Comparative Analysis of Online Web Accessibility Evaluation Tools

Cristian Timbi-Sisalima *ctimbi@ups.edu.ec*
Universidad Politécnica Salesiana
Cuenca, Ecuador

Carlos Iván Martín Amor *carlosivan.martin@edu.uah.es*
University of Alcalá
Alcalá de Henares, Spain

Salvador Otón Tortosa *salvador.oton@uah.es*
University of Alcalá
Alcalá de Henares, Spain

José R. Hilera *jose.hilera@uah.es*
University of Alcalá
Alcalá de Henares, Spain

Juan Aguado-Delgado *j.aguado@edu.uah.es*
University of Alcalá
Alcalá de Henares, Spain

Abstract

In many countries, it is mandatory that Web information systems are accessible so that people with disabilities can use them. The developers of web information systems must ensure that their systems are accessible, and for this it can help the use of automatic evaluation tools. This paper presents the results of a comparative analysis of the performance of online accessibility evaluation tools. This analysis can be useful for developers of information systems, as it provides information that can be taken into account when deciding the tool or tools of this type that they will use in their projects. The paper also includes a proposal for classification of different types of tools that evaluate software accessibility, considering two dimensions: its usage and functionality.
Keywords: Accessibility, Testing, Disability, Accessibility evaluation tool.

1. Introduction

Related with information systems, accessibility is the condition to be met by a system to be understandable, usable and practical for all people, including users with disabilities. In most of the more than 150 countries that have signed or ratified the United Nations Convention on Rights of Persons with Disabilities [1,2] there are laws, or commitment to create them, which state that information systems that can be used over the Internet should be accessible [4]. This obligation usually applies to at least the web information systems of public administrations and strategic companies, or those of particular relevance to citizens (energy, communications, transportation, etc.).

It is therefore very important that developers of web information systems take into account that systems developed should be accessible. There are different standards that establish the requirements to be met by a web information system that is accessible. The best known and applied is the standard entitled "Web Content Accessibility Guidelines" (WCAG 2.0), approved in 2008 by the World Wide Web Consortium (W3C) [11], and it became ISO standard in 2012 [5].

To verify compliance with the accessibility requirements of a web information system, a developer can use automatic evaluation tools. Although a manual ratification of the results provided by such tools will always be necessary. And it is also very important to have the collaboration of end users with disabilities to ratify that the final system is fully accessible and have no trouble using its functionality.

There are many types of automatic accessibility evaluation tools [10]. The most commonly used are online tools that offer a web form where the user can enter the URL of the web information system to analyze, and the tool performs an automatic test of potential accessibility problems that present the pages of that system.

It is not easy for developers to choose the most appropriate online assessment tool. Therefore, this paper has conducted an analysis of existing tools, providing results that can be useful for developers when choosing the tool to use in their projects.

The rest of the paper is organized as follows. A proposal of general classification of automatic accessibility evaluation tools is presented in Section 2. The criteria used for comparing and the values assigned for each tool are set out in Section 3. The comparison of tools is presented in Section 4. Finally, Section 5 summarize conclusions about the work done.

2. Classification of automatic evaluation tools for software accessibility testing

There are a number of tools that allow automatic evaluation of some aspects that influence the accessibility of software, especially software for the web. The authors of this paper have conducted an investigation into the different types of existing automatic evaluation tools, analyzing 126 tools and resulting in a proposal for their classification based on two properties: its functionality and its usage mode.

Regarding functionality, we have identified seven main functions. The main one in the evaluation of the accessibility of a page or website. This is precisely the functionality for which the comparative analysis contained in the following sections of this paper is made. Although the tools discussed in this paper provide such functionality, in the columns of Table 1 other six types of features that may be useful for evaluating the accessibility of software or digital information resources are collected. As is the case of the evaluation of accessibility of documents (in pdf format, docx, or other). They are also useful tools that measure the level of readability of the text, as a criterion to say that a digital resource is accessible is that it is easily understood its content by users

Other features offered by some tools is the possibility of checking whether the contrast in images or text is sufficient, or if a moving image includes flicker that can cause epilepsy. Finally, although the vast majority of assessment tools focus on accessibility of web pages, there are a small number of tools for automatic evaluation of the accessibility of other applications, such as native applications for mobile devices like phones and tablets, or desktop applications.

Regarding how to use, they have identified ten different modes. There are tools that offer functionality as an online service with a Web form interface, so that the tool has an associated URL that the user accesses through a web browser. In the main page of the tool exists a form where the user enters the data necessary for the evaluation. These data are received by a web server, and returns the results that can be displayed in the web browser. This is the case of the tools studied in the following sections of this paper.

Although these are the tools discussed in this paper, to have a general idea of other existing tools, in the rows of Table 1 nine other types of evaluation tools based on its mode of usage are collected. Therefore, there are tools that offer functionality in the form of remote web services such as Web API, which should be invoked using the appropriate protocol, in general RESTFul. Other tools can be integrated directly into web browsers, such as extensions or plugins thereof. There are also extensions for editing environments, such as IDE (e.g. Visual Studio or Eclipse) or text editors (e.g. Word).

There are other assessment tools that can be installed as desktop applications, others like app on mobile devices, and even those that have been created as a web application for installation on a local server. There are also tools used from the command line of an operating system, and others that do not respond to the traditional concept of a program, but software library, offering evaluation and functionality through an API for a particular programming language. Finally there are the so-called meta-tools, i.e. tools that really what they do is to reuse the functionality of one or more tools simultaneously.

Table 1. Classification of 126 automatic evaluation tools for accessibility testing (evaluation functionality in columns, use mode in rows).

	Web page Accessi-bility	**Document Accessi-bility**	**Text Reada-bility**	**Color Contrast**	**Epilepsy**	**Mobile App accessi-bility**	**Desktop software accessi-bility**
Web Form Service	32	2	4	5		2	
Web API service	4						
Web browser plugin	23		1	1			
Authoring tool plugin	13	2				1	1
Desktop application	1	1	1	2	1		
Mobile app						1	
Local web server	2						
Command Line	10						
Software Library	11						1
Meta-tool	4						

After analyzing 126 tools, in Table 1 the number of these tools for the types described above is indicated. It can be seen that the most commonly used mode is through a web form interface, and the most common functionality is the evaluation of complete web pages. We found a total of 32 tools of this latter type, as a starting point for the comparative analysis in the following section.

3. Criteria and comparative analysis

The group of more tools and in turn subject matter of this paper, as previously mentioned are the tools to assess the accessibility of a website, grouped in a total of 32 tools. These tools in the first phase were analyzed and characterized in under two properties such as scope (range of pages to be evaluated by request: a single page or the entire web site or group of pages) and license; table two shows the tools contained within this group.

Table 2. Online accessibility assessment tools.

Tool	URL	Scope	License
508 Checker	http://www.508checker.com/	Web Page	Free
A-Tester	http://www.evaluera.co.uk/	Web Page	Free
A11Y Compliance Platform	http://www.boia.org/	Web Site	Comercial
Accessible-email	http://www.accessible-email.nl/	Web Page	Free

AccessLint.com	http://accesslint.com/	Web Page	Open Source
AccessMonitor	http://www.acessibilidade.gov.pt/accessmonitor/	Web Page	Free
AChecker	http://achecker.ca/	Web Page	Open Source
Analizador WEB Ecuador	http://observatorioweb.ups.edu.ec/	Web Page	Free
Cynthia Says	http://www.cynthiasays.com/	Web Page	Free
DaSilva	http://www.dasilva.org.br/	Web Page	Free
Digital Content Checker	https://console.ng.bluemix.net/catalog/services/digital-content-checker	Web Page	Comercial
Dyno mapper	http://dynomapper.com/features/website-accessibility-testing	Web Site	Trial
Examinator	http://examinator.ws/	Web Page	Free
Functional Accessibility Evaluator (FAE)	http://fae20.cita.illinois.edu/	Web Site	Open Source
HTML_CodeSniffer	http://squizlabs.github.io/HTML_CodeSniffer/	Web Page	Open Source
MAUVE	http://hiis.isti.cnr.it:8080/MauveWeb/	Web Page	Free
Nibbler	http://nibbler.silktide.com/	Web Page	Free
Nu Html Checker	https://validator.w3.org/nu/	Web Page	Free
OzART	http://www.accessibilityoz.com/ozart/	Web Site	Comercial
Pa11y	http://pa11y.org/	Web Site	Open Source
Servicio de Diagnóstico en línea - España	http://forja-ctt.administracionelectronica.gob.es/web/caccesibilidad	Web Site	Private
Siteimprove Accessibility	http://siteimprove.com/features/web-accessibility/	Web Site	Comercial
Sitemorse	http://www.sitemorse.com/	Web Site	Comercial
SortSite	http://www.powermapper.com/	Web Site	Trial
Tanaguru	http://www.tanaguru.com/en/	Web Site	Open Source
Tenon	http://www.tenon.io/	Web Site	Free
Tenon (Monitor)	http://www.tenon.io/	Web Site	Trial
Tingtun Accessibility Checker	http://checkers.eiii.eu/en/pagecheck/	Web Page	Open Source
Vamolà	http://www.validatore.it/	Web Page	Open Source
WAVE	http://wave.webaim.org/	Web Site	Free
WCAG Compliance Auditor	https://www.funnelback.com/understand	Web Site	Comercial
WorldSpace	http://www.deque.com/products-old/worldspace/	Web Site	Comercial

This characterization led us to establish a first selection criterion of the tools that would be part of a second phase of study. This criterion is the type of license of use of the tool. We have selected only tools with Free or Open Source license, which reduce the list from 32 to 21 tools. In addition, three more tools were discarded because they presented execution errors. The final tools analyzed are the 18 included in table 3.

The comparative analysis is based on a series of defined criteria and supported largely by the W3C [6] for the selection of Web accessibility analysis tools. The specific criteria are: guidelines, configuration, language, coverage, repair recommendation, detail report, report formats, ARIA & HTML5, DOM support, API format, and score.

Then we proceed to detail them more thoroughly.

- **Guidelines**: category where the set of consistent rules are as determined by the referee, as an example WCAG1.0, WCAG2.0, Section 508, etc.

- **Configuration**: criterion that indicates whether the evaluator provides the ability to configure and adapt the evaluation to user needs, to adapt accessibility controls in order to achieve better performance in the analysis.
- **Paste/Send HTML code**: functionality that enables an evaluation pasting the HTML code of the web page (paste) or allows to upload (send) an HTML file. Important feature when evaluating page is local and is not published on a web server.
- **Language**: parameter in which the different types of available languages are defined in the tool.
- **Coverage**: some accessibility evaluation tools allow us to analyze an entire site or set of pages in the same analysis. With this approach, we determine the coverage offered distinguishing between: single pages or set of pages.
- **Repair recommendation**: This proposed criterion determine if a tool offers some kind of recommendation to resolve errors encountered during the analysis process.
- **Report detail**: parameter by which we measure the level of detail returned in the analysis report. We establish two possible categories: a basic level in which the result is a superfluous submission of the errors found, without any specification; and moreover a detailed level, obtaining a more comprehensive and concise report.
- **Report formats**: analyze the types of format that offers the tool to show the evaluation. They are represented in HTML format, however, there are a number to consider offering other possibilities such as: JSON, PDF, EARL, CVS, ODS, etc.
- **ARIA & HTML5**: through this category evaluate whether the tool considers rules or failures related to WAI-ARIA [9], included as part of HTM5.
- **DOM support**: parameter that establish whether the tool renders before evaluating the page, with the purpose of detecting all the inconsistencies generated by JavaScript.
- **API format**: defined to determine if the tools offer an API for automatically evaluation, such as web services under JSON or XML formats criteria.
- **Score**: the last attribute to consider in analysis defines whether the assessment tools offer a score of satisfaction with verified rules.

After defining each of the parameters established for the comparison of Web accessibility tools, we proceed to show the results obtained with each of the 18 selected tools (Table 3).

Table 3. Comparison of Web accessibility tools.

	Guidelines	**Configuration (Y:Yes / N:No)**	**Paste (P)/Send (S) HTML code**	**Language (Language codes - ISO 639)**	**Coverage (P: page / G: groups)**	**Repair recomendation (Y: Yes / N: No)**	**Report detail (B: Basic / D: Detail)**	**Report formats (-:No)**	**Supported formats HTML5**	**DOM support (Y:Yes / N:No)**	**API formats (- :No)**	**Score (Y: Yes / N: No)**
508 Checker	508	N		en	P	N	B	-	Y	Y	-	-
Accesible-email	No define	N	P	en	P	N	B	-		N	-	-
AccessLint.com	No define	N		en	P	N	B	-	N	N	-	-
AccessMonitor	WCAG 1.0 WCAG 2.0	Y	P S	pt	P	Y	D	-	N	N	-	Y

AChecker	508, BITV Stanca WCAG 1.0 WCAG 2.0	Y	P S	en gr it	P	Y	D	PDF EARL CVS	N	N	XML	-
A-Tester	WCAG 2.0	N		en	P	Y	D	-	N	N	-	-
Cynthia Says	WCAG 2.0 508 Section	Y		en	P	Y	D	-	N	N	-	-
Examinator	WCAG 2.0	N	P S	es	P	N	D	-	N	N	-	Y
Functional Accessibility Evaluator (FAE)	WCAG 2.0 (A, AA)	Y		en	G	N	D	-	Y	Y	-	Y
HTML_Code-Sniffer	WCAG 2.0 (AAA) Section 508	Y	P	en	P	N	B	-	N	Y	-	-
Nu Html Checker	WCAG 2.0 (A,AA, AAA)	N	P S	en	P	N	D	-	N	N	-	-
OAW Ecuador	WCAG 2.0 (A,AA,AAA)	Y		es	P	N	D	-	N	Y	JSON XML	-
Tanaguru	RGAA (A, AA, AAA), Accessiweb 2.2	Y		es en fr	P	N	D	ODS CSV PDF	N	Y	-	Y
Tenon	WCAG 2.0 (AAA)	Y	P	en	P	N	D	JSON CVS	Y	Y	JSON	-
Tingtun Accessibility Checker	WCAG 2.0	N		en	P	N	D	PDF	N	Y	-	Y
Vamolà	Allegato A L.4/04, WCAG 2.0	Y	P S	en	P	Y	D	-	N	N	-	-
WAVE	Section 508, WCAG 2.0 (A, AA)	N		en	P	N	D	-	Y	Y	JSON XML	-
TAW	WCAG 2.0 (A,AA, AAA)	Y		es	P	N	D	-	N	N	-	-

Analyzing the results, we can see that; practically all the tools comply with all rules of WCAG 2.0, the English language predominates in their interfaces, except a tool, the rest just has implemented the analysis to a single page request. Most tools do not offer any recommendations to facilitate the correction of the errors found and often display a more detailed report.

As for the rendering of the page, much of the tools analyzed can find accessibility errors caused by the DOM. However, there are a few tools that provide the report in another format other than through the display of online content, as well as providing an API to consume its functionality automatically via Web services.

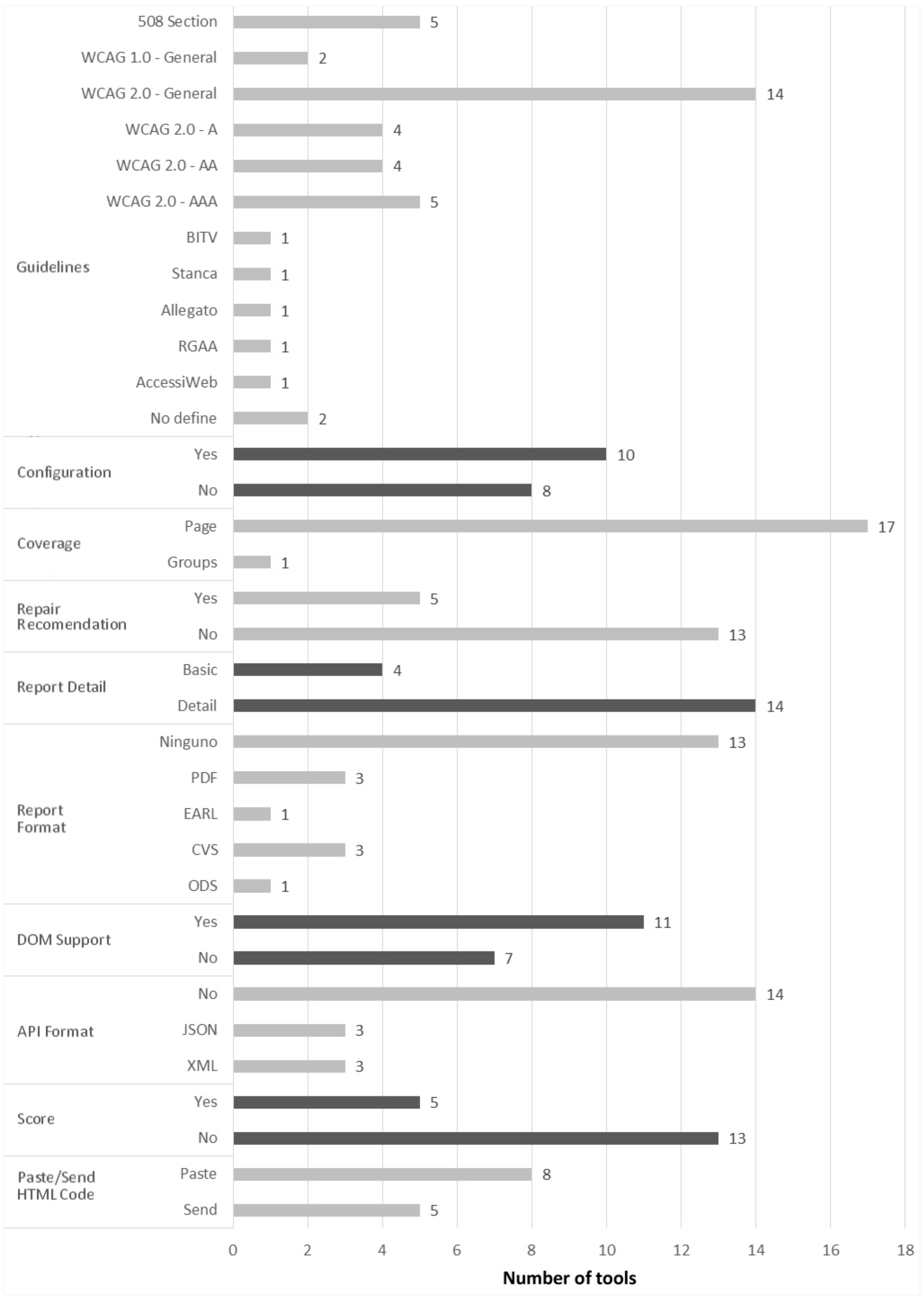


Fig. 1. Results of the comparison of Web accessibility tools.

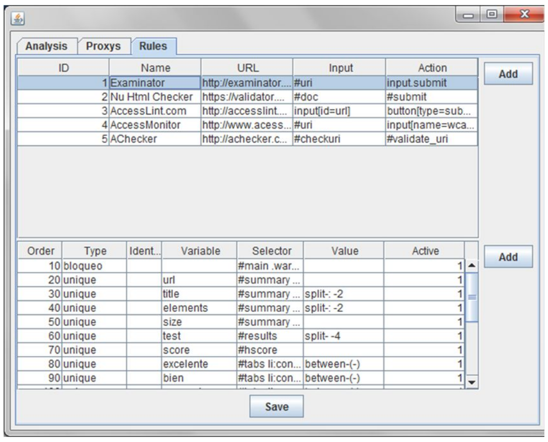
The results of the analysis allow us to establish a ranking of reliability, which will be detail in the next section.

4. Reliability of the tools

In addition to the comparative analysis under the mentioned criteria in the previous section, it was important to analyze for each tool the reliability of the results and the degree of successes to an assessment, this in relation to the number of aspects analyzed by each tool for accessibility errors on a page.

For this analysis a corpus of test cases, cases that are themselves potential accessibility errors on a web page, was built. The construction of this corpus initially agreed to identify areas analyzed and identified as failures by online accessibility evaluation tools of web pages. For this, each tool would pass a set of web pages for evaluation and subsequently detected faults be extracted. Given the number of tools and variety of websites, an application that automates the above process was designed. Then being designed an application of extraction, transformation and loading (ETL), which consisted of: extracting the results in HTML format obtained from the evaluation of a tool, transformation of these results into a common format, load them in a database and export data to spreadsheet for further processing and analysis.

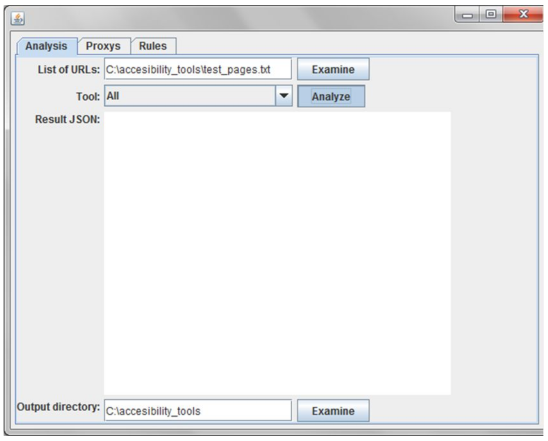
The application consists of a rules engine based on CSS and HTML selectors configurable as a tool for automatic evaluation engine, using the rules we can get, in a structured way, the results returned by a tool after the application for evaluation of a web page. Figure 2 shows the definition of rules for each analysis tool and figure 3 displays the input parameters for the call to the execution of the ETL process.



ID	Name	URL	Input	Action
1	Examinator	http://examinator....	#uri	input.submit
2	Nu Html Checker	https://validator....	#doc	#submit
3	AccessLint.com	http://accesslint....	input[id=url]	button[type=sub...
4	AccessMonitor	http://www.acess...	#uri	input[name=wca...
5	AChecker	http://achecker.c...	#checkuri	#validate_uri

Order	Type	Ident...	Variable	Selector	Value	Active
10	bloqueo			#main .war...		1
20	unique		url	#summary ...		1
30	unique		title	#summary ...	split-: -2	1
40	unique		elements	#summary ...	split-: -2	1
50	unique		size	#summary ...		1
60	unique		test	#results	split- -4	1
70	unique		score	#hscore		1
80	unique		excelente	#tabs li:con...	between-(-)	1
90	unique		bien	#tabs li:con...	between-(-)	1

Fig. 2. Defining data extraction rules.



List of URLs: C:\accesibility_tools\test_pages.txt
Tool: All
Result JSON:
Output directory: C:\accesibility_tools

Fig. 3. Input parameters and execution of the ETL tool.

In figure 3, the first input field corresponds to the path of a text file with the set of URLs of pages to be evaluated, the second input field corresponds to the specification of the tools that the results will be drawn, while the third and fourth fields exposed outputs of the application, expressed as a text string in a common format such as JSON for easy processing and a spreadsheet with the results, respectively.

With the ETL tool a group of 100 web pages corresponding to higher education institutions was selected and the rules of 5 tools randomly selected of the group of 18 tools are also parameterized. After running the application several errors (errors and warnings) for each tool were obtained, these errors were classified and grouped semantically, identifying a total of 63 types of errors.

A second phase of construction of the corpus of evidences was based on the study of Techniques and Failures for Web Content Accessibility Guidelines 2.0 [7], with particular attention to the group of common failures defined by WCAG 2.0 [3], further identified 48 failures other than those identified using the ETL tool. In addition to these errors other were added according to the experience of the authors, thus conforming a total of 127 types of accessibility errors. Already identified the possible errors, we proceeded to develop for each error the corresponding HTML code of the accessibility failure, then given for each error a test case.

Each test case was analyzed and classified according to the WCGA 2.0, principles, guidelines and its relation to compliance levels and also classified according to whether detection can be automated / fully programmed or if required a manual check, either partially or completely; classification shown in Table 3 and Table 4.

Each test case was analyzed and classified by WCGA 2.0, principles, guidelines and turn on the levels of compliance that relate and also ranked according to whether their detention can be automated / scheduled full or if required a manual check, either partially or totally; classification shown in Table 4 and Table 5.

Table 4. Corpus test case for Principles and Conformance Level

Principles and Conformance Level	Automatic	Manual	Total Test Case
Perceivable	43	18	61
A	35	15	50
AA	3	1	4
AAA	5	2	7
Operable	25	5	30
A	17	5	22
AA	2	0	2
AAA	6	0	6
Understandable	18	7	25
A	11	3	14
AA	1	3	4
AAA	6	1	7
Robust	10	1	11
A	10	1	11
Total	**96**	**31**	**127**

Table 5. Corpus test case for WCAG Principles and Guidelines

Principles and Guidelines	Total test cases
Perceivable	61
1.1 Text Alternatives	16
1.2 Time-based Media	2
1.3 Adaptable	29
1.4 Distinguishable	14
Operable	30
2.1 Keyboard Accessible	4
2.2 Enough Time	5
2.4 Navigable	21
Understandable	25
3.1 Readable	12
3.2 Predictable	4
3.3 Input Assistance	9
Robust	11
4.1 Compatible	11
Total	**127**

It can be seen the classification of the 127 test cases, with 96 cases of automatic verification and 31 of manual detection, as mentioned, the automatic verification is easily programmable so it could be automated by a tool. For example: the absence of a title page (F25: Failure of Success Criterion 2.4.2 page Titled, due to the title of a web page not Identifying the contents [3]). Moreover manual verification means that the error detection requires expert judgment, or may be programmable using advanced programming techniques such as language processing, image processing techniques, artificial vision, etc., an example of this error would be: formatted text used instead header element (Failure of Success Criterion 1.3.1 info and Relationships, due to using Changes in text presentation to convey information without using the appropriate markup or text [7]). Although some cases have been identified as manual detection, also they have been included in the study, it is important to analyze the behavior of the tools in these cases, it is possible that some tools could identify partially as warnings accompanied by manual verification or the best detected as failures using advanced programming techniques.

Once defined and validated the corpus of evidences, a set of web pages that included test cases without these conflicts with each other were created, and then we proceeded to evaluate them with the 18 tools object of this study, and finally analyze the behavior of the tool in each case translated into this behavior:

- If the error was detected by the tool completely and in correspondence to the case evaluated,
- If the error was detected partially and requires supplemented by a manual check in correspondence to the case assessed, or
- If the error was not detected or treated by the tool.

After making evaluations, the results revealed differences between the approaches of each tool and attention giving one or another tool to potential accessibility errors committed on a website. Of the 127 errors, including all tools 109 errors are covered, i.e., an error has been detected in whole or in part at least by a tool, but a tool by itself is not enough to cover 50% of possible errors (Table 6).

Table 6 shows the relationship between the number of tools that have detected a number of cases and identified as an error, noting that of the 127 errors, 18 have not been detected by any tool, being then 109 errors that have been detected by at least one tool. Importantly, these 18 errors belong to the group of 31 cases identified as possible manual detection, the difference (13 errors of manual detection) have been detected by some tools but accompanied by a suggestion of manual verification, being classified as partial detection by the tool. The partial rating was also used to test cases of automatic detection, in which part of the evaluated error is detected.

Table 6. Relationship between the number of tools and the number of detected cases.

Number of tools	Detected cases
None	18
1	18
2	13
3	19
4	9
5	7
6	7
7	3
8	7
9	6
10	2
11	3
12	2
13	3
14	4
16	2
17	3
18	1
Total	127

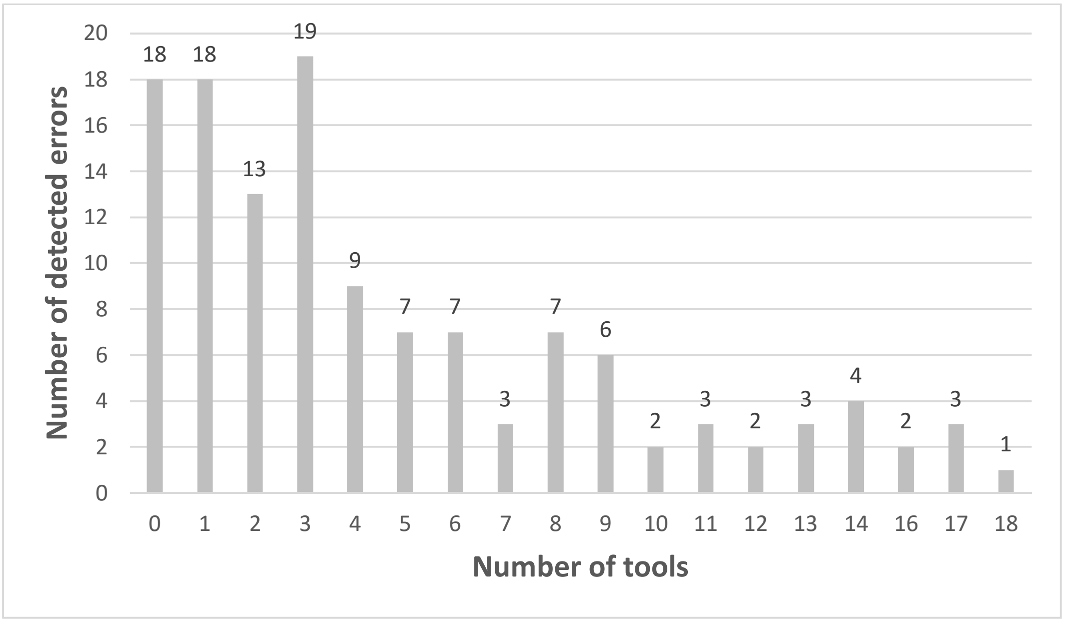


Fig. 4. Relationship between the number of tools and the number of detected cases.

In relation to the number of errors detected by the tools, Table 7 shows performance data for each tool in detecting whether or not a failure, allowing us to appreciate the effectiveness of each tool equivalent to the degree or percentage of correct answers or detections according to the corpus of testing.

Table 7. Evaluation results of test cases per tool.

	Complete detection	Partial detection	Cases detected	Reability (%)
508 Checker	33	5	38	32.57
Accessible-email	3	0	3	2.75
AccessLint.com	18	1	19	16.97
AccessMonitor	42	2	44	39.45
AChecker	35	6	41	34.86
A-Tester	27	3	30	26.15
Cynthia Says	30	7	37	30.73
Examinator	39	0	39	35.78
Functional Accessibility Evaluator (FAE)	33	9	42	34.40
HTML_CodeSniffer	33	1	34	30.73
Nu Html Checker	21	0	21	19.27
OAW Ecuador	39	1	40	36.24
Tanaguru	36	0	36	33.03
Tenon	47	0	47	43.12
Tingtun Accessibility Checker	25	1	26	23.39
Vamolà	31	4	35	30.28
WAVE	38	3	42	36.24
TAW	41	3	44	38.99

As shown in the table, the highest number of errors detected by a tool is 47, which corresponds to Tenon, equivalent value to 37% of all errors evaluated and 43.12% of all errors detected by the toolkit object of the study.

The number of errors detected by the various tools is varied and given that some of the errors have not been detected in full but partially, to estimate the degree of successes of a tool to the corpus of evidences, these partial detections has considered with a rating equivalent to half of a hit, a situation that has been reflected in the formula (1) for calculating the Reliability of a tool.

$$Reliability_{tool} = \frac{Complete\ detection_{tool} + Partial\ detection_{tool}\ \times\ 0{,}5}{Total\ cases\ detected\ by\ all\ tools} \times 100 \tag{1}$$

Estimated the degree of successes for all tools we could determine the effectiveness of them in the evaluation of web pages accessibility in the corpus of evidences. Tenon, AccessMonitor and TAW are the three tools with a higher number of errors detected, covering between the three the 70.64% of the errors detected by all tools, and if you add OAW Ecuador could cover 80.73%.

5. Conclusions

One tool is not enough to assess the accessibility of a web page as there will be cases that errors in a web page will be detected by a tool but not by another, since each tool focuses on certain accessibility evaluation cases. Out of 18 tools, just one managed to detect 40% of the corpus test that was created (Tenon). If this tool is linked to the results of other three tools

(AccessMonitor, TAW, OAW), the percentage of detected errors increases and detects up to 80% of faults. Therefore, it is essential to combine the results of more than one tool.

If all the tools present their evaluation services through an API, we might get an "interface tool" that consume the results of each and through semantic analysis can issue consolidated results, results that increase the degree of verification of the accessibility of a web page, as more accessibility errors would be detected if any.

The tools that have been analyzed, not yet implemented advanced programming techniques to detect certain errors, which implies a manual verification or expert judgment, which so far the tools have done is to detect possible error or susceptible fault case and indicate to the user that require manual verification. There is still a broad field of study and development of assess tools that allow the detection of manual errors, for instance by applying advanced programming techniques such as image processing, language processing and other techniques which derive from artificial intelligence.

As for other similar works to that presented here, since 2008, the year of publication of WCAG 2.0 standard, in which the analyzed tools are based, has only been found a similar work [8]. In this work, six tools were compared. The only tools that meet both works are AChecker and TAW. Regarding these tools, the two studies agree that AChecker gets better results than TAW.

Acknowledgements

Thanks to ESVI-AL EU project, and to the “Master on Software Engineering for the Web” of the University of Alcalá.

References

1. Convention on the Rights of Persons with Disabilities. United Nations (2006), http://www.un.org/disabilities/convention/conventionfull.shtml. Accessed April 28, 2016
2. CRPD List of Countries. Disabled World (2016). http://www.disabled-world.com/disability/discrimination/crpd-milestone.php, Accessed April 28, 2016
3. Failures for WCAG 2.0. World Wide Web Consortium (2015). https://www.w3.org/WAI/GL/WCAG20-TECHS/failures.html, Accessed April 28, 2016.
4. Feingold, L.: Digital Accessibility Laws Around the Globe (2016). http://www.lflegal.com/2013/05/gaad-legal/, Accessed April 28, 2016
5. ISO/IEC 40500:2012. Information technology -- W3C Web Content Accessibility Guidelines (WCAG) 2.0. International Organization for Standardization (2012)
6. Selecting Web Accessibility Evaluation Tools. World Wide Web Consortium (2016). https://www.w3.org/WAI/eval/selectingtools/, Accessed April 28, 2016
7. Techniques for WCAG 2.0. World Wide Web Consortium (2015). https://www.w3.org/WAI/GL/WCAG20-TECHS/, Accessed April 28, 2016.
8. Vigo, M., Brown, J., Conway, V.: Benchmarking web accessibility evaluation tools: measuring the harm of sole reliance on automated tests. Proceedings of the 10th International Cross-Disciplinary Conference on Web Accessibility, pp. 1-10. ACM Press (2013).
9. WAI-ARIA Convention on the Rights of Persons with Disabilities. United Nations (2006), http://www.un.org/disabilities/convention/conventionfull.shtml. Accessed April 28, 2016
10. Web Accessibility Evaluation Tools List. World Wide Web Consortium (2016). https://www.w3.org/WAI/ER/tools/, Accessed April 28, 2016
11. Web Content Accessibility Guidelines (WCAG) 2.0. World Wide Web Consortium (2008). https://www.w3.org/TR/WCAG20/, Accessed April 28, 2016