Abstract
This paper presents our perspectives on the key challenges developing accessibility conformance tools. These challenges are influenced by the rapid evolution of the web ecosystem composed by web technologies (e.g., HTML5, CSS3, JavaScript), user agents (e.g., web browsers) and devices (e.g., smart phones, tablets, televisions), and by the usage of the aforementioned technologies by users with different requirements, preferences and context of use. We introduce these challenges from a tool developer perspective and present attempts to face such problems in a real development environment for a commercial tool. We describe as well related work from different EU-funded research projects and how such knowledge is fed to the imergo® Web Compliance Suite.

Keywords: web accessibility, compliance, service-oriented architecture, web testing, web 2.0

1. Introduction — Challenges for developing accessibility evaluation tools

People with disabilities and older persons are often excluded from using many online applications and services despite numerous efforts to improve their accessibility. The UN Convention on the Rights of Persons with Disabilities (UN CRPD) recognizes web accessibility as a fundamental right of the person and has been adopted by all the EU member states and by the EC itself. Given the importance of the web, already in 2002 the European Council adopted the Resolution 7087/02 which calls for adoption of the Web Content Accessibility Guidelines from the World Wide Web Consortium (W3C) by all EU Member States. In 2006, ministers from 34 European countries unanimously signed the Riga Declaration – “ICT for an inclusive society,” which sets concrete targets for the inclusion of people with disabilities, including the commitment to make all public web sites conform with the W3C accessibility recommendations by year 2010. The Web Content Accessibility Guidelines (WCAG) 2.0, which is also available as an ISO Standard (ISO/IEC 40500:2014), is the internationally recognized standard for web accessibility. However, recent studies carried out by the European Commission continue to show slow progress in the accessibility implementation.
for European web sites, thus excluding people with disabilities from access to education, employment, government services and many more critical aspects of the daily life. This trend is also observable in many countries all over the world.

The Web Content Accessibility Guidelines (WCAG) 2.0 recommendation has provided a stable reference framework for designing and evaluating accessible web sites and applications. In contrast to WCAG 1.0, which was HTML centric, WCAG 2.0 has been designed to be abstract enough to incorporate future technologies offering an enduring standard.

The next step had been to respond to the demand for a harmonized methodology that could be a basis for experts and organizations to evaluate and implement accessibility consistently. The Website Conformance Evaluation Methodology (WCAG-EM) developed under the WAI-ACT project has provided such a common guidance. However, WCAG-EM “describes a procedure to evaluate websites and includes considerations to guide evaluators and to promote good practice. It does not provide instructions for evaluating web content feature by feature, which is addressed by WCAG 2.0 success criteria.”

A big and still unsolved challenge is the harmonized technical guidance and evaluation of web content using a set of specific web technologies. In that direction, W3C has developed the Techniques and Failures supporting documents that are intended to continually respond to the technological advancements. When referring to technological advancements, we mean web technologies, devices, operating systems, user agents (web browsers) and assistive technologies (AT). Moreover the fact that nowadays the technologies are evolving so rapidly, their combination bring emerging characteristics/ issues and different ways of use (development patterns).

Despite these advancements, an obstacle to the successful application of web accessibility is related to the “Accessibility Support” concept as described in WCAG 2.0 and is strongly linked to the pace of technological change and development. In simple terms, it is impossible for web designers and developers to perform evaluation of their developments covering all possible combinations of assistive technologies, user agents, operating systems and technology features. The Accessibility Support concept has more or less replaced the “until user agent support…” statements usually met in WCAG 1.0 that would refer the reader to a supporting document. The problem with WCAG 1.0 was that this supporting reference document stayed out of date and in that way it failed to meet its expectations. The main reason was of course the tremendous speed in which technologies evolve, making impossible for W3C Working Groups to maintain the document. WCAG 2.0 has solved that problem by declaring abstract success criteria, but in practice the problem has been transferred to content and application authors (designers, developers). The authors now need to identify the accessibility support based on their target audience. This means that they would need to do all the user evaluations necessary to determine which ways of using the applied web technologies are actually supported by given versions of assistive technologies and user agents. This is an extremely time-consuming and expensive process. This fact and also given the limited resources of W3C and its publication process makes the situation unworkable when using the traditional top-down approaches. The EC-funded WAI-ACT project allowed W3C to initiate a parallel approach of crowd-sourcing this data (described in section 2).

The same challenge from the perspective of the accessibility evaluator is to develop and sustain technical guidance (rules) for the evaluation of web content given a specific set of technologies. The official W3C answer to this challenge are the Common Failures from the WCAG 2.0 techniques. However, it seems that the same scalability and sustainability problem appear again here and this is the reason different attempts around the world have taken place to develop such technical guidance (rules). Among them, we can cite the Automated WCAG Monitoring Community Group, OpenAjax Web Content Accessibility Guidelines 2.0 Ruleset, Quail Accessibility tests and Google Chrome Accessibility Developer Tools tests. Looking into these attempts we can identify interesting but different rules, software frameworks and APIs. What they seem to share is a similar vision: having up-to-date technology specific rules for web accessibility evaluation. However, the most important observation is that all have also the same problems: after setting up the software, the APIs and populating the repositories with some testing rules, there is no appropriate “exit strategy” (the after-project strategy) that would maximize the chances for sustainability and scalability and they lack interoperability and reusability.

Section 2 of the paper presents a collaborative tool to address the issues described above from the perspective of creating a reliable set of tests that address accessibility compliance and section 3 of the paper introduces the work of the authors when developing a modern compliance tool that respond to different needs of web developers.
2. The Accessibility Support Database

The purpose of the Accessibility Support Database (axsDB) is to provide a public crowd-sourced repository of information on accessibility support. This includes information on support for accessibility in web browsers, assistive technologies, and web technologies. The repository implements a collaborative tool that facilitates public input and contribution, to continually expand the repository with updated and new information on accessibility support.

The vision of the Accessibility Support Database System can be summarized in terms of the following user stories:

(i) As a web developer (simple axsDB user) I would like to know in which combinations of web browsers and assistive technologies a specific WAI-ARIA property is supported, for a specific web technology such as HTML5, thus I can find out how suitable is such a property to cover adequately my target groups or if I need to find an alternative approach.

(ii) As a web accessibility consultant (simple axsDB user) I would like to know how consistently a specific WCAG 2.0 Technique is supported across most web browsers and/or assistive technologies, thus I can benefit by promoting that to my clients.

(iii) As a web accessibility evaluator (simple axsDB user) I would like to know if a specific WCAG 2.0 Technique is supported in particular combinations of web browsers and assistive technologies, thus I can benefit by providing appropriate feedback to my evaluation when the specific technique is being applied.

(iv) As a web browser and/or assistive technology developer (simple axsDB user) I would like to know if a specific WCAG 2.0 Technique is supported in a particular version of my products, thus I can benefit by either improving my products or by promoting such a feature to the marketing department.

(v) As an AT product vendor (simple axsDB user) I would like to be able to provide the testing data I have done for my product as a proof for the accessibility support of my product, thus I can benefit by promoting my features comparing to other competitive products.

(vi) As a WCAG 2.0 techniques contributor (privileged axsDB user) I would like to know how well my proposed technique covers the different browsers and assistive technologies, thus I can benefit by improving my techniques.

(vii) As a web accessibility researcher (simple axsDB user) I would like to know details about the support of a specific WAI-ARIA property, thus I can benefit by improving that through my research.

(viii) As a member of a W3C working group (simple axsDB user) I would like to be able to review test cases and flag them appropriately, thus I can benefit by ensuring their quality through the predefined quality assurance process.
(ix) As a test case tester (simple axsDB user) I would like to be able to save my different testing profiles and my preferences in general, thus I can benefit by not wasting time setting up my profiles and preferences every time I want to carry out new tests.

(x) As a W3C system administrator (privileged axsDB user) I would like to be able to manage axsDB user accounts and privileges through the existing LDAP infrastructure, thus I can benefit by not requiring any time to learn a new system and also ensure user account integrity and privacy.

(xi) As a tester (simple axsDB user) I would like to be able to “play” with axsDB without submitting any data, thus I can benefit from reaching a confidence level before I decide to expose my work to critiques.

The system follows a Service-Oriented Architecture (Fig. 1) to separate completely the back-end infrastructure from the User Interface (UI). The purpose of such an approach is twofold: 1) to provide the data through open APIs for reuse and repurpose by different stakeholders and 2) to allow anybody to be able to develop different UIs for potential different user groups and devices.

Designing both the user navigation but also the UI (Fig. 2) have proved to be challenging tasks given the structural complexity of the data, the diversity of the audience, the motivation that the application needs to provide to the user and the strong requirement for accessibility.

From the user stories, a required feature that comes out is to allow a user aiming to do testing to be able to do it “offline” until at least she can feel comfortable with the system and has the “courage” to submit test cases which means put her work under critique. Based on that, the system has to provide the full functionality to an anonymous user and at the same time give to the user the possibility at any moment to submit the testing results. Such a feature makes more probable that the next time the user will decide to register/login and do more systematic work offering important input data.

3. The imergo® Web Compliance Suite

The next challenge that the authors have addressed was to develop a compliance evaluation tool which can verify automatically (or semi-automatically) the tests and rules presented in the previous section, while keeping in mind a wide variety of requirements or features that an accessibility evaluation tool must integrate. Velasco & Abou-Znahm have classified these features from different perspectives: the resource to be evaluated (i.e., web content and its linked resources, which enable its rendering in the user agent), the evaluation requirements, the reporting customization capabilities of the tool and other tool usage characteristics like the integration into the development and editing workflow of the user.

Because of space constrains, we are not going into a detailed description of such features. We just highlight that some of them are critical in the ubiquitous modern web. For instance, some commercial and open source tools in the market still do not evaluate for accessibility compliance rendered web pages but the source code of an HTML page, which brings incorrect results to the developer. Another important aspect is the integration into the workflow of the developer, so the compliance test falls into it naturally as another part of the quality assurance procedure of the software.
The imergo® Web Compliance Suite is built upon a distributed and scalable Service-Oriented Architecture that allows its integration within many environments. It provides a usable, multilingual web-based interface aimed at optimizing the Web Compliance workflow. It consists of different components:

(i) An integrated state of the art Rendering Engine that allows the interpretation of JavaScript code and CSS rules, leading to the analysis of the web applications as seen by the end user. This allows, for example, the analysis of Single Page Applications, which make high use of client-side generated content.

(ii) A Crawler capable of automatically navigating complex web pages, retrieving their content through the Rendering Engine for analysis. It uses the rendered web page in order to identify its navigation states. Requests of the crawler can be fully customized, e.g., by changing the accepted content type to support content negotiation or by adding cookies or other session information to support authentication. Furthermore the crawler can be configured to automatically create a test sample of web pages for a given web site. For example, it is possible to find pages that contain specific types of content like forms, tables, images or any other custom defined criterion.

(iii) An Analyser Framework with more than 400 predefined rules to evaluate web pages against different compliance criteria, e.g., accessibility, markup validation, search engine optimization and corporate identity. Current accessibility evaluation rule sets include Web Content Accessibility Guidelines WCAG 2.0³ (levels A, AA and AAA, ISO/IEC 40500:2012) and BITV 2.0¹⁹ (German accessibility legislation, priorities 1 and 2). Current markup validation rule sets include XML, XHTML, HTML 4 and HTML5 validation. Current search engine optimization rule sets are based on the Google SEO Guidelines.²⁰ Additionally, the analyser framework allows to create custom rule sets out of these rules or to create own custom rules.

(iv) A Reporting Engine targeting a variety of audiences, from web developers to the company’s management compatible with state of art standards like the machine readable formats EARL²¹ and JSON or the more human readable formats HTML and PDF. The results of the automatic Web Compliance analysis can be easily integrated with results from the manual expert evaluation. Furthermore using EARL and JSON allows integration into issue management systems, bug tracking systems or other workflow support tools.

(v) A JSON-based RESTful Web Compliance API that allows the integration of the suite in any software, framework and tool for the web. Through this API it is possible to evaluate single web pages, complex web sites as well as DOM document fragments. The API has already been integrated in enterprise content management systems, mobile applications, integrated development environments and web applications.

(vi) An Analytics Engine, which allows the owners of the web site to monitor the web site based on the selected rules and make the recommended improvements. This engine saves, retrieves and filters the evaluation results from the Analyser Framework, but also integrates manual expert evaluation feedback. The Analytics Engine uses a highly performant and scalable persistence based upon Big Data infrastructure.

(vii) An User Interaction Simulation Engine which is capable of emulating user interactions on a web site. Simulation scripts are written in JavaScript.

4. Conclusions

The paper has presented a brief overview of the issues arising when evaluating the accessibility compliance of modern and ubiquitous web sites. Additionally, we introduced solutions that address these problems from different perspectives, while at the same time supporting both the web developer and the accessibility expert within their typical workflows.

Some of the results presented in the paper are still under development or in different user testing phases. These evaluations will be analysed and presented to the scientific community in the near future.

Acknowledgments

Parts of this work have been undertaken within the framework of the Inclusive Future-Internet Web Services (i2web) project, co-financed by the 7th Framework Programme of the European Commission (Grant Agreement 257623) and the WAI-ACT project, also co-financed by the 7th Framework Programme of the European Commission (Grant Agreement 287725).
Fig. 3. imergo® architecture overview.

References

11. Automated WCAG Monitoring Community Group. 2015. URL: https://www.w3.org/community/auto-wcag/.
13. QuaLiC tests. 2015. URL: http://www.w3.org/QC/.
15. Accessibility Support Database. 2015. URL: http://www.w3.org/WAI/accessibility-support/.