered complete.

6.2.7 Sub-phase C7

Sub-phase C7 - *Evaluate the Test Coverage of the Component Using the Objective and Scope Definitions from Sub-phase C2*, as with all the previous ones, was repeated for each component selected for testing. For all components, the objective was deemed as being met. That means that there had been automated accessibility testing implemented, that running the tests and then viewing and documenting the results were understandable to the commissioning company, and finally that the knowledge of the accessibility issues of the component had been significantly increased. The scope had not been met for any component, as there remained gathered test cases that had not been converted into one or more automated tests.

6.3 Results of the Evaluation of the Partial Implementation

This section presents the results of the researcher evaluating the partial implementation of the preliminary AAT-WAC method using the evaluation model described in Section 4.1. This evaluation only concerns the altered version of Phase C and its seven sub-phases. An overview of Phase C can be seen in Table 6.1. This evaluation model consisted of five evaluation criteria: (1) *Semantic Correctness*, (2) *Syntactic Correctness*, (3), *Usefulness*, (4) *Method Adaptability and Flexibility*, and (5) *Appropriateness of the ARIA APG Design Patterns as a Requirements Resource for Tests*. The results of evaluating the partial implementation by these criteria are presented in Section 6.3.1-6.3.5, respectively.

6.3.1 Semantic Correctness

This evaluation criterion was used to evaluate the semantic correctness of the method. The researcher felt that the partial implementation revealed that the meaning of all the sub-phases of Phase C were not clear, and that they were not always easy to follow. Especially sub-phase C5, wherein the imported automated tests should be set up was deemed insufficient, but problems were also identified in other sub-phases. The sub-phases and their descriptions can be viewed in their entirety in Appendix B.

6.3.2 Syntactic Correctness

When evaluating the syntactic correctness of the method, the researcher found that no sub-phase of Phase C felt redundant when using the method to conduct a testing process. The sequence of the included sub-phases was also found to be appropriate. However, Phase C did not address some aspects of its domain that it should. One example is identifying if specific technological solutions used by the component that is to be tested make one or more sub-phases of Phase C much harder to carry out.

The most glaring weakness found concerning syntactic correctness was that the method provided no guidelines for how and when to check that the displaying and documenting of the component tests results were handled correctly. After Sub-phase C6, in which the automated tests for the component are written, the method moves on directly to evaluating the test coverage in Sub-phase C7. Handling the test results could be addressed by a step inserted between these two sub-phases.

6.3.3 Usefulness

This evaluation criterion was used to evaluate the usefulness of the method. The researcher believes the preliminary AAT-WAC method to be useful in implementing automated accessibility testing of web application components. Its usability could be improved by addressing its weaknesses, such as not providing guidelines for grouping potential test cases by importance or providing guidelines for what to look for when attempting to match a component to an ARIA APG design pattern.

6.3.4 Method Adaptability and Flexibility

When evaluating the method's adaptability and flexibility, the researcher found that Phase C of the method was satisfactorily adaptable and flexible. No sub-phase presupposed that a certain technological solution was used. A possible objection might be the presupposition of a component-based architecture, but since the method clearly states that it is only intended for web applications built using a component-based architecture, this objection was not deemed as valid.

6.3.5 Appropriateness of the ARIA APG Design Patterns as a Requirements Resource for Tests

This evaluation criterion was used to evaluate the appropriateness of using the ARIA APG patterns as a source for test requirements. The partial implementation revealed that the ARIA APG design patterns well-suited as the sole source of accessibility requirements from which automated component tests could be written. Enough patterns exist that almost every relevant sort of component used in modern web development can be considered a full or partial match to at least one pattern. Furthermore, each pattern consists only of concrete, specific requirements that can fairly easily be converted into usable test cases. While other specifications could be investigated as part of future research, the result of this evaluation was that the ARIA APG design patterns should keep their role in the improved AAT-WAC method.

7 Evaluation of the Preliminary AAT-WAC Method Using Interviews

This section presents the results of evaluating the preliminary AAT-WAC method by interviewing three employees of Zenon AB with relevant expertise in the domain. The interviews were based on the evaluation model described in Section 4.2. The method was presented to the interviewees by providing them with both the method documentation and a presentation of the partial implementation described in Chapter 6. The chosen presentation approach is described in Section 7.1. As can be seen in the evaluation model, the interviews evaluated the method by five different evaluation criteria. These evaluation criteria are: (1) *Interviewee Appropriateness*, (2) *Semantic Correctness*, (3) *Syntactic Correctness*, (4) *Usefulness*, and (5) *Method Adaptability and Flexibility*. They are presented in Section 7.2-7.4, respectively.

7.1 Presenting the Preliminary AAT-WAC Method to the Interviewees

The method was presented using a combination of the method documentation as well as providing the interviewees with a presentation of the partial implementation of the method, detailed in Chapter 6. The method documentation can be viewed in its entirety in Appendix B, and a concise summary can be found in Chapter 5. The presentation of the partial implementation of the method on one of Zenon AB's own web applications consisted of the researcher providing a brief explanation of how the method had been used for guidance when implementing the testing, as well as displaying the resulting tests themselves, together with their results and how the results were stored. Other aspects of web content accessibility testing pertaining to the partial implementation were also discussed with the interviewees during this presentation.

7.2 Interviewee Appropriateness

The answers from all three interviewees revealed that they had the needed experience to be considered appropriate choices by the sampling method described in Section 3.5. All three interviewees work as ICT consultants. The first interviewee held the role of junior developer and had had this role for approximately one year. The second interviewee held the role of developer, a role which they assumed around two years ago. The third interviewee held the role of consultant manager and had previously held the role of system developer. The third interviewee had worked in their current role for around six months and had held their previous role for around four years.

They all had industry experience in web development and could understand all the concepts and techniques mentioned in the preliminary AAT-WAC method documentation. The knowledge and understanding of automated web application testing varied, with only one interviewee having previous experience from an industrial setting. But, as the sampling method describes, this is not a threat to interviewee appropriateness for this evaluation. All three interviewees also had previous experience of working with the *Blixtvakt* web application, which meant that they could form a deeper understanding of the accessibility testing in a fairly short amount of time.

7.3 Semantic Correctness

All three interviewees were of the opinion that the preliminary method was semantically correct, overall. The included phases and sub-phases were deemed to be relevant, and most of them were understandable enough so that they could be followed by web developers with varying preexisting knowledge of the subject. However, one interviewee brought up that Sub-phase C1, where a component of the web application that is being tested is selected and then matched to an ARIA APG pattern was not understandable enough. The interviewee brought up that component complexity can vary to a great degree, and that the method could take steps to provide further information on how to properly select components for ARIA APG pattern matching.

7.4 Syntactic Correctness

When it came to the potential redundancy of the phases and sub-phases of the method, all three interviewees agreed that none of them could be considered fully redundant. As for the appropriateness of the order of the sub-phases, one interviewee thought that the method should make it clearer that all of the sub-phases in Phase A—wherein the required perquisite knowledge of web content accessibility is gathered—are voluntary, and that sub-phases can be modified or even skipped altogether if the user so wishes. The interviewees were also asked if there were important aspects of web application testing that the method didn't address, and one interviewee brought up that the method should make it clearer that it is only concerned with automated testing, and that no consideration had been given to providing guidelines for monitoring how the automated testing affected the web application's performance.

7.5 Usefulness

When it came to the usefulness of the preliminary method, all interviewees agreed that it was useful, but with certain caveats. This means that when the interviewees were presented with the purpose of the method, to provide guidance in implementing automated accessibility testing of web application components, they all concurred that overall, the method was useful. One interviewee discussed the fact that real-world web application development projects are so varied that even a method providing looser guidelines might not always be relevant.

7.6 Method Adaptability and Flexibility

The method's adaptability was considered satisfactory by all three interviewees. When presented with the scenario of using the method with a web application built using some other component-based software framework than Angular, all interviewees agreed that the method could be used as it is. Similarly, if important testing tools were different, such as the testing software framework being replaced, all interviewees thought the method was still functional. All interviewees thought the method was flexible enough to allow for its users to make modifications to it, or skip parts of it entirely, but as answers in the previous evaluation criteria sections revealed, this flexibility needed to be explained more clearly to method users.

8 Improved AAT-WAC Method

This chapter presents the improved AAT-WAC method. The changes can be viewed in full in Appendix C. This improvement is based on the evaluation of the method described in Chapter 6, together with the evaluation described in Chapter 7. The improved method is the result of the *Improved Design* research phase, which is described in detail in Section 3.3.4. First, Section 8.1 presents an overview of the new and improved AAT-WAC method. Section 8.2 describes the improvements to the method that do not belong to one of the method's three main phases. The changes made to (1) *Phase A*, (2) *Phase B*, and (3) *Phase C* of the method are presented in Section 8.3-8.5, respectively. Last, Section 8.6 presents the result of evaluating the improved method from the researcher's perspective.

8.1 Overview of the Improved AAT-WAC Method

To present an overview of the improved AAT-WAC method, a new diagram was created. The old overview diagram, which can be seen in Figure 5-1, was reworked into this new diagram based on the feedback gathered from the evaluations. The new diagram presenting an overview of the method, which can be viewed in Figure 8-1., has been heavily altered and more accurately reflects the important aspects of the methods.

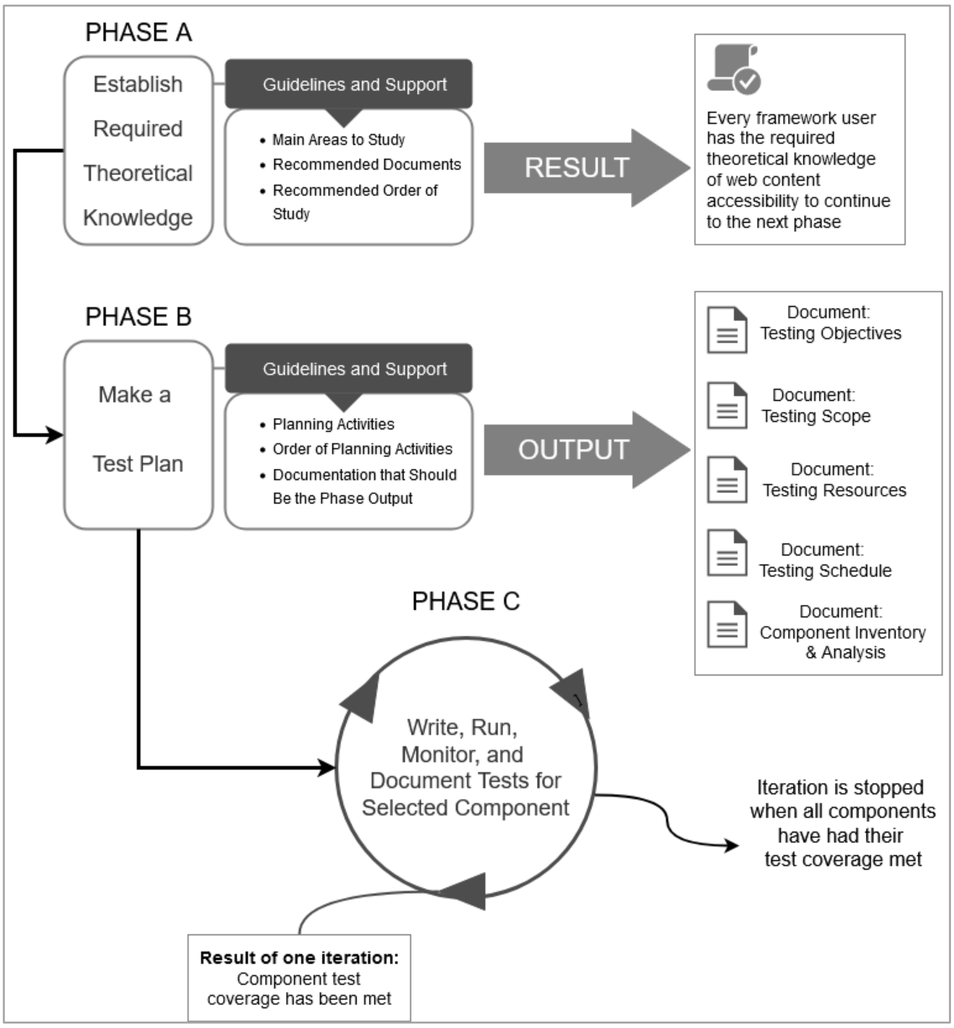

Figure 8-1: The improved AAT-WAC method

Compared to the old diagram that accompanied the preliminary method, it is clearer what the concrete results of each phase are, and that the method provides structure, guidelines, and recommendations instead of clearly defined steps that must be followed to the letter. The improved method also makes it clearer to the user that the third and final phase is iterative, in contrast to the two first phases. Importantly, the overview diagram also contains less information in the form of text than the old version. An overview diagram is supposed to provide a general overview that makes the more detailed documentation easier to follow. The evaluations revealed that the old overview diagram contained too many details and made the method and its documentation difficult to understand.

Further changes to the method resulting from the evaluations described in Chapter 6 and Chapter 7 that do not belong to a specific phase were also made. First, an introduction that concisely presents the prerequisites for the method to be usable was added to the documentation. This introduction addresses what is meant by a component-based architecture in the domain of web application development and provides a more comprehensive introduction to the ARIA APG than the preliminary method. Secondly, the three phases of the method are briefly introduced in a section added just after the general method introduction. These changes can be viewed in Appendix C.

8.2 Changes to Phase A

The changes to Phase A can be seen in Table 8-1, which is included below. In contrast to the previous version of Phase A, the result and end goal of the phase are now instantly visible to the method user. The phase also makes it clear that if the method user wishes to reach the end goal of the phase using their own study plan, they are free to do so, but that the method provides guidelines in form of major areas of web content accessibility to study, and the order in which to study them. A detailed presentation of Phase A can be viewed in Appendix C.

Table 8-1: Phase A of the improved AAT-WAC method

***Phase A: Establish Required Theoretical Knowledge** **Result:** Every method user possesses the required knowledge of *web content accessibility* to understand the rest of the method The end goal of the phase if for each user to understand the *ARIA APG* and its different resources	The Method provides two recommendations in this phase: 1. Major areas of web content accessibility to study and the order in which to study them 2. One introductory document for each area of study that is meant to provide a comprehensive overview If you do not wish to follow these recommendations, you need to identify your own major areas of study, and find your own documents for these areas
Method Recommends: Major Areas of Study, In Order: 1. Basic understanding of web content accessibility 2. WCAG 2.1 3. WAI-ARIA 4. ARIA APG 5. ARIA APG Practices	**Method Recommends: One Document for Each Major Area of Study, In Order:** 1. *Introduction to Web Accessibility*, by W3C 2. *WCAG 2 Overview*, by W3C 3. *WAI-ARIA Overview*, by W3C 4. *Introduction to ARIA APG*, by W3C 5. *APG Practices*, by W3C
*Links to documents not included in table. Links can be viewed in Appendix C.	

8.3 Changes to Phase B

The changes to Phase B can be viewed in Table 8-2, which is included below. Following feedback from the evaluations of the preliminary method, it was decided to change the different steps of the phase being called *sub-phases* to being called *activities*. As with Phase A, Phase B also makes it clear that the method user can construct a test plan using a different approach than what the phase recommends, but that this choice makes it more difficult to properly make use of Phase C.

A new activity has also been added to Phase B. This activity—fourth in order—instructs the method user to list and analyze all the components of the web application, and then document which of those components should be tested in the next phase. A detailed presentation of Phase C can be viewed in Appendix C.

Table 8-2: Phase B of the improved AAT-WAC method

<table>
<tr>
<td>Phase B: Make a Test Plan

Result: The method user has created a plan for the testing process that will be conducted in Phase C

Output: Documentation of the created test plan</td>
<td>The method provides two different recommendations in this phase:

1. Five different planning activities that when carried out addresses one major area of test planning
2. The order in which to perform these activities

If you do not wish to follow these instructions you need to construct your own testing plan, and then document it the way you see fit. Creating you own testing plan makes it more difficult to follow Phase C</td>
</tr>
<tr>
<td colspan="2">Method Recommends: Test Planning Activities to Perform, In Order:

1. Define and document the objectives of the testing process
2. Define and document the scope of the testing process
3. Identify and document the resources needed for the testing process
4. List and analyze all the components of the web application that shall be tested, and document this list and corresponding analysis
5. Create and document a testing process schedule</td>
</tr>
</table>

8.4 Changes to Phase C

The changes to Phase C can be viewed in Table 8-3 and Figure 8-2, both of which are included below. As with Phase B, the different sub-phases have been renamed to activities. Phase C now clearly states that it is iterative, as well as providing the user with the result of completing one iteration, and the result of completing all iterations. Phase C now also makes it apparent to the method user what input is needed in order for the phase to work. In combination with those changes, the phase also includes a flowchart which makes the iterative nature of the phase understandable, in contrast to the diagram of the preliminary method which evaluations revealed was not understandable enough. A detailed presentation of Phase C can be found in Appendix C.

Table 8-3 presents an introductory overview of the new and improved phase. As the table illustrates, the fact that the phase is iterative and cannot be completed without the required input is clearly evident. Because of the iterative nature of the phase, the actual workflow that is carried out during the phase has been illustrated in a separate flowchart, which can be viewed in Figure 8-2. As can be seen in Figure 8-2, one iteration of the phase is carried out for each component that should be tested, and during each iteration the method user has to make three decisions in the form of yes/no questions that guide the workflow of the iteration.

Table 8-3: Introduction to Phase C of the improved AAT-WAC method

<table>
<tr><th>Phase C: Write, Run, and Document Tests</th><th rowspan="4">Required Input

• Input #1: A list and analysis of all the components that should be tested. The method recommends producing this list and analysis by carrying out Activity #4 of Phase B
• Input #2: Documentation of the required test coverage for each component. The method recommends producing this documentation by carrying out Activity #2 of Phase B
• Input #3: Documentation of all the resources that shall be used in the testing process, such as the testing software framework and imported automated accessibility tests. The method recommends producing this documentation by carrying out Activity #3 of Phase B
• Input #4: The ARIA APG design patterns*</th></tr>
<tr><td>This is an iterative phase. It is repeated until every component listed as a result of Activity #4 of Phase B has had its test coverage met</td></tr>
<tr><td>Result of one iteration: Automated accessibility testing has been implemented for the selected component</td></tr>
<tr><td>Result after all iterations have been completed: Automated accessibility testing has been implemented for all the listed components</td></tr>
<tr><td colspan="2">*Links to online resources not included in table. Can be found in Appendix C</td></tr>
</table>

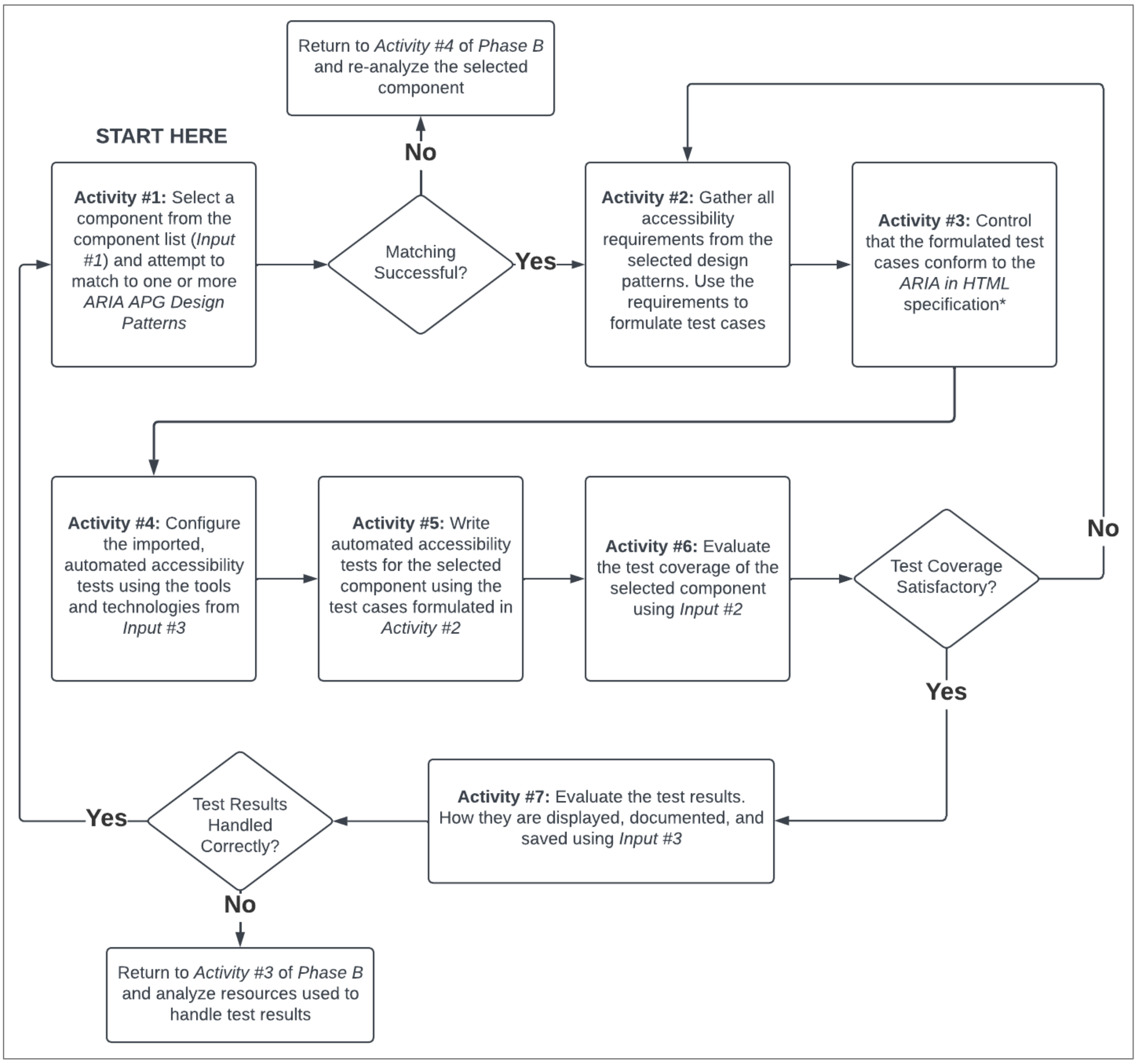

Figure 8-2: **Flowchart of the activities of Phase C of the improved AAT-WAC method**

8.5 Evaluation of the Improved AAT-WAC Method from the Researcher's Perspective

This section presents the result of the method researcher evaluating the improved method using the evaluation model described in Section 4.1. This is the same evaluation model that was used to evaluate the partial implementation of the method in an industrial setting.

8.5.1 Semantic Correctness

Based on the feedback from the two previously conducted evaluations, it is the researcher's opinion that the semantic correctness of the method has been increased. The use of the word *sub-phase* has been replaced with using the word *activity*, which is more in line with how methods are supposed to function. Each of the method's phases are clearer about what the result of each phase is, as well clearly stating the needed input. Importantly, the improved method also explicitly exemplifies the considerable freedom the method user has when following the phases. The graphical representation of the overall structure of the method has been made more intuitive, and it is more readily apparent what purpose each phase serves.

8.5.2 Syntactic Correctness

The improved method should be more syntactically correct, as aspects have been added to it that the previous evaluations revealed were missing. An activity where the method user creates an inventory of the components that should be tested has been added to Phase B, and in Phase C an activity was added that addresses the fact that the preliminary method provided no guidance for how to evaluate the component testing results.

8.5.3 Usefulness

It is the researcher’s opinion that the usability of the improved method has mainly been increased by adding the new activities described in Section 8.5.2, but also by adding a more descriptive introduction that clearly states what the method is used for, and what it is not used for. The improved graphical representations of the method should also increase usability.

8.5.4 Method Adaptability and Flexibility

Due to the improved method making it more apparent where and when method users can use their own solutions, as well as making sure there is guidance for handling different types of components, the method should be more adaptable and flexible.

8.5.5 Appropriateness of the ARIA APG Design Patterns as a Requirements Source for Tests

No changes made to the method should affect this evaluation criteria. The evaluation made after the partial implementation of the method in an industrial setting—which can be viewed in Section 6.2—is still valid for the improved method. The ARIA APG is mentioned more often in the improved method, and it is more apparent that it is of huge importance, but that is not really relevant to this particular evaluation criterion.

9 Analysis and Discussion

This section presents the analysis and discussion of the research results presented through Chapters 5-8. First, Section 9.1 analyses the partial implementation of the preliminary AAT-WAC method in an industrial setting and the evaluation of that implementation. Second, Section 9.2 provides analysis of the full evaluation of the preliminary method that was performed by interviewing employees with relevant expertise of the commissioning company. Then, Section 9.3 details the analysis of the improved AAT-WAC method, alongside its evaluation. In Section 9.4, the research validity threats are analyzed in the context of the results as a whole. Last, Section 9.5 provides a discussion of all the results of the research process.

9.1 Analysis of the Partial Implementation of the Preliminary AAT-WAC Method and its Evaluation

The results of the partial implementation of the preliminary method were used to evaluate the method from the researcher's perspective, as opposed to the later evaluation which evaluated the method from the user perspective. When analyzing this evaluation, there are several important aspects to consider.

The most important aspect is the fact that the whole method was not evaluated. Due to resource constraints—time being the most pressing one—only parts of the method could be implemented. This means that the evaluation of the method from the researcher's perspective is not complete. As stated in Section 4.1, for a method to be evaluated to a satisfactory degree, it should be fully evaluated from the researcher's perspective. Therefore, this is a major weakness of the evaluation, which should be remedied if future research is conducted. In such a remedy, the whole method would be implemented in an industrial setting. The advantages of a full implementation are further discussed in Section 10.3

Another important aspect is the enormous impact on the evaluation made by the choice of web application on which to partially implement the method. Due to resource constraints, both for the researcher and the commissioning company, only one web application was chosen. Evaluating the implementation of the method in an industrial setting would be greatly improved by instead making use of a set of web applications, chosen to expose the method to different environments and industrial realities.

9.2 Analysis of the Full Evaluation of the Preliminary AAT-WAC Method Using Interviews

When interviewing three of Zenon AB's employees with relevant expertise about the preliminary AAT-WAC method, both the method documentation and the partial implementation were used to present the method. The resulting evaluation should be considered as encompassing the entire method and proved the preliminary method to largely comply with the evaluation criteria. Yet, there are major weaknesses present in this evaluation, as well as other areas of concern.

First, three interviewees make for a small sample. The most straightforward improvement of the evaluation—and perhaps the simplest—would be to increase the number of interviewees. Adding to this, the interviewees, while hugely hospitable and kind to the researcher, did not have enough time to truly study the method, its documentation, and its partial implementation. If the method were to be further researched, and more time were available, it would be very reasonable to combine the increased number of interviewees with a more comprehensive presentation of the method.

Second, while the interviewees came from a reasonably varied background in terms of previous roles, they were still fairly similar in this regard. If the number of interviewees is increased in further research, the research methodology should also take into account to more properly investigate the interviewee credibility so that the background of the interviewees could be factored into the research results.

Last, this evaluation suffered from the same fact of the partial implementation, in that only one web application could be used in the presentation. As with the partial implementation, having used the method to implement automated accessibility testing of the components of other web applications would have ensured that the interviewees could have formed a greater understanding of the method.

9.3 Analysis of the Improved AAT-WAC Method

Using the results from the two evaluations of the preliminary method, the researcher attempted to create an improved AAT-WAC method. Enough concrete weaknesses were identified during the evaluations of the preliminary method that this venture should not be considered guesswork, however it has major weaknesses that are more severe than the weaknesses discussed in Section 9.1 and 9.2

The foremost of these weaknesses is that due to a lack of time, the improved AAT-WAC method could only be evaluated from the researcher's perspective, and without implementing the method in an industrial context. This leads to a number of flaws in the results. These flaws could be considered major enough to decide that the researcher does not have enough knowledge to know if the method has actually been *improved*. However, the word *improved* was still used, as this was the intention of the researcher when creating the method, as defined in the research methodology underpinning the whole thesis.

The lack of a user evaluation, or evaluation by implementation in an industrial context, severely damages the credibility and dependability of the method. While real, concrete findings were used to create the improved version of the method, the fact that the preliminary version underwent user evaluation means that it might be the preferable choice if one of the methods were to be chosen to be used in an industrial setting.

Considering the lack of user evaluation of the improved method, the obvious target of future research would be to properly evaluate it. If such an evaluation was made, the same issues brought up in Section 9.1 and 9.2 should be considered. A more thorough evaluation of the method would hopefully indicate that it indeed is an improvement of the preliminary version.

Lastly, one important aspect shared by both the preliminary and improved methods that should be analyzed in future research is the dependence on the ARIA APG, the online resource that makes the whole method function. Theoretical knowledge of web content accessibility on a general scale is fairly stable. Standards, specifications, and recommendations will certainly change in the future, but the method was deemed flexible enough after evaluation to manage such changes. However, when it comes to the ARIA APG, such flexibility is hard to achieve. One of the problems addressed by this thesis is the difficulty of finding concrete, specific accessibility requirements for web content that enable web developers to formulate usable test cases for automated testing. The ARIA APG solves this by being one of few sources providing such requirements for an impressive number of different design patterns used in web development. If the ARIA APG were to drastically change, or worse yet disappear, the method might be rendered unusable.

9.4 Analysis of Validity Threats

In Section 3.6, four qualitative validity criteria were discussed. These criteria were considered at different stages during the whole research process, and it is appropriate to also provide analysis of how they were handled. By analyzing these validity criteria, the results of the thesis can be interpreted—and criticized—with greater understanding. The validity criteria analysis is also important as it makes conducting further research less troublesome. The four validity criteria that are analyzed in this discussion are: (1) *Credibility*, (2) *Transferability*, (3) *Reliability*, and (4) *Objectivity*.

The credibility of the research conducted is mainly affected by two factors that were under the researcher's control. Firstly, the credibility of the interviews should be considered. Increasing the interview's credibility by increasing the number of interviewees would also increase the credibility of the evaluation. Second, the credibility of the partial evaluation and subsequent evaluation should be analyzed. Here, the credibility would be increased by implementing the method on more than one web application, together with implementing the method in full instead of partially.

The transferability could also have been increased by using more than one approach. As with the credibility validity criterion, implementing the method in full on more than one web application would lead to the interviewees being able to provide a more comprehensive analysis of the method's possible transferability. As discussed in Section 9.2, a more varied interviewee background would also be advantageous. If there were interviewees present from several different companies or organizations, the interview results would be considered more reliable when it came to their analysis of transferability.

The reliability of the two created method is not considerable. Reliability of qualitative research is notoriously difficult to achieve and can be improved by repeating the research process in different contexts and comparing the results. The nature of this research process—most importantly the resources available to it—made such an endeavor unfeasible. If the research process were to be repeated, valuable knowledge that had been learned could be used to make slight improvements to the reliability, but for major improvements to be made, the research process would still have to be repeated in different contexts.

The objectivity of the research results is—in the opinion of the researcher—considerably low. The researcher's substantial lack of experience in conducting qualitative research in the chosen research area will have affected all areas of the research process, in combination with qualitative research being susceptible to bias, and even more so if the research is carried out by one person. Major areas of the research process where objectivity was a validity threat were the design of the interview questionnaire, the design of the presentation of the method, and the evaluation from a researcher's perspective that was carried out after the partial implementation of the method in an industrial setting. Efforts to increase the objectivity were made, mostly by using the literature study research phase to increase the researcher's knowledge of qualitative research and using this increased knowledge to identify suitable major works on which to base the research process.

9.5 Discussion of Results

The methods created during this research process are meant to aid people working in web development with implementing automated accessibility testing of the components that make up a web application built using a component-based architecture. The *Problem Verification* research phase—which is described in Section 3.3.1—has shown that no such methods exist, but has the research that was undertaken also shown that such methods are needed? Or that they address real problems that web developers face?

When interviewing employees of Zenon AB with relevant expertise in the area, as well as when reading works related to the subject, it became clear that web content accessibility—and especially web content accessibility testing—still is an area that seem intimidating to many web developers. But is a methods specifically aimed at facilitating automated testing of accessibility issues the best way to help web developers inexperienced in web accessibility to get started?

It is the opinion of the researcher that the research revealed that such a methods could be an early step in such a process, but that even more general methods or methodologies should make up the *very first* step. It is only after getting familiar with web accessibility that the realization dawns that automated testing could be hugely helpful in managing the challenges of accessibility. It is only when that realization has been had that a method like AAT-WAC seems interesting. Presenting the AAT-WAC method to someone with very little experience in web accessibility runs a high risk of making the subject seem even more confusing.

On a more basic level, one major takeaway from the research is that the ARIA APG—the online resource that serves as a foundation for producing test cases in the method—is an underappreciated resource in the quest for a more accessible internet. During the work of the literature study, a large number of candidates were analyzed to see if they were suitable for finding concrete, specific accessibility requirements for web content that could be used to produce automatic tests. The research revealed that such sources were rare, and if they existed, they were often of questionable repute, poorly organized, or poorly written. Without the ARIA APG the whole method and its purpose might have been scrapped, because then no suitable source for finding the type of requirements needed to easily write automated tests would have been identified.

It would be intriguing to conceive of ways in which to make the ARIA APG more well-known, and in doing so encourage other researchers to find more uses for it. Or even if an increased popularity could lead to the authors of the ARIA APG expanding its scope, or for other organizations or entities to create their own, similar resources.

10 Conclusions and Future work

Web accessibility is still sorely lacking. While there are many reasons for this, most of them not addressed by this thesis, one major reason is the perceived difficulty of implementing a structured approach to accessibility testing web applications. This sense of accessibility testing processes being difficult leads to organizations and developers being hesitant to even start trying to improve accessibility. This is the problem that this thesis attempted to address. If there was an available method applicable to component-based web applications, meant for implementing automated accessibility testing, more organizations, but especially individual developers, might feel it was easier to get started with accessibility testing.

If a method is to satisfy these demands it will need to outline the steps that need to be taken to start such a process, and it will also need to present and evaluate the different kinds of non-manual accessibility testing that is available at the moment. For the method to be ethically defensible it will need to present and analyze its possible faults and weaknesses so that entities that choose to utilize it can do so in the right manner.

The purpose of this thesis is to create and evaluate a method for automated accessibility testing of web application components that would address the problem discussed at the start of this chapter. The goal of this thesis is that the created method can be of use to a plethora of actors. Not only web developers that might use the method, or parts of it, but also other people interested in web accessibility. If the method proves useful, and future research based on its findings is carried out, it might help with creating a more accessible Web, and therefore society.

In order to achieve the goal of the thesis, a research methodology was developed. The development of this methodology was informed by consulting literature relevant to research design, research processes, and data collection. This development resulted in a research methodology based on qualitative research with an exploratory approach. Following this methodology resulted in an extensive literature study, a preliminary method being created, then implementing parts of that method on a real web application developed by Zenon AB. The preliminary method was then evaluated using interviews based on relevant evaluation criteria. This evaluation served as the foundation for the creation of the improved AAT-WAC method, which is the main result of this thesis.

Section 6.1 of this chapter presents conclusions drawn analyzing the work and results of this thesis. Section 6.2 describes the limitations of the thesis and its results. It addresses both the limitations defined at the start of the project, as well as those added during its lifetime. Section 6.3 discusses the possible future work that could be conducted to increase the understanding of the problems presented in this thesis. Finally, Section 6.4 contains the results of reflecting on the potential economic, social, and ethical aspects of this research.

10.1 Conclusions

The accessibility testing method created and evaluated for this thesis was based on two main types of non-manual accessibility testing, automated tests imported from an external source, and component tests written by the application developer. Thus, evaluating this method and its potential positive and negative effects largely consisted of attempting to evaluate these two test categories. The strengths and weaknesses of automated accessibility testing has been discussed in plenty of literature, while component testing is a fairly recent approach to testing web applications, and thus little to no literature discussing its usage for accessibility testing could be found.

The interviews with Zenon AB employees with relevant experience in web development revealed that automated accessibility testing, and a method providing concrete instructions on how to incorporate it into a web development project is useful. However, the method is less universally usable because of its usage of component testing. Component testing, as it is defined in this thesis, is still a fairly new approach to testing web applications built using a component-based architecture. This makes the method more specific, and less suited to be the very first resource someone consults when wanting to learn more about automated accessibility testing and its benefits.

The main takeaway from the evaluations carried out by the researcher—as well as the work carried out throughout the whole project—was that the greatest need for a method like AAT-WAC is a reliable source of web content accessibility requirements from which test cases for automated tests can be formulated. Most web accessibility standards, including the most dominant one, provide requirements that are too general and vague to be used as sources for automated tests. As briefly discussed in Section 9.5, finding the ARIA APG changed the trajectory of this thesis. Considering its impressive scope, the concreteness of its requirements, and that it is maintained by the most respected web accessibility organization, it is truly an enormously useful resource for writing automated accessibility tests for web content.

10.2 Limitations

This section attempts to draw conclusions about the limitations that were imposed on this thesis. Were the limitations chosen wise decisions? How did they affect the research conducted? First, Section 6.2.1 discusses the conclusions that were decided on before the research began. Second, Section 6.2.2 discusses the limitations that had to be introduced during the research process.

10.2.1 Limitations Defined Before the Research Process Started

This section analyzes the impact of the limitations that were defined before the research process was started. These limitations, and the motivation for their inclusion, are discussed in greater detail in Section 1.8. These limitations were as follows: (1) *Technologies*, (2) *Blixtvakt Application*, (3) *WCAG*, (4) *Zenon AB's Internal Infrastructure*, and (5) *Mobile Web Applications*.

- **Technologies:** The decision not to compare different technological solutions for tasks such as test automation and web application development was deemed an appropriate one. The resources needed were not there, and the method purposefully allows for using different technological solutions.
- **Blixtvakt Application:** The limitation to not test the full, real-world, Blixtvakt web application, and instead create a simpler prototype application was also deemed an appropriate one. The interviewees agreed that automated accessibility tests written for the prototype application would work equally well with the real application, but using this solution much less work setting up the real-world application's infrastructure was needed.
- **WCAG:** This limitation stipulated that the researcher should not conduct a complete accessibility audit according to the WCAG standard, but that the AAT-WAC method should only test for accessibility issues that can be discovered through non-manual testing. This limitation was also deemed appropriate as the research process revealed that there was indeed no time for a full audit.
- **Zenon AB's Internal Infrastructure:** This limitation stipulated that the researcher would not concern themselves with Zenon AB's internal infrastructure for web application testing. This limitation was also deemed appropriate after the research process was completed, as there would have been no available time to complete such a task.

- **Mobile Web Applications:** This limitation stipulated that the automated accessibility tests written during the research process should focus on web content being displayed on a computer screen, and not smartphones or other mobile devices. This limitation was also deemed appropriate as the researcher did not have time to include the more cumbersome process of adapting the tests to mobile devices.

10.2.2 Limitations Introduced During the Research Process

The first major limitation that was imposed during the research process was to not include the results of comparing the number of accessibility violations that the imported automated tests could discover with the number that the self-written tests could discover. This limitation was imposed as the researcher's knowledge of writing component tests with the chosen technological solutions was too limited to manage to turn every accessibility requirement into a corresponding automated tests. If, for example, the ARIA APG revealed twelve possible test requirements for a certain component, self-written automated tests for all twelve requirements would have to be written to make a fair comparison.

The second major limitation imposed was to not attempt to map out the relationship between the ARIA APG accessibility requirements with the requirements found within WCAG 2.1. While this might be possible, the researcher greatly underestimated the time it would take to create such a mapping. Combine the lack of time with the fact that a faulty mapping would be directly damaging to accessibility because it would mislead method users, and the limitation seemed an appropriate one.

10.3 Future work

There is plenty of future work that could be done to improve the understanding of the problem central to this thesis. Are accessibility testing process methods that are smaller in scope and can be implemented by a single developer a good idea, or is there a reason most approaches take a much wider approach? What are the possible advantages of accessibility testing using component tests? How can the technologies and infrastructure supporting component testing make it easier to use them for accessibility testing? Attempting to answer these, and many other similar questions, would not only help shed greater light on the issues encountered in the research presented here, but also similar problems and possibilities in accessibility testing, and web accessibility.

A specific example of great interest to the author would be to evaluate component testing focusing on accessibility not using an existing web application, but to build an application from the ground up where the aim is for the application to supply the researcher with opportunities to test for more kinds of accessibility issues. If the time and resources were sufficient, the research could identify all the WCAG requirements that automatic testing cannot test for, and then further reduce that group of requirements into a smaller group of requirements that could conceivably be tested for using component testing. Using the testing application these requirements would then all be used to evaluate component testing for accessibility in a much more extensive context.

Finally, as briefly discussed in Section 8.5, further research into the ARIA APG and its uses could be greatly beneficial to web content accessibility. Could the ARIA APG be expanded, or altered, in order to become an even more useful resource? What aspects of the ARIA APG can other web content accessibility standards and guides learn from in order to make it easier for testers to convert their requirements into automated tests?

10.4 Reflections

As the research revealed, there is no available method for automated accessibility testing of web application components. The usefulness of the AAT-WAC method can be divided into two categories.

First, the method, or an adapted version of it, can be used by web developers and testers to create more accessible web content. Second, the questions and findings revealed by the research process from which the method was created can contribute to a greater knowledge of web content accessibility and enable further research into important areas. It is the hope of the researcher that both the benefits of both these categories can contribute to a more accessible Web, and in doing so also create a more accessible society.

References

[1] W3C Web Accessibility Initiative (WAI), "Introduction to Web Accessibility," *Web Accessibility Initiative (WAI)*. https://www.w3.org/WAI/fundamentals/accessibility-intro/ (accessed Sep. 09, 2023).

[2] A. Firth, *Practical Web Inclusion and Accessibility: A Comprehensive Guide to Access Needs*. Berkeley, CA: Apress, 2019. doi: 10.1007/978-1-4842-5452-3.

[3] Myndigheten för delaktighet, "Statistik om personer med funktionsned-sättning," *MFD*, Sep. 12, 2022. https://www.mfd.se/resultat-och-uppfoljning/statistik-om-personer-med-funktionsnedsattning/ (accessed Sep. 09, 2023).

[4] World Health Organization (WHO), "Disability - Key Facts," *Disability*, 2023. https://www.who.int/news-room/fact-sheets/detail/disability-and-health (accessed Sep. 20, 2023).

[5] Sheri Byrne-Harper, *Giving a Damn About Accessibility*. UX Collective (ISSN: 2766-5267), 2021. Accessed: Sep. 08, 2023. [Online]. Available: https://www.accessibility.uxdesign.cc/

[6] World Wide Web Consortium (W3C), "Web Content Accessibility Guidelines (WCAG) 2.1," *Web Accessibility Content Guidelines (WCAG) 2.1*, 2023. https://www.w3.org/TR/WCAG21/ (accessed Sep. 20, 2023).

[7] WebAIM Institute for Disability Research, Policy, and Practice, "WebAIM: The WebAIM Million - The 2023 report on the accessibility of the top 1,000,000 home pages," *The WebAIM Million The 2023 report on the accessibility of the top 1,000,000 home pages*. https://webaim.org/projects/million/ (accessed Sep. 09, 2023).

[8] Y. Yesilada and S. Harper, Eds., *Web accessibility: a foundation for research*, 2nd edition. London: Springer, 2019. ISBN: 978-1-4471-7439-4.

[9] Mozilla Foundation, "Web Components - Web APIs | MDN," Jun. 11, 2023. https://developer.mozilla.org/en-US/docs/Web/API/Web_components (accessed Sep. 20, 2023).

[10] Hamir Nandaniya, "A Guide to Component-Based Architecture: Features, Benefits and more," *Maruti Techlabs*. https://marutitech.com/guide-to-component-based-architecture/ (accessed Oct. 01, 2023).

[11] J. Vom Brocke, A. Hevner, and A. Maedche, Eds., *Design Science Research. Cases*. Cham: Springer International Publishing, 2020. doi: 10.1007/978-3-030-46781-4.

[12] "Qualitative or Quantitative Research? | MQHRG - McGill University." https://www.mcgill.ca/mqhrg/resources/what-difference-between-qualitative-and-quantitative-research (accessed Sep. 27, 2021).

[13] "Exploratory Research - Research-Methodology." https://research-methodology.net/research-methodology/research-design/exploratory-research/ (accessed Sep. 28, 2021).

[14] Zenon AB, "App – Blixtvakt," 2023. https://blixtvakt.se/app/ (accessed Sep. 09, 2023).

[15] Institute of Electrical and Electronics Engineers (IEEE), "IEEE Code of Ethics," 2023. https://www.ieee.org/about/corporate/governance/p7-8.html (accessed Sep. 10, 2023).

[16] P. Baxter and S. Jack, "Qualitative Case Study Methodology: Study Design and Implementation for Novice Researchers," *Qual. Rep.*, Jan. 2015, doi: 10.46743/2160-3715/2008.1573.

[17] W3C Web Accessibility Initiative (WAI), "Evaluating Web Accessibility Overview," *Web Accessibility Initiative (WAI)*, 2023. https://www.w3.org/WAI/test-evaluate/ (accessed Sep. 24, 2023).

[18] M. Dowden and M. Dowden, *Approachable Accessibility: Planning for Success*. Berkeley, CA: Apress, 2019. doi: 10.1007/978-1-4842-4881-2.

[19] W. W. A. Initiative (WAI), "WAI-ARIA Overview," *Web Accessibility Initiative (WAI)*. https://www.w3.org/WAI/standards-guidelines/aria/ (accessed Dec. 12, 2023).

[20] "Accessible Rich Internet Applications (WAI-ARIA) 1.2." https://www.w3.org/TR/wai-aria/ (accessed Dec. 12, 2023).

[21] W. W. A. Initiative (WAI), "ARIA Authoring Practices Guide (APG) - Introduction," *Web Accessibility Initiative (WAI)*. https://www.w3.org/WAI/ARIA/apg/about/introduction/ (accessed Dec. 12, 2023).

[22] "Intro to Advanced ARIA," *Deque*, Apr. 25, 2017. https://www.deque.com/blog/advanced-aria/ (accessed Dec. 15, 2023).

[23] W3Techs - World Wide Web Technology Surveys, "Usage Statistics of JavaScript as Client-side Programming Language on Websites, September 2023," 2023. https://w3techs.com/technologies/details/cp-javascript (accessed Sep. 20, 2023).

[24] R. E. Johnson, "Documenting frameworks using patterns," in *Conference proceedings on Object-oriented programming systems, languages, and applications*, Vancouver British Columbia Canada, Oct. 1992, pp. 63–76. doi: 10.1145/141936.141943.

[25] Stack Exchange Inc, “Stack Overflow Developer Survey 2023,” *Stack Overflow*, 2023. https://survey.stackoverflow.co/2023/?utm_source=social-share&utm_medium=social&utm_campaign=dev-survey-2023 (accessed Sep. 09, 2023).

[26] K. Seth, A. Seth, and A. Tripathi, *Component-based systems: estimating efforts using soft computing techniques*, First edition. Boca Raton: CRC/CRC Press, Taylor & Francis Group, 2021. ISBN: 978-1-00-301388-4.

[27] K.-K. Lau and S. Di Cola, *An Introduction to Component-Based Software Development*, vol. 03. World Scientific, 2017. doi: 10.1142/10486.

[28] N. Kaul, *Implementing Automated Software Testing*. S.l.: ARCLER PRESS, 2023. ISBN: 978-1-77469-608-8.

[29] Yash Bansal, “Complete Guide To Automation Testing Frameworks [2023],” *LambdaTest*, Feb. 02, 2023. https://www.lambdatest.com/blog/automation-testing-frameworks/ (accessed Sep. 27, 2023).

[30] Devographics Collective, “State of JavaScript 2022: Testing,” *State of JavaScript 2022: Testing*, 2022. https://2022.stateofjs.com/en-US/libraries/testing/ (accessed Sep. 27, 2023).

[31] J. Palmer, C. Cohn, M. Giambalvo, C. Nishina, and B. Green, *Testing Angular applications*. Shelter Island, NY: Manning Publications, 2018. ISBN: 978-1-61729-364-1.

[32] Applitools, “Keynote: The State of the Union for Front End Testing,” *Automated Visual Testing | Applitools*, 2023. https://applitools.com/event/opening-remarks-keynote-the-state-of-the-union-for-front-end-testing/ (accessed Sep. 28, 2023).

[33] Cypress.io, Inc., “Announcing Cypress 10 with Component Testing Beta!,” 2022. https://www.cypress.io/blog/2022/06/01/cypress-10-release (accessed Sep. 28, 2023).

[34] E. Kinsbruner, *A frontend web developer’s guide to testing: explore leading web test automation frameworks and their future driven by low-code and AI*, [First edition]. Birmingham, United Kingdom: Packt Publishing, 2022. ISBN: 978-1-80324-967-4.

[35] W. W. A. Initiative (WAI), “ARIA Authoring Practices Guide (APG) - Patterns,” *Web Accessibility Initiative (WAI)*. https://www.w3.org/WAI/ARIA/apg/patterns/ (accessed Dec. 12, 2023).

[36] W3C Web Accessibility Initiative (WAI), “WCAG-EM Overview: Website Accessibility Conformance Evaluation Methodology,” 2023. https://www.w3.org/WAI/test-evaluate/conformance/wcag-em/ (accessed Sep. 24, 2023).

[37] DIAS GmbH, “BIK BITV-Test | Home,” 2023. https://www.bitvtest.eu/home.html (accessed Sep. 24, 2023).

[38] TecEd, Inc., “Accessibility Evaluation Methodology - TecEd,” 2023. https://www.teced.com/services/web-accessibility/accessibility-evaluation-methodology/ (accessed Sep. 24, 2023).

[39] W3C Web Accessibility Initiative (WAI), “Website Accessibility Conformance Evaluation Methodology (WCAG-EM) 1.0,” *Website Accessibility Conformance Evaluation Methodology (WCAG-EM) 1.0*, 2014. https://www.w3.org/TR/WCAG-EM/ (accessed Oct. 01, 2023).

[40] J. Vom Brocke, A. Hevner, and A. Maedche, Eds., *Design Science Research. Cases*. Cham: Springer International Publishing, 2020. doi: 10.1007/978-3-030-46781-4.

[41] J. W. Creswell, *Research design: qualitative, quantitative, and mixed methods approaches*, 4th ed. Thousand Oaks: SAGE Publications, 2014. ISBN: 978-1-4522-2609-5.

[42] K. S. Bordens and B. B. Abbott, *Research design and methods: a process approach*, 8th ed. New York: McGraw-Hill, 2011. ISBN: 978-0-07-353202-8.

[43] K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee, “A Design Science Research Methodology for Information Systems Research,” *J. Manag. Inf. Syst.*, vol. 24, no. 3, pp. 45–77, Dec. 2007, doi: 10.2753/MIS0742-1222240302.

[44] “IEEE Xplore.” https://ieeexplore.ieee.org/Xplore/home.jsp (accessed Oct. 28, 2023).

[45] “Document Search - Web of Science Core Collection.” https://www-webofscience-com.focus.lib.kth.se/wos/woscc/basic-search (accessed Oct. 28, 2023).

[46] “Scopus preview - Scopus - Welcome to Scopus.” https://www.scopus.com/home.uri (accessed Oct. 28, 2023).

[47] “Google Scholar.” https://scholar.google.com/ (accessed Oct. 28, 2023).

[48] R. M. Gilbert, *Inclusive Design for a Digital World: Designing with Accessibility in Mind*. Berkeley, CA: Apress, 2019. doi: 10.1007/978-1-4842-5016-7.

[49] R. C. Martin, J. Grenning, S. Brown, and K. Henney, *Clean Architecture: a craftsman’s guide to software structure and design*. Boston Columbus Indianapolis New York San Francisco Amsterdam Cape Town Dubai London Madrid Milan Munich Paris Montreal Toronto Delhi Mexico City São Paulo Sydney Hong Kong Seoul Singapore Taipei Tokyo: Prentice Hall, 2018. ISBN: 978-0-13-449416-6.

[50] P. Leloudas, *Introduction to Software Testing: A Practical Guide to Testing, Design, Automation, and Execution*. Berkeley, CA: Apress, 2023. doi: 10.1007/978-1-4842-9514-4.

[51] J. Rasmusson, *The way of the web tester: a beginner's guide to automating tests*. Raleigh, North Carolina: The Pragmatic Bookshelf, 2016. ISBN: 978-1-68050-183-4.

[52] J. W. Creswell, *Educational research: planning, conducting and evaluating quantitative and qualitative research*, Pearson new international edition, Fourth edition. Harlow, Essex: Pearson, 2014. ISBN: 978-1-292-02112-6.

[53] H. F. Hansen, "Choosing Evaluation Models: A Discussion on Evaluation Design," *Evaluation*, vol. 11, no. 4, pp. 447–462, Oct. 2005, doi: 10.1177/1356389005060265.

[54] M. N. Marshall, "Sampling for qualitative research," *Fam. Pract.*, vol. 13, no. 6, pp. 522–526, 1996, doi: 10.1093/fampra/13.6.522.

[55] R. K. Yin, *Case study research: design and methods*, 4th ed. Los Angeles, Calif: Sage Publications, 2009. ISBN: 978-1-4129-6099-1.

[56] A. K. Shenton, "Strategies for ensuring trustworthiness in qualitative research projects," *Educ. Inf.*, vol. 22, no. 2, pp. 63–75, Jul. 2004, doi: 10.3233/EFI-2004-22201.

[57] N. K. Denzin and Y. S. Lincoln, Eds., *The SAGE handbook of qualitative research*, Fifth edition. Los Angeles London New Delhi Singapore Washington DC Melbourne: SAGE, 2018. ISBN: 978-1-4833-4980-0.

[58] Sedera, Wasana, Gable, Guy, and Rosemann, Michael, "A success model for business process modeling: findings from a multiple case study.," *Proc. Eighth Pac. Asia Conf. Inf. Syst.*, pp. 485–498, 2004.

Appendix A: Problem Verification Sub-phase

This appendix documents the work done in the *Problem Verification* sub-phase of the *Literature Study* research phase. After conducting the literature study and consulting with the thesis supervisor Mira Kajko-Mattson, the researcher proposed that there was no method for implementing automated accessibility testing of web application components. However, it is not enough to simply propose this, it also had to be verified.

This verification consisted of searching the well-established research databases IEEE Explore, Scopus, and Web of Science using the following keywords: “web accessibility”, “component testing”, “method”, “automated testing”, and a combination of these.

A.1 IEE Explore

- (("Document Title":web accessibility) AND ("Document Title":component testing) AND ("Document Title":method)) OR ((“Document Title”:software accessibility) AND (“Document Title”:component testing))
- (("Abstract":web accessibility) AND ("Abstract":component testing) AND ("Abstract":method)) OR (("Abstract":software accessibility) AND ("Abstract":component testing) AND ("Abstract":method))

In total the above query strings yielded three results. None of the three results were closely related to the proposed research problem.

A.2 Scopus

- (TITLE(“web accessibility” AND “component testing” AND “method”) OR (“accessibility” AND “component testing” AND “method”))
- (ABS(“web accessibility” AND “component testing” AND “method”) OR (ABS(“accessibility” AND “component testing” AND “method”))

In total the above query strings yielded four results. None of the four results were closely related to the proposed research problem.

A.3 Web of Science

- ((TI=”web accessibility) AND (TI=”component testing”) AND (TI=”method”)) OR ((TI=” accessibility) AND (TI=”component testing”) AND (TI=”method”))
- ((AB=”web accessibility) AND (AB=”component testing”) AND (AB=”method”)) OR ((AB=” accessibility) AND (AB=”component testing”) AND (AB=”method”))

In total the above query strings yielded seven results. None of the results were closely related to the proposed research problem.

Appendix B: Preliminary AAT-WAC Method

This appendix presents the full version of the preliminary AAT-WAC method. A condensed summary of the method can be found in Chapter 5 of the thesis. The method consists of three main phases, and these main phases in turn consist of a set of sub-phases. These three main phases together with their sub-phases are presented in appendix sections B1-B3, respectively.

B.1 Phase A: Gain Prerequisite Knowledge of Web Content Accessibility

In this phase the method user makes sure that they have the perquisite knowledge of web content accessibility needed in order to carry out the subsequent phases of the method. This phase consists of five sub-phases, each recommending one document pertaining to web content accessibility. The order and content of these sub-phases are recommendations, and if the user already understands the concepts connected to one sub-phase they can move on to the next. This phase should only be performed once before continuing on with the subsequent phases.

Sub-phase A1: Introductory Document for Web Content Accessibility

After A1, the user should have a basic understanding of web content accessibility. There is a myriad of documents aimed at accomplishing this task, but the method recommends the *Introduction to Web Accessibility*, by W3C, as it is authored by a trusted source and conforms perfectly to the purpose of the sub-phase.

- **Recommended document**: https://www.w3.org/WAI/fundamentals/accessibility-intro/

Sub-phase A2: Introductory Document for WCAG 2.1

After A2, the user should have a basic understanding of WCAG 2.1, the completely dominant web content accessibility standard in use today. The method strongly recommends the official *WCAG 2 Overview,* by W3C, as it is accessible to people unfamiliar with the subject as well as being written by the authors of WCAG 2.1.

- **Recommended document:** https://www.w3.org/WAI/standards-guidelines/wcag/

Sub-phase A3: Introductory Document for WAI-ARIA

After A3, the user should have a basic understanding of the WAI-ARIA suite of web content accessibility standards. The method recommends the *WAI-ARIA Overview*, by W3C, as it is written by the authors of ARIA and provides a clear introductory overview to the subject.

- **Recommended document:** https://www.w3.org/WAI/standards-guidelines/aria/

Sub-phase A4: Introductory Document for the ARIA APG

After A4, the user should have a basic understanding of the ARIA APG, a diverse guide aimed at helping web developers create more accessible interactive web content. The method recommends the *Introduction to ARIA APG*, by W3C, as it is written by the authors of the ARIA APG.

- **Recommended document:** https://www.w3.org/WAI/ARIA/apg/about/introduction/

Sub-phase A5: Introductory Document for the ARIA APG Practices

After A5, the user should have a basic understanding of the official ARIA APG practices, which are a set of guidelines for developing accessible web content. The method recommends reading the practices themselves, instead of another document introducing them. This is because they are practical in nature, and an introductory document was deemed to be of little help.

- **Recommended document:** https://www.w3.org/WAI/ARIA/apg/practices/

B.2 Phase B: Make a Test Plan

This phase contains guidelines for how to construct a test plan for the automated accessibility component tests that are to be written. A test plan often consists of identifying the objectives, scope, resources, and schedule of the testing process that shall be undertaken. While a truly comprehensive test plan can grow to a considerable size, this method has reduced this phase into four sub-phases that it feels are essential to accessibility testing.

Sub-phase B1: Define Testing Process Objectives

When this sub-phase is completed, the user should have defined the objective of the testing process. This objective consists of specific goals and outcomes. Such goals and outcomes can vary widely, but a concrete example is wanting to make the web application more accessible through automated testing. The work in this sub-phase should be documented as it is needed in later phases of the method.

Sub-phase B2: Define the Scope of the Testing Process

When this sub-phase is completed, the user should have defined the scope of the testing process. The scope determines the testing boundaries, what the test coverage of each component should be when the testing process is completed, and what functionality needs to be tested. The scope definition will be affected by the complexity of the web application being tested, the resources available to the testers, such as time and personnel, the security demands of the web application, and many other possible factors. The work in this sub-phase should be documented as it is needed in later phases of the method.

Sub-phase B3: Identify Testing Process Resources

Identify the skills, tools, and infrastructure needed in order to carry out the testing process. This includes but is not limited to:

- *Personnel:* Identify testing team members, test leads, and test managers.
- *Testing Tools and Technologies:* In the case of implementing automated component testing for finding accessibility issues, this includes among other things: deciding on what software framework to use to write and execute the automated tests, what imported accessibility testing technology to employ, and what environment to use for displaying and documenting the test results.
- *Company/Organization Web Application Usage Data:* Data on how users interact with the web application that is going to be tested that is available to the company or organization that owns the application.

Sub-phase B4: Create Testing Process Schedule

When this sub-phase is completed, the user should have created a schedule for the testing process. This schedule should outline the testing activities, how these activities should be ordered, and how important each activity is. These activities should be placed into a timeline that serves as the major component of the schedule. The schedule should also contain documentation on how the resources available to the testing team should be allocated during this timeline.

B.3 Phase C: Test Execution

Test execution, in the context of this method, refers to all of the following activities: (1) Gathering the required set of test cases, (2) configuring the automated testing environment and tools, (3) writing the automated tests, (4) running the tests, (5) correctly displaying the test results, (6) documenting the test results, (7) evaluating the test coverage. In order to achieve all these activities, the guidelines contained in this phase have been divided into six sub-phases. Unlike the two earlier main phases—Phase A and Phase C—this phase is iterative and is repeated until the test scope defined in Sub-phase B2 has been met.

This phase, and this method in general, relies heavily on the ARIA APG design patterns to function. If the user has worked through Phase A of the method, they should have the required knowledge to start using the design patterns:

- **ARIA APG Design Patterns:** https://www.w3.org/WAI/ARIA/apg/patterns/

If the content of the ARIA APG cannot be understood, return to Phase A, and consult the recommended documents.

C1: Select Component and Pattern Match

Prioritizing using the testing process schedule produced in Sub-phase B4, select one component of the web application to start this iteration of Phase C. Analyze this component and attempt to match it to one or more design patterns contained in the ARIA APG design patterns. Some components will match two or more patterns, while others are a partial match only to one pattern. If a component cannot be considered a partial match to any pattern, the ARIA APG cannot be used to produce test cases for that component. When the component has been matched, document this matching in a way selected by the user, and then proceed to C1.

C2: Identify Test Case Candidates

Work through all the design patterns that the component was deemed to match and gather all accessibility requirements listed therein. Using a filtering method left up to the method user, decide on which accessibility requirements the user has the technological proficiency and resources to convert into automated tests. After the accessibility requirements have been filtered, convert the remaining requirements into test cases. These test cases can be formal or informal, or written according to some other standard chosen by the method user.

C3: Control Test Case Candidates Using the ARIA in HTML Specification

Using ARIA accessibility requirements to construct tests can be unsafe unless the tester controls that the test cases do not go against the *ARIA in HTML* specification:

- **Link to ARIA in HTML:** https://www.w3.org/TR/html-aria/

In this sub-phase, all test cases gathered in Sub-phase C1 should be controlled using the above specification. After this has been done enough times, the specification might not need to be consulted anymore as the test team has gained a significant enough understanding of the dangers of using ARIA in faulty ways.

C1: Set up Imported Automated Accessibility Testing

Using the technologies, environments, and tools chosen in Sub-phase B3, set up the imported automated testing of the selected component. When and how these tests are run, as well as how to handle the test results is left up to the method user. These imported tests form one of the two categories that the automated accessibility tests for the selected component are divided into. How the imported tests should be configured, or whether they should be adapted or modified is also left up to the method user.

C1: Write Automated Accessibility Tests

Using the test cases that were gathered through Sub-phases C2 to C3, write automated accessibility tests for the selected component. The technologies and environments that are used to write these tests should have been chosen in Sub-phase B3. When every gathered test case for the selected component has been converted into an automated accessibility test, this sub-phase is completed.

C1: Evaluate Component Test Coverage

When all the test cases have been converted into automated component tests, it is time to use the documentation of the scope definition produced in Sub-phase B2 to evaluate the component's test coverage. After the tests have been analyzed using the scope definition, the following question is answered:

- **Is the component's test coverage satisfactory? YES/NO**
  - *YES:* Current iteration of Phase C is complete. Start a new Phase C iteration by going to Sub-phase C1 and selecting a new component for testing.
  - *NO:* If the test coverage is deemed insufficient, the iteration is not done, and more test cases need to be produced. Return to Sub-phase C2 of the current iteration and attempt to find more candidates for test cases. If this is not possible, the decision is left up to the method user to either go back to Sub-phase B2 and redefine the scope of the testing process, or proceed in some other fashion, such as simply moving on to a new component.

Appendix C: Improved AAT-WAC Method

This appendix presents the improved AAT-WAC method that was the result of the *Improved Design* research phase. A presentation of that research phase can be viewed in Section 3.3.4, and an overview of the improved AAT-WAC method that outlines the changes made from the preliminary method can be viewed in Chapter 8.

Introduction

This method should be used if you wish to implement automated accessibility testing of a web application that uses a component-based architecture. This method satisfies a specific need, and it is important that method users understand what the method does and does not do. This introduction aims to concisely present the most important concepts the method user needs to understand:

- **Automated Testing:** This method is only concerned with *automated testing*. There are plenty of available methodologies, process models, and strategies available to implement full-scale accessibility testing of a whole web application. These solutions always require a large amount of *manual testing* in combination with automated testing. Automated accessibility testing is harder to understand, and there are fewer resources to help beginners get started. This method is a tool for precisely that.
- **Self-written Automated Tests versus Imported Automated Tests**: Automated tests can either be self-written, or imported, in which case they have been written by someone else. Many developers believe that imported automated tests are the only available option for automated accessibility testing. This method is meant to show that this is not the case, it is both *feasible* and *useful* to write your own automated accessibility tests. This method is unique in providing support for implementing both categories of automated accessibility tests.
- **Component-based Architecture:** This method is *only* for web applications that use the popular component-based architecture. Examples of such web applications are those built using the popular Angular, React, and Vue software frameworks, among others. This method is *not usable* if the web application under question is not built using a component-based architecture.
- **The Accessible Rich Internet Applications (ARIA) Authoring Practices Guide (APG):** The ARIA APG is an online guide—consisting of many different resources—that is meant to help developers create more accessible web content. The ARIA APG is an *absolutely integral* part of this method, as it is the only source used to produce accessibility requirements from which the automated tests are written. No matter how the user chooses to follow the method, at some point every method user needs to make sure they understand the ARIA APG, or the method will not be usable.

If the concepts listed above have been understood, the user can proceed to the documentation of the phases of the method, starting with Phase A.

Phase A: Establish Required Theoretical Knowledge

End Goal of Phase: Being able to understand the ARIA APG design patterns:

- https://www.w3.org/WAI/ARIA/apg/patterns/

In this phase the method user makes sure that they have the prerequisite knowledge of web content accessibility needed in order to carry out the subsequent phases of the method. This phase consists of five sub-phases, each recommending one document pertaining to web content accessibility. The order and content of these sub-phases are recommendations, and if the user already understands the concepts connected to one sub-phase they can move on to the next. This phase should only be performed once before continuing on with the subsequent phases.

The user is also allowed to define their own major areas of research, decide on the order of study, as well as how to organize the whole process. The important point is that after this phase, every user participating in the method understands the ARIA APG design patterns and how to use them.

First Major Area of Study: Introductory Document for Web Content Accessibility

After A1, the user should have a basic understanding of web content accessibility. There is a myriad of documents aimed at accomplishing this task, but the method recommends the *Introduction to Web Accessibility*, by W3C, as it is authored by a trusted source and conforms perfectly to the purpose of the sub-phase.

- **Recommended document**: https://www.w3.org/WAI/fundamentals/accessibility-intro/

Second Major Area of Study: Introductory Document for WCAG 2.1

After A2, the user should have a basic understanding of WCAG 2.1, the completely dominant web content accessibility standard in use today. The method strongly recommends the official *WCAG 2 Overview,* by W3C, as it is accessible to people unfamiliar with the subject as well as being written by the authors of WCAG 2.1.

- **Recommended document:** https://www.w3.org/WAI/standards-guidelines/wcag/

This Major Area of Study: Introductory Document for WAI-ARIA

After A3, the user should have a basic understanding of the WAI-ARIA suite of web content accessibility standards. The method recommends the *WAI-ARIA Overview*, by W3C, as it is written by the authors of ARIA and provides a clear introductory overview to the subject.

- **Recommended document:** https://www.w3.org/WAI/standards-guidelines/aria/

Fourth Major Area of Study: Introductory Document for the ARIA APG

After A4, the user should have a basic understanding of the ARIA APG, a diverse guide aimed at helping web developers create more accessible interactive web content. The method recommends the *Introduction to ARIA APG*, by W3C, as it is written by the authors of the ARIA APG.

- **Recommended document:** https://www.w3.org/WAI/ARIA/apg/about/introduction/

Fifth Major Area of Study: Introductory Document for the ARIA APG Practices

After A5, the user should have a basic understanding of the official ARIA APG practices, which are a set of guidelines for developing accessible web content. The method recommends reading the practices themselves, instead of another document introducing them. This is because they are practical in nature, and an introductory document was deemed to be of little help.

- **Recommended document:** https://www.w3.org/WAI/ARIA/apg/practices/

Phase B: Make a Test Plan

Output After Completing this Phase:

- **Document #1:** Text
- **Document #2:** Text
- **Document #3:** Text
- **Document #4:** Text
- **Document #5:** Text

This phase contains guidelines for how to construct a test plan for the automated accessibility component tests that are to be written. A test plan often consists of identifying the objectives, scope, resources, and schedule of the testing process that shall be undertaken. While a truly comprehensive test plan can grow to a considerable size, this method has reduced this phase into activities that it feels are essential to accessibility testing.

If the method users wish to construct their own test plan, or organize the planning process in another way, they are free to do so. However, the five documents listed above need to be produced in order to properly follow Phase C of the method. The structure and precise contents of these documents are left up to the method user.

Activity #1: Define Testing Process Objectives

When this activity is completed, the user should have defined the objective of the testing process. This objective consists of specific goals and outcomes. Such goals and outcomes can vary widely, but a concrete example is wanting to make the web application more accessible through automated testing. The work in this sub-phase should be documented as it is needed in later phases of the method.

Activity #2: Define the Scope of the Testing Process

When this activity is completed, the user should have defined the scope of the testing process. The scope determines the testing boundaries, what the test coverage of each component should be when the testing process is completed, and what functionality needs to be tested. The scope definition will be affected by the complexity of the web application being tested, the resources available to the testers, such as time and personnel, the security demands of the web application, and many other possible factors. The work in this activity should be documented as it is needed in later phases of the method.

Activity #3: Identify Testing Process Resources

Identify the skills, tools, and infrastructure needed in order to carry out the testing process. This includes but is not limited to:

- *Personnel:* Identify testing team members, test leads, and test managers.
- *Testing Tools and Technologies:* In the case of implementing automated component testing for finding accessibility issues, this includes among other things: deciding on what software framework to use to write and execute the automated tests, what imported accessibility testing technology to employ, and what environment to use for displaying and documenting the test results.

- *Company/Organization Web Application Usage Data:* Data on how users interact with the web application that is going to be tested that is available to the company or organization that owns the application.

If this activity has been properly carried out, the method user will not need to make any sort of decision on which technologies or strategies to use in the testing stage. The software framework used to write the automated tests should already be decided on, as should the technology used to organize and save the test results, and the strategy when writing the test case proposals, and so on.

Activity #5: Component Inventory and Analysis

During this activity, all the components of the web application that is being tested should be listed and analyzed. Start by listing every single component used, and then attempt to use the components complexity, relation to each other, and usage in the application to decide which components are suitable for automated accessibility tests. After this activity has been concluded, the testing team should have a clear understanding of what components that they will be writing automated tests for in Phase C.

Activity #5: Create Testing Process Schedule

When this activity is completed, the user should have created a schedule for the testing process. This schedule should outline the testing activities, how these activities should be ordered, and how important each activity is. These activities should be placed into a timeline that serves as the major component of the schedule. The schedule should also contain documentation on how the resources available to the testing team should be allocated during this timeline.

Phase C: Write, Run, Monitor, and Document Tests for Selected Component

Phase Input (Mandatory):

- **Input #1:** A list of all documents that should be tested. The method recommends producing this list by performing *Activity #4* of Phase B
- **Input #2:** A document defining the required test coverage for each listed component. The method recommends producing this document by performing *Activity #2* of Phase B
- **Input #3:** Documentation of all the resources that are needed in the testing process. The method recommends producing this documentation by performing *Activity #3* of Phase B
- **Input #4:** The ARIA APG design patterns (link included below)

This phase is iterative. It is repeated until every component that was selected for testing in Activity #4 of Phase B has been tested. During this phase, the imported automated tests are configured, the self-written automated tests are implemented, and the results of the tests are monitored, documented, and saved for future analysis. The phase consists of seven activities, and each activity is carried out for every iteration of the phase.

This phase, and this method in general, relies heavily on the ARIA APG design patterns to function. If the user has worked through Phase A of the method, they should have the required knowledge to start using the design patterns:

- **ARIA APG Design Patterns:** https://www.w3.org/WAI/ARIA/apg/patterns/

If the content of the ARIA APG cannot be understood, return to Phase A, and consult the recommended documents.

Activity #1: Select Component and Pattern Match

Select one component of the web application from *Input #1* to start this iteration of Phase C. Analyze this component and attempt to match it to one or more design patterns contained in the ARIA APG design patterns. Some components will match two or more patterns, while others are a partial match only to one pattern. If a component cannot be considered a partial match to any pattern, the ARIA APG cannot be used to produce test cases for that component. When the component has been matched, document this matching in a way selected by the method user, and then proceed to *Activity #2*.

Activity #2: Identify Test Case Candidates

Work through all the design patterns that the component was deemed to match and gather all accessibility requirements listed therein. Using a filtering method left up to the method user, decide on which accessibility requirements the user has the technological proficiency and resources to convert into automated tests. After the accessibility requirements have been filtered, convert the remaining requirements into test cases. These test cases can be formal or informal, or written according to some other standard chosen by the method user.

Activity #3: Control Test Case Candidates Using the ARIA in HTML Specification

Using ARIA accessibility requirements to construct tests can be unsafe unless the tester controls that the test cases do not go against the *ARIA in HTML* specification:

- **Link to ARIA in HTML:** https://www.w3.org/TR/html-aria/

In this activity, all test cases gathered in *Activity #1* should be controlled using the above specification. After this has been done enough times, the specification might not need to be consulted anymore as the test team has gained a significant enough understanding of the dangers of using ARIA in faulty ways.

Activity #4: Set up Imported Automated Accessibility Testing

Using *Input #3* set up the imported automated testing of the selected component. When and how these tests are run, as well as how to handle the test results is left up to the method user. These imported tests form one of the two categories that the automated accessibility tests for the selected component are divided into. How the imported tests should be configured, or whether they should be adapted or modified is also left up to the method user.

Activity #5: Write Automated Accessibility Tests

Using the test cases that were gathered through activities #2 and #3, write automated accessibility tests for the selected component. The technologies and environments that are used to write these tests can be found in *Input #3*. When every gathered test case for the selected component has been converted into an automated accessibility test, this activity is completed.

Activity #6: Evaluate Component Test Coverage

When all the test cases have been converted into automated component tests, it is time to use the documentation of *Input #2* to evaluate the component's test coverage. After the tests have been analyzed using the scope definition, the following question is answered:

- **Is the component's test coverage satisfactory? YES/NO**
  - *YES:* Continue to *Activity #7* of the current iteration
  - *NO:* If the test coverage is deemed insufficient, the iteration is not done, and more test cases need to be produced. Return to *Activity #2* of the current iteration and attempt to find more candidates for test cases. If this is not possible, the decision is left up to the method user to either go back to *Activity #2* of Phase B to redefine the scope of the testing process, or proceed in some other fashion, such as simply moving on to a new component.

Activity #7: Evaluating the Test Results

During this activity, all the test results of the component, as well as how they are handled, are evaluated. The method user should check that the test results are displayed in the correct fashion, that they have been documented in the way that *Input #3* describes, and that they are saved for future use in the right way. After the test results have been evaluated, answer the following question:

- **Are the component's test results handled in the correct way? YES/NO**
  - *YES:* Iteration is complete. Initiate a new iteration starting *Activity #1*. If this was the last listed component, the whole method has been implemented
  - *NO:* Return to *Activity #3* of Phase B in order to find a new solution for how to handle the test results. This action will result in a new version of *Input #3* for the current iteration of Phase C.

TRITA – EECS-EX-2024:15