A Knowledge-Based Framework for Improving Accessibility of Web Sites

Jens Pelzetter(✉)

FB3 Informatik, Universität Bremen, Bremen, Germany
jens.pelzetter@uni-bremen.de

Abstract. Many sites in the World Wide Web are, unfortunately, not accessible or usable for people with impairments despite several existing guidelines. This paper describes an approach for improving the accessibility of web sites using ontologies as the foundation for several tools. The approach is investigated as a PhD thesis as part of other research that uses ontologies to provide disabled or elderly people with assistance for several everyday tasks.

Keywords: Accessibility · World Wide Web · Ontologies

1 Introduction

For many people the World Wide Web has become their primary source of information. The technologies developed for the Web are used in many other areas. Many organizations have intranets to share informations with their employees. Many applications are provided as Web Applications. These only require a web browser as client.

Developing good web sites or web applications is a quite challenging task. Applications must be usable with a variety of devices (smartphones, tablets and PCs). On top of this, the web sites or web applications should be *accessible*. Accessibility of web sites or web applications is a wide field. Many people only think about blind people when they hear the term *accessibility* in association with web sites or web applications. Accessibility for the web includes much more. Impairments that can affect the way how people may interact with web sites and web applications include problems with all senses and also motoric impairments.

However there are not many tools that support developers to create accessible web sites. Many sites currently available on the web are not accessible and will not become accessible very soon. In this paper, an approach is proposed to use ontologies as the foundation of several tools to improve web accessibility.

The terms *web site* and *web application* will be used interchangeably in this paper since the difference has become very small in the last years. The term *web site* will be used for a collection of web pages, the term *web page* refers to a single page inside a web site.

E. Blomqvist et al. (Eds.): ESWC 2017, Part II, LNCS 10250, pp. 226–235, 2017.
DOI: 10.1007/978-3-319-58451-5_17

2 State of the Art

There are several guidelines for accessible web sites. The most current and widely adopted standard are the *Web Content Accessibility Guidelines 2.0* (WCAG 2.0) [5] created by the W3 Consortium. Most of the other standards for accessible web sites are based on it, for instance the German BITV [22].

Nevertheless many web sites either ignore these standards completely or do not implement them correctly. One reason might be that there are no good, simple to use tools to test a web site for accessibility. All test procedures for accessibility we are aware of either only check some of the very basic requirements or require manual testing.

For the WCAG 2.0, there are several studies which examine, how reliable the results of these tools are [1,3,4]. These studies discovered that the reliability of the results depends on the experience of the tester. Unexperienced testers either find very few or too many problems.

Garrido et al. propose the use of *client side refactorings* to make web pages accessible [8]. Such refactorings use small pieces of JavaScript altering a web pages to make it more accessible. However, the user has to know which refactorings must be applied to a certain web page to make the site accessible for him or her.

The Cloud4all project [18] was a broad attempt funded by the European Union to improve the accessibility of IT technology. In this project an infrastructure was developed that should allow users to store a profile with their preferred settings in the Cloud. Applications can retrieve that profile from the Cloud. The appropriate settings from these profiles are applied to a specific environment using so called *matchmakers*[24]. The primary focus of the project was on traditional, native applications and the usage of native accessibility functions of the operating systems and desktop environments [9]. Besides implementations for Windows and Gnome [2], a proof of concept implementation for web sites was created [19].

Other researchers have tried to use semantic web technologies to enhance the accessibility of web pages and applications. Kouroupetroglou et al. [15] developed a framework which uses annotations in the pages. These annotations can be used by a user agent to provide a better user experience for users of assistive technology. Their research was focused on visually impaired people.

A similar approach is described by Semaan et al. [21], but with a stronger focus on describing the relationship between the several blocks of information on a web page. Their approach was to transform a web page into an RDF document which could then be viewed in the special browser. This special browser uses the additional informations about the document structure to enhance the user experience for users with special needs.

In 2014 the W3C has published ARIA 1.0 [7] (**A**ccessible **R**ich **I**nternet **A**pplications). ARIA uses an approach similar to the approaches described in [15,21]. A web page is annotated with special attributes. The informations provided by these annotations are used by the browser to provide assistive technology with additional information about the elements of a web page.

In other areas, such as mobility assistance, formal modeling approaches have been used with some success [16,20]. Our research group at Universität Bremen and DFKI Bremen has already created a large ontology in OWL-DL [14] describing illnesses, impairments and how they effect abilities such as sight. This ontology describes what of mobility assistance a person with certain impairments requires.

3 Problem Statement and Contributions

Despite the various approaches described in Sect. 2 and the availability of standards like the WCAG 2.0 [5] many web sites are still not accessible. There is also a lack of good tools for checking the accessibility of web sites. All tools which do an automatic check of the accessibility of a web page only check a limited range of requirements. An example for such an tool is the WAVE tool[1]. Test procedures which check a larger range of accessibility requirements require extensive manual work. An example is the BITV-Test[2].

But even if we get good tools for evaluating the accessibility of web pages there will still be many non accessible pages. Therefore it is also necessary to provide tools for disabled users to provide them with a better user experience when accessing non accessible web pages.

The primary research question of this work is whether ontologies can be used to model the knowledge about accessible web pages in a formal way *and* whether they can be used to automatically infer knowledge about accessible web page. One of the possible use cases is a tool which analyses a web page and then uses the knowledge from an ontology about accessibility for web pages to automatically apply refactorings as described in [8].

A common accessibility problem on web pages is an insufficient contrast between the background color and the color of the text. In many cases, this problem could easily be fixed by a client side refactoring. The WCAG 2.0 [5] contains two Success Criteria for contrast. Success Criterion 1.4.3 specifies the minimal requirement for contrast, Success Criterion 1.4.6 specifies an enhanced requirement. Often the only thing necessary to match the requirements and make a web page better readable for people with sight problems is to make the darker color a bit darker and the lighter color a bit lighter.

Another common accessibility problem is that many web sites do not specify their primary language (WCAG 2.0 Success Criterion 3.1.1). This information is needed by screen readers to choose the right pronunciation. A screen reader is a program, which presents the informations normally perceived visually either as speech or as tactile output using a Braille output device. In HTML it is also possible to specify the language of parts of a document by using an attribute (WCAG 2.0 Success Criterion 3.1.2). This information is also useful for screen readers. If the language is provided for a word or part of a web page, which is

[1] WAVE Tool: http://wave.webaim.org/.
[2] BITV-Test http://www.bitvtest.eu/.

not in the primary language of the web page, the screen readers can pronounce this word or part correctly.

A third example is the provision of alternative texts for images. These texts are often missing or applied incorrectly. The alternative text for an image is provided by the `alt` attribute of the `img` Element. For decorative images it is necessary to specify an *empty* `alt` attribute. Otherwise the screen readers use the filename as an alternative text. More details about alternative texts for images on web pages can be found in the description of the HTML element in the HTML5 standard [12] and in the description of technique H67 in [6].

There are several challenges along the way to accessible web pages. The first one is to translate the Web Content Accessibility Guidelines 2.0, a semi-formal specification written in natural language, to a formal description (an ontology).

The WCAG 2.0 consists of several documents. The primary one is written in a technology neutral manner. This document has not been updated since 2008. It describes several Success Criteria for accessible web sites, grouped into guidelines and principles. How these Success Criteria can be implemented is described in separate documents. The document describing the possible techniques [6] to implement the Success Criteria is regularly updated (last updated in October 2016) to include new technologies and other developments.

The W3 provides a tool [23] to connect the Success Criteria and the techniques. The challenging part for modeling the ontology is the connection between the Success Criteria and the techniques (which also describe test procedures). For some Success Criteria, the applicable techniques depend on certain conditions (called situations). For others this is not the case. Some techniques are meta techniques, which can be implemented by several other techniques. For some Success Criteria, two technologies are combined into a new one in the descriptions provided by the tool.

A second challenge lies in the nature of web sites. Web sites are written in HTML. The HTML standard has seen many different versions in the last 25 years, the current one is HTML 5 [12]. For several reasons, some web sites have been written in a very sloppy manner. Even today many web sites are not completely valid when checked with a validation tool for HTML. Users usually don't notice this because the user agents (browsers) have been become very good in making sense of defect HTML documents. Thus there are many invalid HTML documents out there. Analyzing them using formal methods should be quite challenging.

4 Research Methodology and Approach

To achieve the goals described above, the first step was to identify the relevant standards for accessible web sites including literature research about current approaches for test tools and methods for making inaccessible web sites accessible. To learn how users with impairments use the web sites, several afflicted users have been interviewed.

The next step is to create an ontology describing the relevant standards and methods to represent the properties of the web site under test. This also involves

combining the ontology describing the WCAG 2.0 with the existing ontology of impairments and abilities.

Using the ontologies, some tools will be created as "proof of concept" and tested with users and web developers. The tools for web developers will use the ontologies to guide web developers through an accessibility test of a web site. The accessibility test itself will be semi-automatic. Some requirements can be checked without human interaction. Some requirements can not be check automatically, for example if the alternative text for an image is sufficient. For these requirements the test tool will guide the tester through the test procedure using structured questions.

The tools for users will include a browser plugin using the ontologies and automatic test procedures to automatically apply refactorings to web sites, depending on the abilities of the user and the properties of a web site. An example of an accessibility problem that can thus be fixed is insufficient contrast between foreground and background colors (cf. Sect. 3).

There are several different groups of impairments that affect how users can or can not use web sites. The most well-known are of course blindness or the inability to use standard input devices. However, there are many other forms and degrees of impairment that are relevant for accessible web sites, for example color blindness or a reduced field of vision. Moreover, people with cognitive impairments (caused for example by a head injury) might have problems using web sites. A test setup will be developed to evaluate how helpful the tools developed are for users with different kind of impairments. These relationships between the impairments of person how they effect the abilities of person and how they can compensated will be modelled in several interlinked ontologies.

There are several standards, which can be used by web sites to provide a formal description about their content. These include Microformats [17], Microdata [13], and RDFa [11]. If a web site provides such information, it should be possible to use this information to provide some kind of navigation assistance for the web site. This could be very useful for users with cognitive impairments who have difficulty finding information in a complex web site.

The ontologies developed as the foundation of the tools for web accessibility will be used to verify and test the pattern-based ontology tools developed by our research group. One focus of the ontology tools is to support ontology designers with safe maintenance support for ontologies.

5 Preliminary Results and Current Work

It has been more difficult than expected to translate the WCAG 2.0 into an ontology. Several versions of an ontology describing the WCAG 2.0 had to be developed to test different modeling approaches. Some relations between the concepts of the WCAG 2.0 and their instances could not be expressed with OWL 2 DL alone. To express these relations some SWRL rules [10] are used. Current work is focused on the ontology representing the WCAG 2.0 and the supporting documents using OWL 2 as well as a first simple tool.

The ontology describing the WCAG 2.0 has been divided into three parts representing the concepts and relations in these documents. They do not contain any of the Success Criteria, Techniques etc. Due to the amount of data – the *Techniques for WCAG 2.0* document for example contains several hundred techniques – a web scrapping tool has been developed that extracts the data from the web site of the W3C using the jsoup library[3]. The extracted data is used to create the OWL objects representing the Success Criteria etc. using the OWL API[4]. Fortunately this was quite easy thanks to the well-structured HTML format of the relevant documents. In addition to the six ontology documents for the WCAG 2.0, an additional ontology has been created containing some SWRL rules used to infer whether a web page is satisfying a conformance level.

The WCAG 2.0 defines several conformance levels for the accessibility of web pages. To achieve a conformance level, a web page has to meet several success criteria. This relation is represented by the `requiresSuccessCriterion` object property and the inverse object property `requiredByConformanceLevel`.

For each success criterion, several test cases are provided in the supporting documents, grouped into two major categories: *Techniques* describe an approach for meeting a success criterion in a specific situation; *Failures* describe the conditions under which a success criterion cannot be met by a web page.

The ontology contains classes for situations, techniques and failures, and the web pages under evaluation. Success criteria and situations are related by the object properties/inverses `hasSituation`/`isSituationForSuccessCriterion`, a success criterion and a failure by `hasFailure`/`isFailureForSuccessCriterion`, and `hasSufficientTechnique`/`isSufficientTechniqueForSituation` relates, which techniques are sufficient for a specific situation.

To achieve a particular conformance level, a web page must meet all its success criteria. A success criterion is met by a web page, if the web page does not contain any of the failures and meets the requirements for all situations of the success criterion. To meet the requirements of a situation, the web page has to implement at least one of the sufficient techniques for the situation successfully. The requirements for a situation to be met, if the situation is not applicable for a webpage, are also considered. Whether a web page meets a success criterion, contains a failure, etc., is represented by several object properties in the ontology. The foundation are the object properties `successfullyImplementsTechnique`, `containsFailure` and `notContainsFailure`.

To infer that a web page does not match a success criterion, if one of the failures is present on the web page, the following rule is used:

```
WebPage(?p), SuccessCriterion(?sc), Failure(?f),
isFailureForSuccessCriterion(?f, ?sc), containsFailure(?p, ?f)
-> notMeetsSuccessCriterion(?p, ?sc)
```

Whether a web page matches the requirements for a specific situation is inferred using two rules:

[3] jsoup library: https://jsoup.org.
[4] OWLAPI: https://github.com/owlcs/owlapi.

```
WebPage(?p), Situation(?s), notAppliesToSituation(?p, ?s)
-> matchesRequirementsForSituation(?p, ?s)

WebPage(?p), Situation(?s), Technique(?t),
isSufficientTechniqueForSituation(?t, ?s),
successfullyImplementsTechnique(?p, ?t)
-> matchesRequirementsForSituation(?p, ?s)
```

The first rule simply states that a web page matches the requirements for a situation, if the situation is not applicable for the web page, even if the web page does not implement any of the sufficient techniques for the situation. If the web page implements at least one of the sufficient techniques for a situation successfully, the web page matches requirements for that situation.

Now we need to define an rule to infer that a web page meets a success criterion, if the web page matches the requirements for situations of the success criterion and does not contain any of the failures for the success criterion. But due to the Open World Assumption of OWL 2, we cannot simply assert this. Unless stated otherwise, there may be failures or situations that are not described in the ontology. Therefore it is necessary to explicitly assert that there are no other situations or techniques. For this purpose, two classes are added for each success criterion, which contain only the situations and failures for this success criterion.

An example for failures of Success Criterion 1.1.1 in OWL functional syntax is

```
Declaration(Class(wcag20tf:FailureForSuccessCriterion-1-1-1))
EquivalentClasses(wcag20tf:FailureForSuccessCriterion-1-1-1
    ObjectOneOf(<wcag20-techniques#F13> <wcag20-techniques#F20>
                <wcag20-techniques#F72>))
SubClassOf(wcag20tf:FailureForSuccessCriterion-1-1-1
           wcag20-techniques:Failure)
SubClassOf(wcag20tf:FailureForSuccessCriterion-1-1-1
           ObjectHasValue(wcag20tf:isFailureForSuccessCriterion
               <wcag20#successCriterion-1-1-1>))
```

An earlier version of the ontology used cardinality assertions to achieve the same effect, but these made reasoning extremely slow. Using these classes, the following rule states that a web page meets a specific success criterion:

```
WebPage(?p), (matchesRequirementsForSituation
    min 6 SituationForSuccessCriterion-1-1-1)(?p),
(containsFailure max 0 FailureForSuccessCriterion-1-1-1)(?p)
-> meetsSuccessCriterion(?p, successCriterion-1-1-1)
```

This rule uses a class expression with a cardinality requirement. It requires that the web page `?p` have a associated to at least six instances of the class `SituationForSucessCriterion-1-1-1` by the object property `matchesRequirementsForSituation`.

The same approach is used to infer that a web page achieves a conformance level. Which success criteria are required by a conformance level is asserted using a class to infer whether a web page achieves a particular conformance level:

```
WebPage(?p), (meetsSuccessCriterion min 25
    SuccessCriterionForConformanceLevel-A)(?p)
-> compliesToConformanceLevel(?p, ConformanceLevel-A)
```

The next step will be to develop ontologies for the test procedures for the techniques, the refactorings and the requirements of users with impairments.

6 Evaluation Plan

When the first tools are ready to use the tools will be tested with various users. The first group of testers will be users with impairments, who will test the tools that automatically apply refactorings to a web site. The sites used in these tests will be evaluated with the tools developed before for testing web sites for accessibility problems to find out, which accessibility problems they might have. The users will have to execute several tasks, such as finding a specific information, on each site. For these tests the users will be split into two groups. The first group will execute the tasks without the support of the tools developed. The second group will use the tools developed for automatically applying client side refactorings and execute the tasks with the support of these tools. The results of the two groups will be compared to find out whether the tools improve the usability of the web sites for these users. To ensure that both groups are balanced regarding their abilities we will do interviews with each participant before they execute the tasks.

The second group of testers will consist of several web developers, who will use the tools in their daily work. This group will include web developers in larger companies with a solid background in programming and web design, but also developers and designers from small companies, who only occasionally develop web sites and have little programming background. Before the testers will start to use the tools, they will be asked to fill in a questionnaire with some questions estimating their experience in the field of accessibility. After about four to eight weeks, the testers will interviewed about their experience with the tools. The web sites that have been created with the help of the tools developed will be analyzed to investigate whether they have less accessibility problems than average sites.

7 Conclusions

The primary goal of the PhD thesis outlined in this paper is to develop a solid foundation for tools to improve the accessibility of web applications and web sites. This will allow developers to provide tools and better web applications and web sites. Users, especially those who rely on assistive technologies, will

get web sites and web applications that are hopefully more accessible and thus better usable.

Accessibility for web sites and web applications is a complex domain. The lessons learned while developing the ontologies describing the knowledge about this domain will be helpful for other developers, who create ontologies in other similarly complex domains.

Acknowledgments. I would like to thank my supervisor Bernd Krieg-Brückner and my mentor Vojtěch Svátek for their valuable feedback.

References

1. Alonso, F., Fuertes, J.L., González, Á.L., Martínez, L.: On the testability of WCAG 2.0 for beginners. In: Proceedings of the 2010 International Cross Disciplinary Conference on Web Accessibility (W4A). p. 9. ACM (2010)
2. Antúnez, J.H., Clark, C., Markus, K.: The GPII on desktops in PCs OSs: Windows and GNOME. In: Stephanidis, C., Antona, M. (eds.) UAHCI 2014. LNCS, vol. 8516, pp. 390–400. Springer, Cham (2014). doi:10.1007/978-3-319-07509-9_37
3. Brajnik, G., Yesilada, Y., Harper, S.: Testability and validity of WCAG 2.0: the expertise effect. In: Proceedings of the 12th International ACM SIGACCESS Conference on Computers and Accessibility, pp. 43–50. ACM (2010)
4. Brajnik, G., Yesilada, Y., Harper, S.: Is accessibility conformance an elusive property? A study of validity and reliability of WCAG 2.0. ACM Trans. Accessible Comput. (TACCESS) **4**(2), 8 (2012)
5. Caldwell, B., Cooper, M., Reid, L.G., Vanderheiden, G.: Web Content Accessibility Guidelines (WCAG) 2.0, December 2008. http://www.w3.org/TR/WCAG20/
6. Cooper, M., Kirkpatrick, A., Connor, J.O.: Techniques for WCAG 2.0 (2016). https://www.w3.org/TR/2016/NOTE-WCAG20-TECHS-20161007/
7. Craig, J., Cooper, M., Pappas, L., Schwerdtfeger, R., Seeman, L.: Accessible rich internet applications (WAI-ARIA) 1.0. Technical report, W3C, March 2014. https://www.w3.org/TR/wai-aria/
8. Garrido, A., Firmenich, S., Rossi, G., Grigera, J.: Personalized web accessibility using client-side refactoring. IEEE Internet Comput. **17**(4), 58–66 (2013)
9. Gemou, M., Bekiaris, E., Vanderheiden, G.: Auto-configuration through cloud: initial case studies for universal and personalised access for all. In: 2013 IST-Africa Conference and Exhibition (IST-Africa), pp. 1–8. IEEE (2013)
10. Harrocks, I., Patel-Schneider, P.F., Boley, H., Tabet, S., Grosof, B., Dean, M.: SWRL: A Semantic Web Rule Language Combining OWL and RuleML (2004). https://www.w3.org/Submission/SWRL/
11. Hermann, I., Adida, B., Sporny, M., Birbeck, M.: RDFa 1.1 Primer - Third edition. Rich Structured Data Markup for Web Documents (2015)
12. Hickson, I.: HTML5. A vocabulary and associated APIs for HTML and XHTML, May 2011. http://www.w3.org/TR/html5/
13. Hickson, I.: HTML microdata (2013). https://www.w3.org/TR/microdata/
14. Hitzler, P., Krötzsch, M., Parsia, B., Patel-Schneider, P.F., Rudolph, S.: OWL 2 Web Ontology Language Primer, December 2012. https://www.w3.org/TR/owl2-primer/

15. Kouroupetroglou, C., Salampasis, M., Manitsaris, A.: A semantic-web based framework for developing applications to improve accessibility in the WWW. In: Proceedings of the 2006 International Cross-disciplinary Workshop on Web Accessibility (W4A): Building the Mobile Web: Rediscovering Accessibility? W4A 2006, NY, USA, pp. 98–108 (2006). doi:10.1145/1133219.1133238
16. Krieg-Brückner, B.: Generic ontology design patterns: qualitatively graded configuration. In: Lehner, F., Fteimi, N. (eds.) KSEM 2016. LNCS (LNAI), vol. 9983, pp. 580–595. Springer, Cham (2016). doi:10.1007/978-3-319-47650-6_46
17. Microformats 2. http://microformats.org/wiki/microformats2
18. Ortega-Moral, M., Peinado, I., Vanderheiden, G.C.: Cloud4all: scope, evolution and challenges. In: Stephanidis, C., Antona, M. (eds.) UAHCI 2014. LNCS, vol. 8516, pp. 421–430. Springer, Cham (2014). doi:10.1007/978-3-319-07509-9_40
19. Peinado, I., Ortega-Moral, M.: Making web pages and applications accessible automatically using browser extensions and apps. In: Stephanidis, C., Antona, M. (eds.) UAHCI 2014. LNCS, vol. 8516, pp. 58–69. Springer, Cham (2014). doi:10.1007/978-3-319-07509-9_6
20. Rink, M., Krieg-Brückner, B.: Wissensbasierte Konfiguration von Mobilitäts-Assistenten. In: VDE e.V. (ed.) Zukunft Lebensräume Kongress 2016 (ZL 2016), pp. 201–206. VDE Verlag, April 2016
21. Semaan, B., Tekli, J., Issa, Y.B., Tekli, G., Chbeir, R.: Toward enhancing web accessibility for blind users through the semantic web. In: 2013 International Conference on Signal-Image Technology Internet-Based Systems, pp. 247–256, December 2013
22. Verordnung zur Schaffung barrierefreier Informationstechnik nach dem Behindertengleichstellungsgesetz, Barrierefreie Informationstechnik-Verordnung - BITV 2.0 (2002)
23. How to Meet WCAG 2.0. A customizable quick reference to Web Content Accessibility Guidelines (WCAG) 2.0 requirements (success criteria) and techniques. https://www.w3.org/WAI/WCAG20/quickref/
24. Zimmermann, G., Strobbe, C., Stiegler, A., Loitsch, C.: Global public inclusive infrastructure (GPII)-personalisierte benutzerschnittstellen/global public inclusive infrastructure (GPII)-towards personal user interfaces. i-com **13**(3), 29–35 (2014)