Flexible Automatic Support for Web Accessibility Validation

BROCCIA GIOVANNA, CNR-ISTI, HIIS Laboratory, Italy
MARCO MANCA, CNR-ISTI, HIIS Laboratory, Italy
FABIO PATERNO', CNR-ISTI, HIIS Laboratory, Italy
FRANCESCA PULINA, CNR-ISTI, HIIS Laboratory, Italy

Automatic support for web accessibility validation needs to evolve for several reasons. The increasingly recognised importance of accessibility implies that various stakeholders, with different expertise, look at it from different viewpoints and have different requirements regarding the types of outputs they expect. The technologies used to support Web application access are evolving along with the associated accessibility guidelines. We present a novel tool that aims to provide flexible and open support for addressing such issues. We describe the design of its main features, including support for recent guidelines and tailored results presentations, and report on first technical and empirical validations that have provided positive feedback.

CCS Concepts: **Human-centered computing→Systems and tools for interaction design**; User interface toolkits

KEYWORDS

Accessibility; Automatic validation; Guidelines; Tailored presentation of validation results

ACM Reference format:

In EICS '20: ACM SIGCHI Symposium on Engineering Interactive Computing Systems, June 23–26, 2020, Sophia Antipolis, France. ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

Providing accessible websites is becoming an increasingly important topic for people's daily lives since it results in services that can be perceived, understood and used by all users including users with disabilities. In several countries there are various efforts to promote best accessibility practices (see for example [13], [14]), and recently the European legislation (European Accessibility Directive, WAD, 2016/2102 on the accessibility of the websites and mobile applications of public sector bodies) [10] has further promoted the right of disabled people to have access to online public information and services. In addition, the WAD directive places quite a substantial emphasis on the periodical monitoring of the accessibility conformance levels of public bodies' websites, stressing that quantitative information should be regularly collected and that the procedure should be transparent, reproducible and lead to comparable results.

In order to obtain accessible applications, the World Wide Web Consortium (W3C) started the Web Accessibility Initiative (WAI), whose strategies, standards, and supporting resources have deeply influenced national and international legislation, in particular through the Web Content Accessibility Guidelines (WCAG). Such guidelines have evolved over time, and recently the W3C has put forward the WCAG 2.1 [9] version. The new guidelines were initiated with the goal to improve accessibility

guidance to address mobile accessibility – which refers to making web resources more accessible when accessed with mobile devices – and the needs of people with low vision and cognitive or learning disabilities. One further important contribution within the WAI initiative, from the perspective to improve automatic support, is the Evaluation and Report Language (EARL) [1] – a machine-readable format for expressing test results – promoted as a standard for coding accessibility test results performed automatically as well as manually, in order to support the interoperability among tools and experts.

In addition, in order to obtain more consistent automatic validation support, the W3C/WAI has started to consider how the accessibility requirements are expressed, described, and also linked to the relevant techniques to be followed in order to be compliant to them. This leads to the introduction of a shared common format for writing rules to test accessibility, the Accessibility Conformance Testing (ACT) rules format[2]. The ACT rules are useful when developing automated testing tools and manual testing methodologies since, for each test required by the WCAG, they offer a shared interpretation of what elements the rule is applied to, and what are the checks to be made automatically and/or manually. This enables transparency and harmonization of testing methods, including methods implemented by accessibility testing tools.

Unfortunately, checking the accessibility according to the WCAG guidelines is often a tedious and error-prone process, not only because the requirements are sometimes difficult to understand without deep knowledge in web technologies, but also because it implies looking at many types of details at the same time. In order to address such problems, tools called accessibility validators were soon introduced. Such tools aim to reduce the cost of validation, increase consistency in the identification of the problematic parts, and extend the set of features that can be assessed [14]. Even if such tools are not able to identify all the possible accessibility problems, since in some cases human intervention is necessary for this purpose [24], their support is greatly appreciated, especially in public services where the amount of content to validate is growing exponentially.

As Abascal et al. [1] indicate, such tools can be classified according to various criteria: the type of license (free versus commercial); the platform where they can be executed; the evaluation scope (ranging from single pages to entire websites); the support provided for repairing identified issues; how the evaluation results reporting methods, guidelines supported and detected issues are rendered, that can be a more graphical or code-oriented view, and also exported in different formats (mainly HTML, PDF, EARL).

Over time several validators have been proposed (e.g. [3], [4], [11], [11], [12], [18], [25]). One significant effort was the European Internet Inclusion Initiative Checker (EIII Checker), which was designed and implemented as part of the EIII project supported by the European Commission [20], [22]. This was a Coordination and Support Action running from October 2013 until September 2015. However, the only public result of that project that is still available is the Tingtun checker[3], which analyses the page content against tests derived from the success criteria of the old version of the WCAG, reporting results in terms of “pass”, “fail” or “to be verified”, which are more oriented to people with programming knowledge.

More generally, the W3C provides a list of web accessibility evaluation tools[4] based on descriptions provided by their authors. Unfortunately, often such tools are not able to keep up with the latest technology and become rapidly obsolete [23]. For example, mobile browsing has become increasingly widespread, affecting the technical design and development of websites, which nowadays must be able to be viewed on a variety of devices and different screen resolutions. The WCAG 2.1 aim to provide support for checking accessibility also in these cases, but so far only a small portion of automatic tools

[1] https://www.w3.org/WAI/standards-guidelines/earl/
[2] https://www.w3.org/TR/act-rules-format/
[3] http://checkers.eiii.eu/
[4] https://www.w3.org/WAI/ER/tools/

have aimed at addressing them, such as for example ARC Toolkit by The Paciello Group[5] (which is available as a Chrome plugin), TAW by CTIC Technology Centre[6] and WAVE by WebAIM[7].
Overall, there is a need for a new generation of accessibility validators aiming to address the various emerging challenges determined by the technological evolution and the growing importance that accessibility is acquiring in our society. In this paper, we present a novel accessibility validator tool, which is able to support the new guidelines for mobile browsing (WCAG 2.1), analyse also entire websites, present the validation results in a way that can be tailored for different types of users, provide the evaluation report in various formats (including the standard one), and that is available for open free access. We report on its application on a representative set of web pages, and compare it with those obtained through a tool with similar goals, and also present a usability study that provided useful feedback on the understandability of the validation results and some useful indications for further small improvements.
After a discussion of related work in Section 2, we present the approach and the main features of MAUVE++, then detail how the support for the WCAG 2.1 has been implemented. In the following, we describe the design of how the tool provides validation results for different types of stakeholders. Then, we present its technical validation performed on a set of representative websites, and report on the usability test carried out with the user feedback received. Lastly, we draw some conclusions and provide indications for future work.

2 RELATED WORK

Several automatic and semi-automatic tools for checking accessibility compliance have been put forward, and already represent useful instruments to support developers, public officials and decision-makers in reducing the time and effort in conducting an accessibility evaluation. However, they often present a number of limitations in the quality, completeness, and accuracy of the provided support. In fact, tools cannot guarantee the automatic checking of the accessibility guidelines that require human evaluation, sometimes they produce false positives and false negatives, and lastly, the completeness of the results is not the same for all tools [1]. In addition, they often are not are not transparent in indicating what they cover. Thus, experts and policymakers continuously pose demands for better tools to enable fast processing of dynamic and large volumes of web pages, and their content and data, the detection of accessibility hurdles, and assistance in their repair.
In the area of tool support for accessibility, the work by Miniukovich et al. [17] has focused on readability issues, and provides some automatic support but only for assessing some text-related properties. The contribution by Moreno et al. [19] does not aim at validating accessibility guidelines but it focuses on providing some automatic support for simplifying textual expressions. Previous work [25] proposed a categorization of tools for the automatic evaluation of multiple guideline sets based on the way in which guidelines are internally codified and expressed (through an external language or an internally defined one), and also distinguishing between "deterministic" and "probabilistic" approaches to the validation, which means tools that have a clear distinction of what can be automatically checked and tools that aim to provide some automatic support also for guidelines that need a human check. Another possible way to classify accessibility evaluation tools is to consider the following features: if they are free or commercial; their type of platform (standalone application, browser extension, online service); their evaluation scope; if they offer support features for the repairing of the identified accessibility issues, in addition to the evaluation; how accessibility issues are reported to the user; how the guidelines are supported [1].

[5] https://www.paciellogroup.com/toolkit/
[6] https://www.tawdis.net
[7] WAVE by WebAIM

Several tools have been put forward in this area but for some reasons they had limited impact. Some are just local efforts that do not even support the English language (such as Hera FFX [12] and Vamolà [18]). Other tools only focus on limited accessibility aspects (for example they only analyze color contrasts). Moreover, there are commercial tools that cannot be freely used, such as Imergo[8], Deque Systems[9], Siteimprove[10], Make Sense[11].

In our analysis of the state of art in this area, we have also considered various technical aspects: the audit depth of these tools, that is their validation scope (if they validate single web pages and/or entire websites, and also password protected pages); the validation source, that is the way users can start the validation of the web page (copied URL, file upload, snippets of code directly entered by the user). We have checked if they allow the customization of accessibility audits, for example if the user can specify the guidelines or their conformance level for the validation. We have also focused on the presentation of test results provided. First, we identified the report type: if it is mainly code-oriented, statistical, or graphical (or a mix of these features). Then, we looked at the distinction they make between different types of problems: errors, warnings, potential problems and information, that usually require additional manual check. In the end, we looked at both the summarizing overview and the full report of the validation that they provide, and the kind of filtering that is possible to apply to the results in order to get specific information.

We noticed that current accessibility tools provide limited opportunities for customization, both of the accessibility audit and the evaluation results. For example, among freely available tools the customization of accessibility audits is provided by the majority of them, such as AChecker[12], Cynthia Says[13], TAW Web[14]. However, there are different levels of customization: some of them only allow the selection of the preferred guidelines, others also the selection of the priority level of the guidelines.

Moreover, the presentation of such results is mainly code-oriented, thus it does not always enable all users to fully understand the detected accessibility barriers. Among the freely available tools that we have considered, only WAVE[15] and the Siteimprove plugin provide a graphical report, pinpointing accessibility barriers in the rendered webpage. DESTINE [6] was a useful initial contribution to highlight the importance of more flexible reports. Here we propose a more systematic approach for such reports, identifying more clearly the target user roles, reporting a richer set of pieces of information, also because we support validation of the recent WCAG 2.1 guidelines (which are much more structured and detailed than those that it addressed), and we also provide user feedback on the usability of the generated reports. Thus, we address the need for advanced solutions able to help even accessibility experts, not only developers, to fully understand accessibility barriers more easily and in a more cost-effective manner, so as to better act upon them.

As regards the supported guidelines, the continuous evolution of web technologies and the associated accessibility guidelines requires continuous effort from developers of the validation tools in order to keep them updated. This is demonstrated by the still limited number of tools currently supporting the WCAG 2.1, which have recently become a W3C recommendation (5 June 2018[16]). Moreover, among the freely available tools that support also the latest version of the WCAG, there are tools that check only the accessibility barriers on a particular feature of the webpage (they are mainly colour contrast

[8] https://imergo.com/solutions/imergo.html
[9] https://www.deque.com/products/
[10] https://accessibility.siteimprove.com/
[11] http://mk-sense.com/
[12] https://achecker.ca/checker/index.php
[13] http://www.cynthiasays.com/
[14] https://www.tawdis.net/
[15] https://wave.webaim.org/
[16] https://www.w3.org/TR/WCAG21/

checkers, as for example A11y Color Contrast Accessibility Validator[17]), and others that are not completely free (they offer a free evaluation of a maximum of 5 pages, e.g. Access Assistant Community Edition[18]), or that at this time do not support the AAA level of conformance (e.g. WAVE evaluator).
Even previous studies on automatic web accessibility evaluation (e.g. [2]) have only considered support for the previous WCAG 2.0 guidelines. A study [4] has considered a set of guidelines for mobile apps accessibility, and applied them to a set of Android apps, but the validation was carried out in a completely manual manner through a kind of heuristic evaluation exercise, which can provide limited information and requires particular effort.
Overall, an integrated, holistic and comprehensive solution to web accessibility assessment is still lacking. One important drawback is that current accessibility tools provide their users with rather limited opportunities for customization, therefore accessibility tools are not sufficiently flexible in coping with the various needs raised by their users. For example, the presentation of the results collected after the test does not always enable users to fully understand the discovered accessibility problems. It would be desirable that a validation environment provides customized presentations of its results based on the role of the interested user. Indeed, there are accessibility-related features in which web developers are more interested, and others that are more relevant to non-accessibility experts and public officials.

3 THE TOOL FEATURES

Our approach aims to indicate how a new generation of tools should address the emerging needs for large scale accessibility validation by providing a solution that is open, interoperable by supporting standard semantic interpretations of the accessibility rules, extensible with limited effort to new guidelines, flexible in defining what to validate (single pages, groups of pages, entire web sites) and in reporting results tailored for different relevant roles, and able to support validation according to the hierarchy of accessibility requirements (principles, guidelines, success criteria, techniques) and for different types of devices. It also shows that accessibility tools need to better consider the information needs of the various relevant stakeholders (Web commissioners, web content and application developers, accessibility experts, end users) when reporting the validation results.
The proposed tool (MAUVE++) takes into consideration previous work [25], and extends it in several directions, addressing a number of issues not considered previously:

- First of all, the proposed tool is able to analyze websites according to their compliance to WCAG 2.1 guidelines, which provide 17 additional success criteria to address barriers on mobile accessibility and the needs of people with low vision and cognitive or learning disabilities.
- It provides a double view of the accessibility results: one is a code-oriented view – where errors and warnings are presented in the line number of the evaluated web page source code which generates them, the other one is a graphical view for users without programming skills, which shows errors and warnings through the help of charts and tables. Moreover, it provides an overview of the validation results showing the number of errors, warnings and successes and a measure called *accessibility percentage*, which indicates how much the evaluated resource is accessible.
- It is able to evaluate the accessibility of entire websites. Users who want to use this service just need to register to the tool (the registration is completely free).
- In addition to the XML report and the code-oriented web report, the proposed tool can now provide three new outputs:
    - A report in EARL, language built on top of the Resource Description Framework (RDF) – which can be expressed using different formats among which JSON-LD.

[17] https://color.a11y.com/?wc3
[18] https://www.webaccessibility.com/tes

Being the W3C standard for expressing accessibility test results, the new report in EARL/JSON-LD supports the interoperability among tools and human experts;
    - A report in PDF containing the results of the evaluation of entire websites, which is sent via e-mail to registered users.
    - An "end user" web report, which, as mentioned previously, shows the results of the validation in a graphical view through charts and tables.
- Finally, the tool has a responsive user interface, which allows its use from a mobile device, and support multidevice scenarios where the user checks the accessibility issues on the mobile device while working on the web page specification on a desktop workspace.

As shown in Figure 1, the validation can be performed through a web interface or a Google Chrome plugin. In this last case, if the validation is performed through the plugin the evaluation of the DOM of the web page currently running on the browser is triggered, including dynamic elements created after the page has been loaded. Otherwise, if the validation is performed through the web interface it is possible to choose among the single web page and the multiple pages' validation.
In case of single web page validation, it is possible to validate the page by entering its URL, by uploading the HTML file from the local computer, or by pasting its code inside a dedicated form. In the case of multiple pages' validation, users have to set the base URL of the website and the crawling configurations (i.e. select the number of pages to fetch and the crawling depth). In both cases, the web interface enables selecting one of the predefined accessibility guidelines and the level of conformance, and the version of the web page specific to a certain type of device (e.g. personal computer, smartphone, tablets, etc.).
The tool engine is the module which, after retrieving the selected guidelines specification, actually performs the page(s) validation. Thus, it analyses the elements in both the DOM and associated CSS files, it considers the techniques that are relevant to the analysed elements, and applies the checks defined in the guideline specification file for the considered element. Based on the results it decides whether the check results it is a success, an error, or a warning (which indicates that the check cannot be done fully automatically and that the occurrence could be a potential error).
As a result, the validation can produce:

- a XML report, useful in case the validation tool is used as a library by other software;
- a report in EARL;
- a PDF report with the results of the evaluation of the entire website, that the registered user receives by email;
- a web interface report of single web page validation, which has a separate view according to whether the user is a web developer or an end user without programming knowledge;

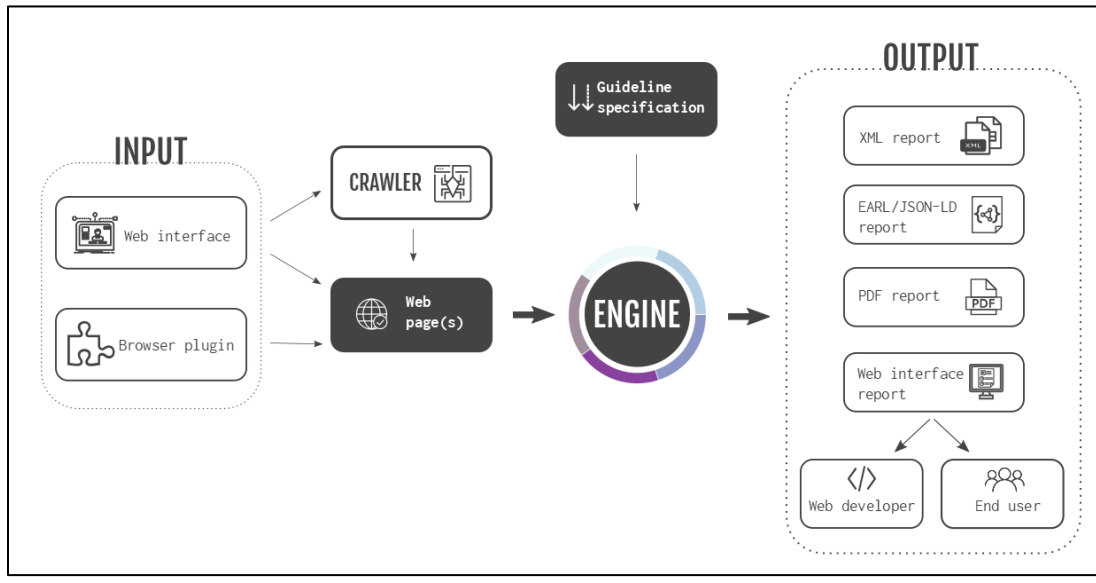


Figure 1. The proposed tool architecture

In the following we detail how the support of the additional success criteria related to the WCAG 2.1 has been introduced, the design of the flexible results view, and how the validation of entire websites is supported.

3.1 Support for WCAG 2.1

As explained previously, in order to make web content more accessible, in 1999 the W3C defined the WCAG guidelines. This first set of guidelines was soon found not adequate to support the validation of content because it was not sufficiently complete and precise in its definition. Thus, in 2008 a new version – WCAG 2.0 – was put forward to address such issues, being independent of the technology used to produce web content, and applicable also to dynamic content. Recently, in June 2018, a new effort has been carried out with the goal to better support people with cognitive and learning disabilities, people with low vision, and mobile browsing, which has become the most prevalent form of content access also for impaired people.

The WCAG are structured into four layers:

- **Principles**: the web content should be Perceivable, Operable, Understandable, Robust; the grouping of guidelines into these four principles has been introduced with the 2.0 version;
- **Guidelines**: for each principle, the guidelines provide the basic goals that authors should work toward in order to make content accessible, according to the principle they are related to;
- **Success Criteria (SC)**: for each guideline, SC are provided as testable statement representing the requirements to be compliant to the standard. In most cases, they can be tested automatically by a software, but sometimes the manual intervention of an expert is required in order to assess whether the criterion is satisfied or not; at this layer, the WCAG 2.0 introduced the three Levels of Conformance, that classify the success criteria based on their impact on the overall accessibility level of the web content: A (low), AA, AAA (high);
- **Sufficient and Advisory Techniques**: each SC is associated with a number of informative techniques, that could be sufficient for meeting the compliance, or just advisory suggestions on how to improve the web content compliance with respect to the considered success criterion. Techniques themselves are divided into categories, depending on the intervention they describe; this category is indicated in the initial letter(s) of the technique name (e.g. H*x* is related to HTML, C*x* is related to CSS, G*x* is a general technique). Among techniques, there are also the

so-called Failures (starting with the capital F): the failure of just one of such techniques causes the full non-compliance with the success criterion.

Our tool specifies the guidelines verification through a set of checkpoints consisting of one or more checks which express the techniques that must be met by one or more components of the web page to make it compliant with a specific SC.
The WCAG 2.1 introduced 17 new success criteria, in addition to the success criteria of the 2.0 version. The 17 new success criteria are spread across different guidelines, even if some guidelines are more involved than others: e.g., guideline 1.3 (Adaptable) has been updated with 3 new SC, guideline 1.4 (Distinguishable) with 4 new SC, and the guideline 2.5 (Input Modalities) has been created with 6 new SC. The main aim of this effort is to guarantee that:

- Web content can be presented in different ways without losing information and structure, for example changing the orientation of a mobile device; a possible use case is a person with a physical disability who must use the horizontal rotation of the tablet attached to the wheelchair;
- Web content is easy to read and to hear, and information in the foreground is effectively separated from the background; relating to mobile access of content, there is, for example, need to avoid excessive horizontal or vertical scrolling for a person with low vision who increases the text size;
- The provided functionalities can be accessed through various inputs beyond keyboard, even from a mobile device which requires touch input with pointer gestures; a possible use case is a person with low dexterity of fingers who needs a simpler alternative to a multi-point touch gesture, as the pinch-to-zoom, to be able to access the zoom functionality.

For supporting these features, the new SC of WCAG 2.1 have been specified using the LWGD language [25] for guidelines specification. For each new SC, we defined the checkpoints to be evaluated in order to assess if the relevant accessibility techniques have been applied or not.
To support the validation of the novel criteria, we implemented a number of new checking functionalities. Such implementations affected both the guideline specification and its management, introducing new terms and conditions to evaluate (e.g. the terms related to the media rules), and their management in the tool engine, which has to check the conditions for the new terms (e.g. the presence of a certain *feature* with a specific *value* in a @media *rule*, and of certain CSS *properties* inside it). In particular, to assess the compliance with some SC of WCAG 2.1, we implemented two relevant engine functionalities in order to:

- analyse the CSS code, and check if there are specific media rules (e.g. orientation) accounting specific features (e.g. landscape/portrait);
- check the CSS style properties specified for the hover and focus state of an element (thus considering pseudo-classes).

Such checks are useful to assess the accessibility of contents in particular for people with problems in visual perception. In general, the Orientation success criterion has been introduced to firstly guarantee that the web content will be accessible from mobile devices regardless of the orientation (landscape or portrait mode) that the user chooses for browsing the resource.
With the aim to facilitate the mobile browsing where users want to complete tasks faster, the WCAG 2.1 also address the cases when input is required from users to facilitate the form completion task, in particular for people with cognitive disabilities. Input elements (e.g. `textarea`) should have `autocomplete` attributes specified with specific values, in order to save the user's cognitive effort of completing the same field with the same value several times. Thus, also this type of check has been introduced.
As regards the accessibility of contents, the proposed tool is also able to check whether the user can reflow the content written into HTML containers without requiring excessive scrolling in the two directions (horizontal or vertical). This aspect is particularly useful when accessing the content with a

mobile device whose screen size is obviously smaller respect to a computer screen size. In addition to this, the tool can check that the spacing between letters and lines has been defined with styling rules allowing users with cognitive disabilities to customise the text size, for example, to enable a faster or slower reading speed.

3.2 Tailored Results View

The web interface report of the validation results has an overview on the top of the page and then a double view in order to show the results in a way that can be tailored for different types of users.
As regards the overview (see Figure 2), we have designed it as divided into three main sections: one showing the evaluation summary, namely all the validation settings (URL, selected guideline, selected level of conformance, and the selected web page version related to a specific device); a second section composed of three boxes reporting the number of evaluated checkpoints with results "error", "warning", and "success" respectively; a third one showing the accessibility percentage, namely a measure indicating how much a resource is accessible in terms of number of checkpoints successfully evaluated over the total number of evaluated checkpoints.

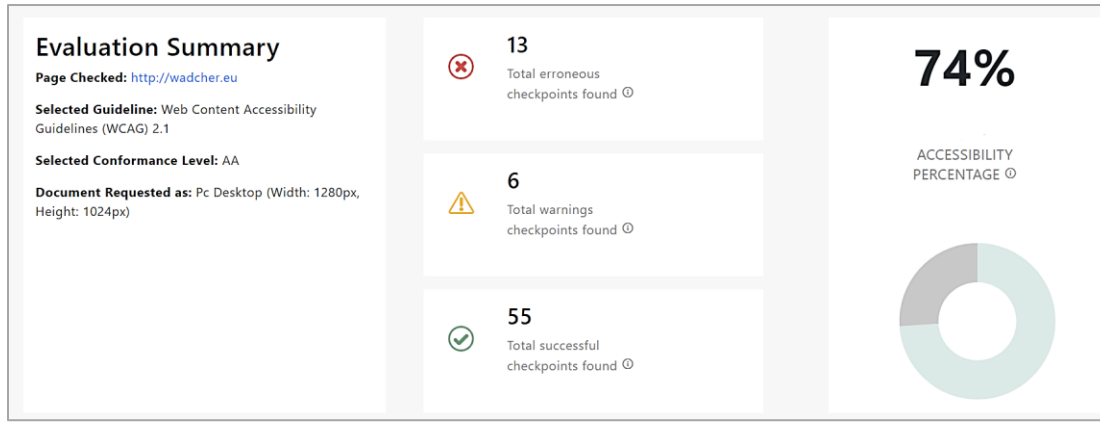


Figure 2. Tool Results Overview

In order to obtain tailored results presentations, we provide different views (which can be selected through an associated tab) for two main types of users:

- Web developers, interested in producing web pages which are compliant with a given accessibility guidelines' set, with expertise in web programming, and who know how to read the source code of a web page and modify it;
- End users (such as web commissioners) without web programming knowledge, interested in checking whether the web pages commissioned are compliant with a given accessibility guidelines' set.

In the first case, we adopted a source code view. In such a view, called "web developer" view, the source code of the web page evaluated is reported, and each error or warning is indicated just above the code line which generated it (see Figure 3 for an example). Such errors and warnings are highlighted with the colour and the icons used to indicate them in the results overview: red for the errors, yellow for the warning issues. Moreover, they are generated as links that point at the W3C documentation for each violated technique.

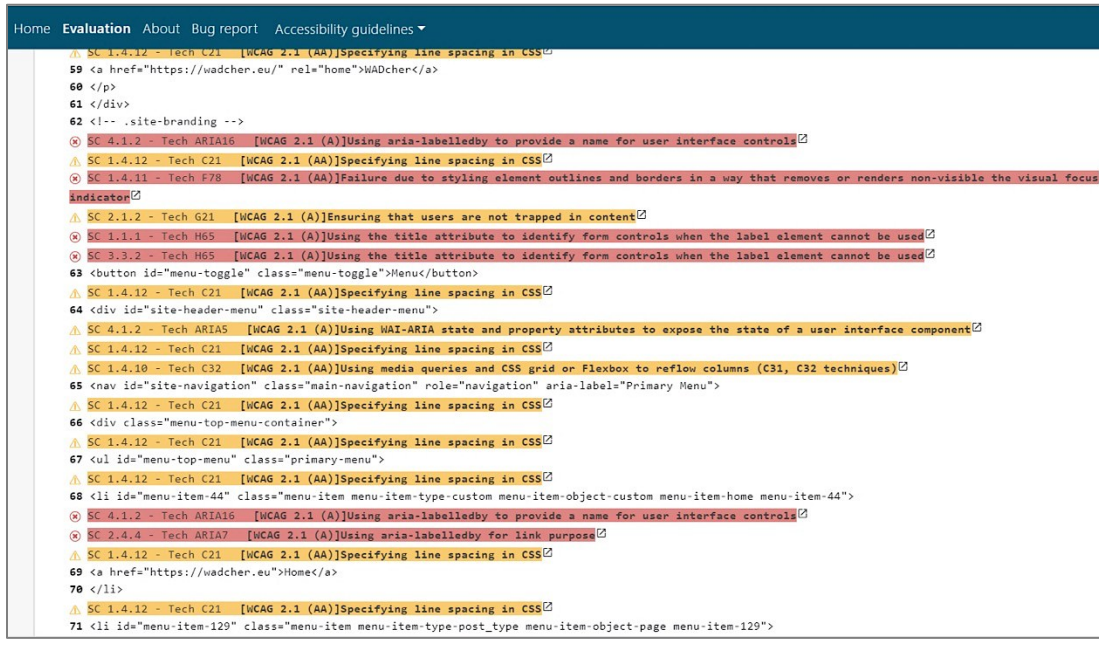
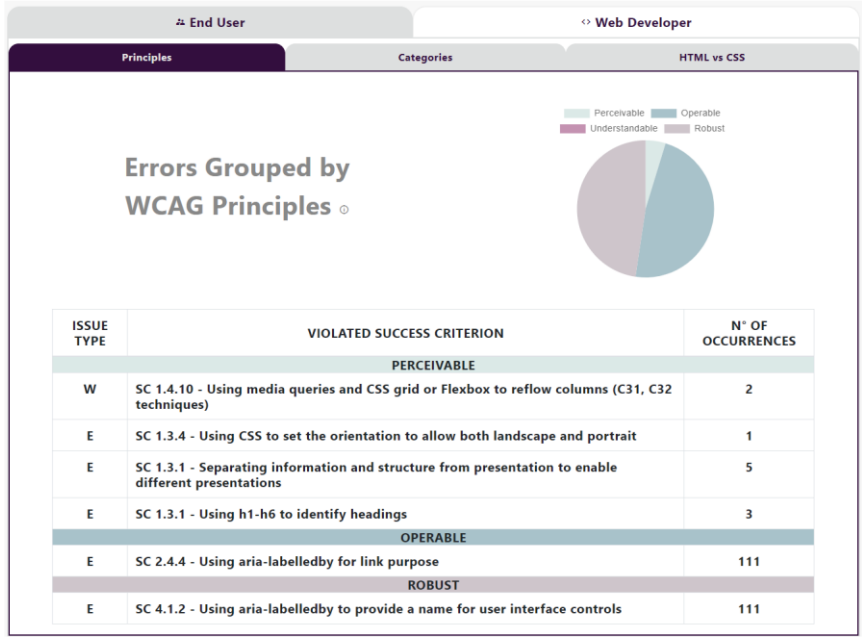


Figure 3. Detail on an example of report in the web developer view

The second view is in turn divided into three different parts according to the way errors and the warnings are grouped. Specifically, we devise three different groupings according to:

- the violated WCAG principles (i.e. Perceivable, Operable, Understandable, and Robust): view that we named "Principles";
- the type of content or element where the problem occurred: aria, colour, content, form, HTML, images, links, media, meta, responsive, tables: view named "Categories";
- the type of code where the problem occurred (i.e. CSS or HTML): view named "HTML vs CSS".

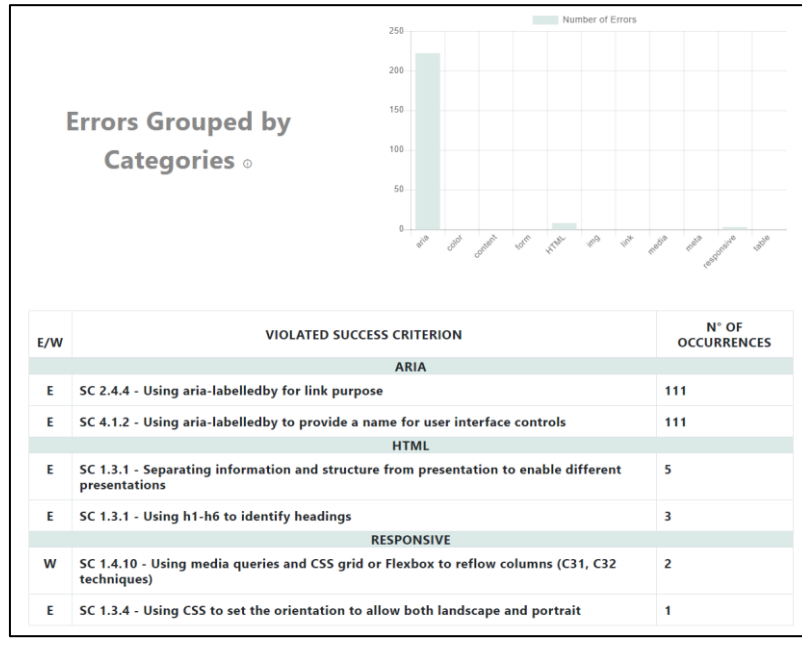
All the end-user views provide a graphical representation of the number of errors and warnings through charts. In particular, the "Principles" and the "HTML vs CSS" views show a pie chart (see an example in Figure 4), while the "Categories" view shows a bar chart with a bar for each content type (see an example in Figure 5).



ISSUE TYPE	VIOLATED SUCCESS CRITERION	N° OF OCCURRENCES
	PERCEIVABLE	
W	SC 1.4.10 - Using media queries and CSS grid or Flexbox to reflow columns (C31, C32 techniques)	2
E	SC 1.3.4 - Using CSS to set the orientation to allow both landscape and portrait	1
E	SC 1.3.1 - Separating information and structure from presentation to enable different presentations	5
E	SC 1.3.1 - Using h1-h6 to identify headings	3
	OPERABLE	
E	SC 2.4.4 - Using aria-labelledby for link purpose	111
	ROBUST	
E	SC 4.1.2 - Using aria-labelledby to provide a name for user interface controls	111

Figure 4. Example of report according to principles violated

Moreover, the detail about the errors and warnings are presented in a table (present in all the end-user views), reporting if the accessibility problem is an error or a warning, the name of the technique violated, and the number of occurrences. All the table's lines are grouped according to the view type (e.g. in the "Principles" view, all the accessibility problem related to the perceivable principle are under the line "perceivable").



E/W	VIOLATED SUCCESS CRITERION	N° OF OCCURRENCES
	ARIA	
E	SC 2.4.4 - Using aria-labelledby for link purpose	111
E	SC 4.1.2 - Using aria-labelledby to provide a name for user interface controls	111
	HTML	
E	SC 1.3.1 - Separating information and structure from presentation to enable different presentations	5
E	SC 1.3.1 - Using h1-h6 to identify headings	3
	RESPONSIVE	
W	SC 1.4.10 - Using media queries and CSS grid or Flexbox to reflow columns (C31, C32 techniques)	2
E	SC 1.3.4 - Using CSS to set the orientation to allow both landscape and portrait	1

Figure 5. Example of report according to elements' categories

3.3 Multipage Validation

As already mentioned, users who access the validation tool can choose if they want to validate a single web page or multiple pages. In the case of multipage validation, users need to register by inserting name, surname, and e-mail. The tool uses these data to store the validation results and send them via e-mail.
To trigger a multipage validation, users need to set the base URL of the website they want to validate, the accessibility settings (i.e. guidelines and level of conformance), and the crawling configuration settings:

- the maximum number of pages to fetch by the crawler, and
- the crawl depth, namely the depth with which the crawler carries out the analysis of the web pages to fetch starting from the base URL.

The results of the validation are sent to users asynchronously through e-mail as a PDF report, that can be stored locally by users in order to keep track of the accessibility trend and performance of their web resources.
The PDF report provides a glossary of the terms that will be used in the report itself, and an evaluation overview reporting all the validation data (i.e. the base URL, the crawling settings, the number of pages evaluated and the evaluation date and time) and the overall accessibility percentage. For both errors and warnings, it reports the total number of checkpoints with erroneous/warning results, the number of occurrences, and the average number per page. Moreover, the report presents a rank of the most erroneous pages of the website, according to the number of errors' occurrence.
Then, for each evaluated web page the report presents errors and warnings with the web interface end-user view presented in Section 3.2, which is by grouping errors according to principles, categories and code type, using charts and tables. We do not present in the report the source code view being more difficult to read from a PDF document, and we invite users to use the single validation through the web interface as support to correct the accessibility issues via the source code view of a specific webpage.
Furthermore, we devise the proposed tool as an accessibility observatory through which it is possible to check previous validations and improvements or regression of the monitored websites. For this purpose, users once logged in, can access a private results page where all their previous validations are presented (see Figure 6).

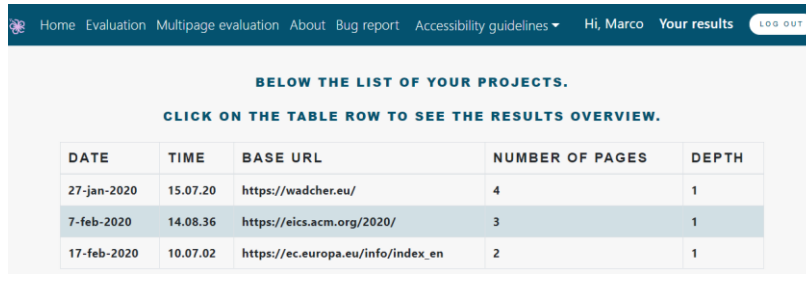


Figure 6. List of previous validation

As shown in Figure 7, for each of these projects, the tool interface presents an evaluation overview with base URL, number of pages, crawl depth and accessibility percentage; an overview of errors and warnings found in the evaluated website (as the one presented in the PDF report); and a rank of the most erroneous pages of the evaluated website (as the one presented in the PDF report).

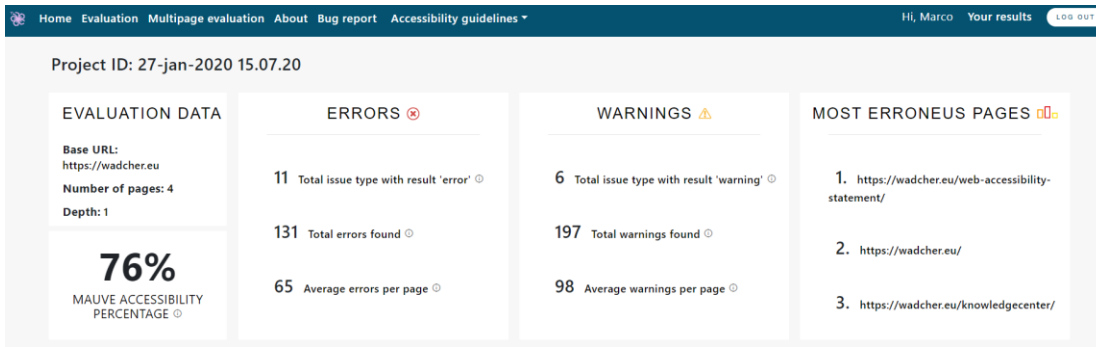


Figure 7. Result overview for multipage validation

4 TECHNICAL VALIDATION

In order to carry out a first test of our environment validation, we performed a comparison between our tool and a second well-known validation tool in terms of validity, completeness and consistency in supporting the conformance of web sites with respect to the WCAG 2.1 guidelines.
The comparison tool has been selected among the currently available software programs or online services that automatically check the accessibility of web pages. In order to make the comparison more consistent, we have selected a tool which – similarly to our validator – is able to support WCAG 2.1 and with open access. Furthermore, we restricted the choice to:

- Tools which can be used easily by every kind of users (e.g. tools which do not require a long and complex installation);
- Tools that require specific software installations and programming knowledge;
- Tools which do not focus on a single accessibility aspect (e.g. tools analysing exclusively images or the colour contrast).

The tool selected is the online version of the WAVE tool, a validator belonging to a suite of instruments that facilitate the web accessibility evaluation; the chosen instrument is able to validate a web page by entering its URL. For each evaluated web page WAVE shows the errors in a code-oriented view, where errors are marked with an icon on the code row which generated them. Moreover, the tool is able to provide a visual representation of the accessibility aspects within the page, by marking with icons not only the accessibility errors, but also the structural elements in the page, and the ARIA elements. In order to help the navigation among the errors, WAVE provides a side menu showing all the marked aspects on the page and their details.

For performing the comparison, we have selected a set of twelve well-known websites in application domains with relevance on users' daily life (there are two websites for each domain): e-commerce, public body, cultural heritage, health, travelling, and education. For each of these websites, four pages have been selected belonging to the following criteria: home page, a page with forms, a page with a lot of textual contents, and a page with many links. Appendix I lists the websites used for the validation.
The webpages have been validated against the WCAG 2.1 and with respect to the AA conformance level – which the WAD directive recommends as a requirement [8] – with both validators.

To carry out the analysis we followed the approach proposed by Brajnik [7]**Errore. L'origine r iferimento non è stata trovata.**, and also used in [2]: each selected webpage is validated through the two tools and manually inspected by a human accessibility expert to determine the validity and the completeness of the tools' results. In particular, each issue can be categorized as:

- True positive (TP). An issue generated by the validator is classified as TP if, after inspection, the accessibility expert considers it relevant and correct.
- False positive (FP). An issue generated by the validator is classified as FP if, after inspection, the accessibility expert considers it wrong or irrelevant.
- False negative (FN). An issue is considered FN if the validator is not able to detect it and the accessibility expert considers it relevant and correct.

We call *Validity* a measure which indicates how much the tool detects accessibility issues correctly, namely how many of the accessibility errors detected by the tool are TP. We compute such a measure as follows:

$$Validity = \frac{TP}{TP + FP}$$

We call *Completeness* a measure which indicates how much the tool detects the actual accessibility issues, namely how many TP errors are detected by the tool among all the real accessibility errors. We compute such a measure as follows:

$$Completeness = \frac{TP}{TP + FN}$$

Moreover, in order to analyse whether the tools provide consistent results, we consider the data collected through the validation in order to find which Success Criteria is most violated according to the results of both tools.

The test has been performed during January 2020 using the latest available version of both tools. In this test, we found that MAUVE++ provided more valid and complete results than WAVE. In fact, considering the results of all the evaluated webpages, the mean values of Validity and Completeness are:

	Validity	**Completeness**
MAUVE++	90%	93.2%
WAVE	62.9%	12.9%

We found that there were no particular differences in the tools' performances based on the type of the evaluated web page. As can be seen from the results shown in Table 1, which refer to the evaluation of the four web pages of the IKEA website, the page structure does not affect the overall results in terms of completeness and validity, and we have observed the same for both tools.

IKEA		**Tot N of detected Errors**	**FP**	**FN**	**TP**
Home page	MAUVE++	803	30	0	773
	WAVE	1	1	773	0
Page with links	MAUVE++	812	20	1	792
	WAVE	1	1	793	0
Page with textual content	MAUVE++	757	12	1	745
	WAVE	1	1	747	0
Page with forms	MAUVE++	2	1	742	1
	WAVE	767	21	1	741

Table 1. Evaluation results of the IKEA website's selected pages

The mean values of Validity and Completeness are given in Table 2 and Table 3 based on the application domains of the selected web pages, which are eight for each domain, in total.

Category	Validity	
	MAUVE++	WAVE
E-commerce	96.4	60
Public Body	87.3	50
Cultural Heritage	89	71.2
Health	88.4	59.9
Travelling	92	65.6
Education	89.6	70.8

Table 2. Validity mean values (%) of the two tools, grouped by domain

Category	Completeness	
	MAUVE++	WAVE
E-commerce	98.9	50
Public Body	98.7	0.2
Cultural Heritage	98.7	3.4
Health	90.4	9
Travelling	87.5	13.5
Education	97	1.3

Table 3. Completeness mean values (%) of the two tools, grouped by domain

The reported mean values also show that MAUVE++ was slightly more complete than valid, whereas WAVE, on the contrary, was more valid than complete; moreover, there was a significant difference in terms of completeness among the two tools.

WAVE performed the evaluation with the lowest value of Completeness (0.1%) in the evaluation of the second website in the Public Body domain. This page, in fact, contained a lot of links for which WAVE did not detect the violations of the SC 4.1.2 (mainly for missing aria-label and aria-labelledby attributes). The lowest value of Completeness observed in the proposed tool - 75% -, regards the evaluation of the second website in the Travelling domain, which had a lot of problems in the colour contrast between textual content and the background, which MAUVE++ did not detect.
As regards the Validity measure, the lowest recorded values were:

- 20% for WAVE, yielded by the evaluation of the first website in the E-commerce domain; here WAVE generated false positives in detecting colour contrast errors;
- 77.3% for the proposed tool, registered in the evaluation of the second website in the Public body domain; here MAUVE++ generated false positives in particular in detecting colour contrast errors in form elements that can have focus.

Finally, regarding the consistency, as stated above we looked at the Success Criteria that were most often violated according to the two tools. We first looked at the number of Success Criteria of WCAG 2.1 with AA level of conformance, 50 in total, that the two tools were able to check. MAUVE++ performed the checking of 33 success criteria over 50, whereas WAVE 25. Despite this difference in the Success Criteria coverage, we found that they were consistent in assessing that most accessibility barriers in the

evaluated websites regarded the missing ARIA attributes and labels in links and UI elements with link purpose.
Going deeper, in the results of MAUVE++, the most detected TP issues were related to Success Criterion 4.1.2 (Name, Role, Value)[19] with a total of 5995 occurrences in the tests performed; Success Criterion 4.1.2 is followed by Success Criterion 1.4.10 (433 occurrences), and Success Criterion 2.4.4 (113 occurrences).
In the results of WAVE, the most detected TP issues regarded Success Criterion 2.4.4 (Link Purpose, In Context)[20] with a total of 188 occurrences, followed by Success Criterion 1.4.3 (65 occurrences).
It is important to highlight that the Success Criterion 1.4.3 (Contrast, Minimum)[21], which is the second most detected violation by WAVE, was not accounted for amongst the most detected violations by MAUVE++. Moreover, MAUVE++ was less complete in detecting issues related to Success Criterion 1.4.3, while we note that WAVE was less complete in detecting issues related to Success Criterion 4.1.2, which never appeared in the WAVE evaluation results.

5 USABILITY TEST

In order to evaluate the usability of MAUVE++ with users, we conducted a study with 19 people. The tool has been designed for different audiences, hence we recruited people belonging to one of the two categories of envisioned users: "end-users" and "web developers".
The main objective of the study is to evaluate the usability of the tool, in particular of the two different ways to present validation results to users interested in the accessibility of webpages. We directly observed users while performing representative tasks, and collected their qualitative feedback with a post-study questionnaire.
The study design reproduces a realistic interaction scenario, where users access the validation environment while sitting at their working station to check if the webpage they are working on contains accessibility barriers.
To recruit suitable participants, we sent an invitation email through the mailing list of our research community, where we could find people who are both content authors/commissioners and developers interested in assessing the accessibility level of their web resources. We prepared a booklet showing and explaining the new features and we sent it to people who answered the invitation mail a few days before the date agreed for participating in the study.
The study took place in a laboratory, in the presence of two test moderators who guided and observed the participants during task execution. We set up a PC giving participants access to the tool. Each participant was assigned to a role corresponding to his/her expertise, then was informed about how the study was organised. The moderators observed and annotated behaviour and comments during tasks' execution, paying attention to problems and hesitations, and recorded completion times as well.

5.1 Study setup

The study started with a familiarisation phase: participants were allowed to evaluate a web page of their choice, browsing the results without constraints for about five minutes. In this phase, they could ask questions to the moderators if something about the information presented or the procedure was not clear.

[19] https://www.w3.org/WAI/WCAG21/Understanding/name-role-value
[20] https://www.w3.org/WAI/WCAG21/Understanding/link-purpose-in-context
[21] https://www.w3.org/WAI/WCAG21/Understanding/contrast-minimum

The core phase of the study consisted of the completion of 4 tasks:

- Task 1: validate the given web page through URL, selecting WCAG 2.1 guidelines and AAA conformance level;
- Task 2: view and analyse the overview of the evaluation results:
  - Sub-task 1: Indicate the accessibility percentage of the page;
  - Sub-task 2: Indicate the number of checkpoints evaluated with success.
- Task 3: view and analyse the detailed report of the evaluation results;
- Task 4: search for information regarding the tool.

The four tasks were identical for both user roles, except Task 3, which has different sub-tasks for end-users and web developers.
In particular, the sub-tasks for the "end-user" were:

- Indicate the number of errors associated with the operable principle;
- Indicate the category with the highest number of issues found;
- Indicate how many issues of this category are warnings;
- Indicate whether more problems were found in the HTML or in the CSS code;
- Indicate the most recurrent error in the HTML code.

The sub-tasks for the "web developer", instead, were:

- Point to a code line causing both a warning and an error;
- Find a warning with A level of conformance;
- Follow the link to the external documentation regarding the detected warnings.

At the end of the core phase, we asked participants to complete a questionnaire composed of three different sections. The first one regarded personal information, professional background, knowledge in the accessibility domain, previous experience with automatic accessibility evaluation tools. The second section aimed to gather feedback and suggestions about clearness, completeness, easiness of navigation of the information presented in the detailed report. Finally, in the third section, a SUS (System Usability Scale) [22] and an NPS (Net Promoter Score)[23] questionnaire were provided and filled in.

5.2 Participants

Nineteen people participated in the study: 8 users (6 females, 2 males; mean age: 39.7 years, σ 14.2) participated as end-users and 11 (3 females, 8 males; mean age: 32.2, σ 8.9) as web developers. As regards their education, eight participants have a master's degree, six have a PhD, five have a bachelor's degree. Currently, they are all working in the academic research field: they are researchers (8), technicians (4), PhD students (3), master's degree students (4).
Web developers know the following programming languages and frameworks, ordered by the most known: HTML5, JavaScript, CSS3, jQuery, PHP, Java; none of them know Angular.js and React.js. Only six developers say that they consider accessibility during their work, in particular paying attention to the fundamental aspects: colour contrast, avoiding fixed sizes, fluid layout.
On the end-users' side, they evaluated their web programming skills on a scale from 1 (no skills) to 5 (I know HTML, CSS, JavaScript, PHP, and other languages), with a mean of 2.65 (median: 2.5). Both groups self-evaluated their level of knowledge in the web accessibility domain on a scale from 1 (novice) to 5 (expert), with a mean of 2.63 for web developers (median: 2) and 2.5 for end-users (median: 2.5); mostly, they have not been trained in web accessibility (8), being self-taught, or they attended a university course (6). Eleven participants are familiar with the WCAG 2.1 guidelines; 8 people do not

[22] https://www.usability.gov/how-to-and-tools/methods/system-usability-scale.html
[23] https://en.wikipedia.org/wiki/Net_Promoter

know any accessibility standard or guideline. As regards other tools or plugins, 9 participants answered that they have used various ones, such as MAUVE (7), WAVE (2), AChecker (3), Siteimprove (1), W3C validators (2), EIII (1), Bobby (1).

5.3 Quantitative and qualitative results

The mean total time (in mm:ss.s) to complete the four tasks was 01:56.5 for web developers and 03:05.2 for end-users. Among the tasks, the one with the longest completion time was Task 3 for both groups, since it was made up of multiple sub-tasks, and, as we will report in the following, participants encountered difficulties with some of them. The box plots in Figure 8 and Figure 9 show the completion times for each task for the two types of users.

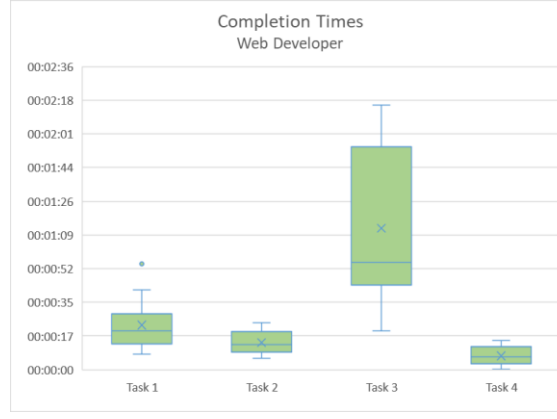


Figure 8. Task completion times of web developers

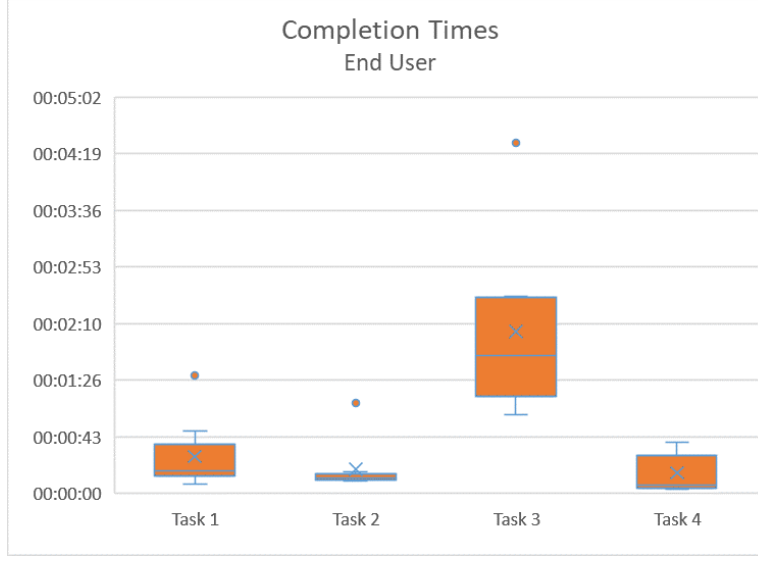


Figure 9. Task completion times of end-users

By observing users, we noticed whether the tasks were completed with success, with problems, or failed. We used the collected data to compute the *Success rate* metric, a useful indicator for the percentage of tasks that users completed correctly. It is computed by considering the sum of successfully completed tasks, plus the tasks completed with problems weighted half a point, divided by the total number of completion attempts (the number of tasks*the number of participants). When measuring task success rate, it is common practise to weight 0.5 tasks that have been performed partially correctly (to distinguish them from those completely unsuccessful and those completely correctly performed) [21].

$$Success\ rate\ = \frac{taks\ with\ + (tasks\ with\ problems\ \times 0.5)}{number\ of\ tasks\ \times number\ of\ participants}$$

We thus obtained the following results:

- 92% for the "web developer" group;
- 82% for the "end-user" group.

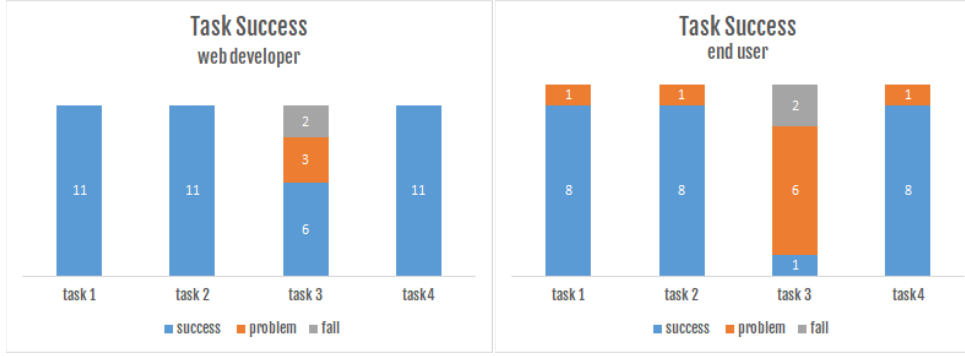


Figure 10. Task successes of the two users' groups

The main difficulties we observed in web developers occurred when completing Task 3, Subtask 1, regarding the identification of the precise code line where the issue occurred. In five cases, the first answer was wrong: for example, if the correct line number was 75 (a line with both an error and warning), they answered 74. This is insightful for better designing this aspect, in order to make the grouping of code line and corresponding error/warning messages more perceivable.

End users, instead, found difficulties mainly with the naming of the tabs, and some accessibility terms, such as *Principle*. In fact, we observed that the tab labels chosen were not particularly comprehensible for some users: *Principles* and *Categories* are often considered synonymous, whereas the groupings under the tab *Tags* were named *Categories*, and this caused some problems in completing Subtask 2. For example, one user, asked to complete Subtask 4 (Are there more problems in the HTML or in the CSS code?), became confused and said that she expected to find the source code under the "HTML vs CSS" tab.

Participants evaluated the clarity and usefulness of the main features of the tool on a scale from 1 to 5. The overview section was evaluated by both groups (mean: 4.81 web developers, 4.65 end-users).

Then, the web developers also rated: the clarity of the distinction between error and warning (mean: 4.45) and the reference to the code line (mean: 3.81, positive, despite some difficulties observed, as reported above), the issue description (mean: 3.9), the link to the official documentation explaining the evaluated checkpoint and techniques/failures (mean: 4.18).

End users were asked to evaluate the features of the end-user presentation of the results, namely: the meaning of the three groupings we chose for presenting the accessibility issues in the sub-tabs (mean:

4.62) and their labels (mean: 4,25), the utility of the summarising graphs (mean: 5), and finally the clearness of the evaluation information as they are presented in the tables (mean: 4.12).
The collected suggestions on how to improve the tool regarded mainly the way in which the detected barrier is referenced to the corresponding code line, on the web developer side. End users suggested improving the readability of the information presented in the tables by adding the code line where the issue occurred, making the difference between an error and a warning clearer, for example giving a different background colour to table lines based on the evaluation outcome.
We also elicited feedback on desired features and information that they would find useful or that they would need, so we provided a specific question for this purpose. We received ideas in particular from web developers, who mentioned the possibility of also seeing the CSS code with highlighted lines (currently, the problems with CSS properties are shown in the corresponding HTML element), and solutions (e.g. a tooltip or a to-do list) to see a brief and clear explanation on how to solve the issues or examples of correct coding.
Finally, as regards the NPS (Net Promoter Score), 57.9% of participants were promoters, 31.6% passives, and 10.5% detractors of the tool. In order to assess participants' satisfaction and perceived usability of the tool, we administered the SUS questionnaire, which gave a positive result. Just one participant (a "web developer") gave a global score of 65, which is under the minimum value of 68, which indicates a sufficient evaluation. Instead, the mean rating was slightly different among the two groups: 84.06 for the end-users and 85.23 for the web developers; the mean global value was 84.74.

6 CONCLUSIONS AND FUTURE WORK

We have presented a novel tool that extends previous experiences in automatic accessibility validation by providing a novel solution for a number of issues emerged in recent years in order to make automatic support for accessibility evaluation more effective. While some tools exist that address some of such issues, in our case we are able to provide an open and flexible integrated solution. Indeed, the presented tool is able to support both single page and multi-page validation, generating reports in various formats (web, PDF, EARL) and for different audiences (web developers, end-users), and validating the accessibility according to WCAG 2.1 guidelines. In this way, it can provide accessibility checking for mobile-oriented web interactions, which are dominant at this time. It is a responsive web application so that it can also be accessed through a mobile device while editing the code of a page on the desktop.
We also report on a technical validation that has shown positive results with respect to other tools currently available, while the user test indicated that it can be used easily by people of the targeted audiences.
Future work will be dedicated to integrating this tool with a content management system in order to exploit its functionalities while creating web content, introducing some support for more structured metrics that take more into account the differences in impact of the diverse criteria, and further adding monitoring features in order to let stakeholders observe the trends of the accessibility compliance over time, and be able to take action in order to address the detected issues. We also want to extend the set of tailored reports, for example creating reports that focus on guidelines that are more relevant for specific disabilities.

The tool is publicly available for use at https://mauve.isti.cnr.it/.

REFERENCES

[1] Abascal, J., Arrue, M., & Valencia, X. (2019). Tools for web accessibility evaluation. *Web Accessibility* (pp. 479-503). London: Springer.

[2] Abduganiev, S. G. (2017). Towards automated web accessibility evaluation: a comparative study. *Int. J. Inform. Technol. Comput. Sci, 9*(9), 18-44.

[3] Arrue, M., Vigo, M., & Abascal, J. (2008). Including heterogeneous Web accessibility guidelines in the development process. *IFIP International Conference on Engineering for Human-Computer Interaction* (pp. 620-637). Berlin: Springer.

[4] Ballantyne, M., Jha, A., Jacobsen, A., Hawker, J. S., & El-Glaly, Y. N. (2018). Study of Accessibility Guidelines of Mobile Applications. *17th International Conference on Mobile and Ubiquitous Multimedia* (pp. 305-315). ACM.

[5] Beirekdar, A., Vanderdonckt, J., & Noirhomme-Fraiture, M. (2002). Kwaresmi–Knowledge-based Web Automated Evaluation with REconfigurable guidelineS optiMIzation. (Springer, Ed.) *DSV-IS, 2545*, 362-376.

[6] Beirekdar A., Keita M., Noirhomme M., Randolet F., Vanderdonckt J., Mariage C. (2005) Flexible Reporting for Automated Usability and Accessibility Evaluation of Web Sites. In: Costabile M.F., Paternò F. (eds) Human-Computer Interaction - INTERACT 2005. INTERACT 2005. Lecture Notes in Computer Science, vol 3585. Springer, Berlin, Heidelberg

[7] Brajnik, G. (2004). Comparing accessibility evaluation tools: a method for tool effectiveness. *Universal access in the information society, 3*(3-4), 252-263.

[8] Brajnik, G., & Vigo, M. (2019). Automatic Web Accessibility Metrics. Where we were and where we went. (Springer, Ed.) *Web Accessibility*, 505-521.

[9] Consortium, W. W. (2018). *Web content accessibility guidelines (WCAG) 2.1.* Retrieved from https://www.w3.org/TR/WCAG21/

[10] EU Commission. (2016, October 26). *Directive (EU) 2016/2102 of the European Parliament and of the Council.* Retrieved from https://eur-lex.europa.eu: https://eur-lex.europa.eu/eli/dir/2016/2102/oj

[11] Fernandes, N., Kaklanis, N., Votis, K., Tzovaras, D., & Carriço, L. (2014). An analysis of personalized web accessibility. *Proceedings of the 11th Web for All Conference* (p. 19). ACM.

[12] Fuertes, J. L., González, R., Gutiérrez, E., & Martínez, L. (2009). Hera-FFX: a Firefox add-on for semi-automatic web accessibility evaluation. *Proceedings of the 2009 International Cross-Disciplinary Conference on Web Accessibililty (W4A)* (pp. 26-35). ACM.

[13] Gay, G., & Li, C. Q. (2010). AChecker: open, interactive, customizable, web accessibility checking. *Proceedings of the 2010 International Cross Disciplinary Conference on Web Accessibility (W4A)* (p. 23). ACM.

[14] Gulliksen, J., Von Axelson, H., Persson, H., & Göransson, H. (2010). Accessibility and public policy in Sweden. *Interactions, 17*(3), 26-29.

[15] Ivory, M. Y., & Hearst, M. A. (2001, December). State of the Rt in Automating Usability Evaluation of User Interfaces. *ACM Computing Surveys, 33*(4), 470-516.

[16] Lazar, J., & Olalere, A. (2011). Investigation of best practices for maintaining section 508 Compliance in US federal web sites. *International Conference on Universal Access in Human-Computer Interaction* (pp. 498-506). Berlin: Springer.

[17] Aliaksei Miniukovich, Michele Scaltritti, Simone Sulpizio, and Antonella De Angeli. 2019. Guideline-Based Evaluation of Web Readability. In Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19). Association for Computing Machinery, New York, NY, USA, Paper 508, 1–12. DOI:https://doi.org/10.1145/3290605.3300738

[18] Mirri, S., Muratori, L. A., & Salomoni, P. (2011). Monitoring accessibility: large scale evaluations at a geo political level. *The proceedings of the 13th international ACM SIGACCESS conference on Computers and accessibility* (pp. 163-170). New York: ACM.

[19] Lourdes Moreno, Rodrigo Alarcon, Isabel Segura-Bedmar, and Paloma Martínez. 2019. Lexical simplification approach to support the accessibility guidelines. In Proceedings of the XX International Conference on Human Computer Interaction (Interaccion '19). Association for Computing Machinery, New York, NY, USA, Article 14, 1–4. DOI:https://doi.org/10.1145/3335595.3335651

[20] Mucha, J., Snaprud, M., & Nietzio, A. (2016). Web page clustering for more efficient website accessibility evaluations. *International Conference on Computers Helping People with Special Needs* (pp. 259-266). Springer.

[21] Jacob Nielsen, Success Rate: The Simplest Usability Metric, https://www.nngroup.com/articles/success-rate-the-simplest-usability-metric/

[22] Nietzio, A., Eibegger, M., Goodwin, M., & Snaprud, M. (2011). Towards a score function for WCAG 2.0 benchmarking. *Proceedings of W3C Online Symposium on Website Accessibility Metrics.* Retrieved from https://www.w3.org/WAI/RD/2011/metrics/paper11

[23] Paternò, F., & Schiavone, A. G. (2015). The role of tool support in public policies and accessibility. *Interactions, 22*(3), 60-63.

[24] Power, C., Freire, A., Petrie, H., & Swallow, D. (2012). Guidelines are only half of the story: accessibility problems encountered by blind users on the web. *Proceedings of the SIGCHI conference on human factors in computing systems* (pp. 433-442). ACM.

[25] Schiavone, A. G., & Paternò, F. (2015). An extensible environment for guideline-based accessibility evaluation of dynamic Web applications. *Universal access in the information society, 14*(1), 111-132.

APPENDIX I

Websites used for validation

CATEGORY	WEBSITE	PAGE	URL
E-commerce	Ikea	Home	https://www.ikea.com/ch/en/
		Links	https://www.ikea.com/ch/en/customer-service/faq/
		Contents	https://www.ikea.com/ch/en/customer-service/services/rent-a-van-pubf869a3b1
		Forms	https://www.ikea.com/ch/en/customer-service/contact-us/
	Apple	Home	https://www.apple.com/
		Links	https://support.apple.com/
		Contents	https://www.apple.com/apple-watch-series-5/health/
		Forms	https://support.apple.com/en-us/HT207581
Public body	UK Government	Home	https://www.gov.uk/
		Links	https://www.gov.uk/browse/education

		Contents	https://www.gov.uk/appeal-exam-result
		Forms	https://logon.slc.co.uk/cas/login
	European Commission	Home	https://ec.europa.eu/info/index_en
		Links	https://ec.europa.eu/info/topics_en
		Contents	https://ec.europa.eu/info/priorities/europe-fit-digital-age_en
		Forms	https://ec.europa.eu/info/publications_en
Culture	Uffizi Museum	Home	https://www.uffizi.it/en
		Links	https://www.uffizi.it/en/artworks
		Contents	https://www.uffizi.it/en/artworks/woman-with-a-veil
		Forms	https://www.uffizi.it/en/contacts
	Library of Trinity College	Home	https://www.tcd.ie/library/
		Links	https://www.tcd.ie/library/collections/
		Contents	https://www.tcd.ie/library/about/history.php
		Forms	https://library.catalogue.tcd.ie/iii/cas/login?service=https%3A%2F%2Fstella.catalogue.tcd.ie%2Fiii%2Fencore%2Fj_acegi_cas_security_check%3Bjsessionid%3D2755A9EB2A96B89614C71E6ECDEC6087
Health	French National Medicine Academy	Home	http://www.fam.fr/en/
		Links	http://www.fam.fr/en/news/
		Contents	http://www.fam.fr/en/the-foundation/
		Forms	http://www.fam.fr/en/our-actions/
	Lilly	Home	https://www.lilly.com/
		Links	https://www.lilly.com/products
		Contents	https://www.lilly.com/diabetessolutioncenter
		Forms	https://investor.lilly.com/stock-information/investment-calculator
Travel	SNCF	Home	https://www.sncf.com/en
		Links	https://www.sncf.com/en#menu+our-passenger-offer
		Contents	https://www.sncf.com/en/sncf-around-the-world
		Forms	https://www.sncf.com/en/booking-itinerary/itinerary
	Terravision	Home	https://www.terravision.eu/
		Links	https://www.terravision.eu/faq_company_information.html
		Contents	https://www.terravision.eu/group_transfer.html?noredirect=en_US
		Forms	https://book.terravision.eu/login
Education	University of Cagliari	Home	https://www.unica.it/unica/en/homepage.page
		Links	https://www.unica.it/unica/en/studenti_s01.page
		Contents	https://www.unica.it/unica/en/futuri_studenti_s01_ss01.page
		Forms	https://www.unica.it/unica/en/ateneo_s08_ss03_sss03.page

	Harvard University	Home	https://www.harvard.edu/
		Links	https://www.harvard.edu/about-harvard/frequently-asked-questions
		Contents	https://www.harvard.edu/about-harvard/harvard-glance/history
		Forms	https://www.harvard.edu/feedback?