Three Web Accessibility Evaluation Perspectives for RIA

Nádia Fernandes, Ana Sofia Batista, Daniel Costa, Carlos Duarte, Luís Carriço
LaSIGE/University of Lisbon
Campo Grande, Edifício C6
1749-016 Lisboa, Portugal
{nadiaf,abatista,dancosta,cad,lmc}@di.fc.ul.pt

ABSTRACT

With the increasing popularity of Rich Internet Applications (RIAs), several challenges arise in the area of web accessibility evaluation. A particular set of challenges emerges from RIAs' dynamic nature: original static Web specifications can change dramatically before being presented to the end user: a user triggered event may provide complete new content within the same RIA. Whatever the evaluation alternative, the challenges must be met.

We focus on automatic evaluation using the current W3C standards. That enables us to do extensive evaluations in order to grasp the accessibility state of the web eventually pointing new direction for improvement.

In this paper, we present a comparative study to understand the difference of the accessibility properties of the Web regarding these different evaluation perspectives: 1) before browser processing; 2) after browser processing (dynamic loading); 3) and, also after browser processing, considering the triggering of user interaction events.

The results clearly show that for a RIA the number of accessibility outcomes varies considerably between these three perspectives. First of all, this variation shows an increase of the number of assessed elements as well as passes, warnings and errors from perspective 1 to 2, due to dynamically loaded code, and from 2 to 3, due to the new pages reached by the interaction events. This shows that evaluating RIAs without considering its dynamic components provides an erroneous perception of its accessibility. Secondly, the relative growth of the number of fails is bigger than the growth of passes. This signifies that considering pages reached by interaction reveals lower quality for RIAs. Finally, a tendency is shown for the RIAs with higher number of states also exposing differences in accessibility quality.

Categories and Subject Descriptors

H.4 [Information Systems Applications]: Miscellaneous; D.2.8 [Software Engineering]: Metrics—complexity measures, performance measures

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

Copyright 2013 ACM 978-1-4503-1844-0/13...

General Terms

Measurement, Human Factors.

Keywords

Web accessibility, Web science, Web browser processing, Automated evaluation.

1. INTRODUCTION

Rich Internet Applications (RIAs) are becoming a new trend on the Web. These are no longer static Web pages but rather complex applications that approximate the behaviour of common native desktop applications. Amongst the vehicles that enable this complexity of RIAs are scripting languages, such as Javascript/AJAX. When executed, these can modify several features, including the page content, layout, etc., within the same URI, sometimes without leaving the browser client side context. As a consequence a URI can correspond to a complex graph of states [19], which are only: 1) complete after all browser processing is done (e.g. load events); 2) reachable though triggering the events (e.g. mouse click, key press) available through each static component (state).

Regarding accessibility, RIAs provide real opportunities and treats. Opportunities, as, for example, they provide means to enrich and diversify the presentation and adapt it to specific users’ needs. Enriching the applications may seriously preclude the use of Assistive Technologies (AT) not ready to cope with the dynamic content. Also, a RIA can easily and automatically load hidden components, again and again, containing a significant amount of bad-practices, which are, on first analysis, not detectable.

Evaluating the accessibility in RIAs may, thus, be an exhausting task. From an end-user point of view, each change in presentation and interaction element, reachable through one of the possible RIAs events, should be also checked for accessible compliance. That means letting the browser process the automatic events and following all the interaction opportunities available, covering the whole graph of states of the application. Here, as in large-scale evaluations, automatic evaluators can play a determinant role [14, 17].

Most existing automated evaluators are not capable of assessing dynamically injected content. Also, to our knowledge none is able to fully access the underlying scripting languages. Therefore, the use of a traditional evaluator may lead to an erroneous accessibility evaluation, as several problems and quantities may pass unnoticed. This problem was partially exposed in a preliminary study by Fernandes et
al [9]. There, the authors addressed a small number of RIAs and a partial view of the whole page rendering process.

In this paper, we perform a deeper analysis. We extend the sample of assessed Web sites to over 10000, of which more than 8000 included some form dynamic change through user interaction events. Also, we use an updated version of QualWeb Evaluator [9], implementing a larger thoroughly revised set of Web Content Accessibility Guidelines (WCAG) 2.0 techniques [5]. Finally, we make a comparative analysis of three perspectives of evaluation: 1) corresponding to the traditional approach of evaluation before browser processing; 2) considering the evaluation after browser processing; 3) and, also after browser processing, considering the triggering of user interaction events. In the latter we look into the influence of states in the evaluation outcomes.

2. WEB BROWSING PROCESS

The dynamics of Web pages centre around a sequence of communication steps between the Web browser and Web servers, as depicted in Figure 1.

2.1 Web Page Loading Process

Several steps are executed before users are able to interact with the Web page, as depicted in Figure 2.

The first step in the Web page loading process, a Request, concerns with getting the resources that compose the Web page. Another step is the parsing of these resources in order to build the HTML Document Object Model (DOM) tree. Afterwards, the Web browser triggers the DOM Ready event, when DOM hierarchy has been constructed. Finally, the DOM Load event is triggered after all the initial script execution and resources are rendered (e.g., CSS, images, etc.).

Dynamic Web pages often attach a set of behaviours to these events. This way, scripts are executed before the user gets the chance to start interacting. Since the HTML DOM tree is available for manipulation by these scripts, they can enable the addition/removal/transformation of this tree. Consequently, the Web page presented to the user (page after browser processing) might be significantly different from the URI's resource representation that is initially transmitted to the browser from the Web server (page before browser processing).

In Rich Internet Applications (RIAs) new content can be obtained using Javascript/AJAX, without refreshing or loading a new page. A user interaction (Figure 2) can easily modify visible elements without requesting data from a server (e.g., through Javascript). Alternatively an AJAX request to the server can fully modify the presented content. In both cases a new version of a page will be available without changing the URI [16, 27], i.e., a new state of the Web page.

Figure 3 shows an example of two possible states of a Web page. Clicking a button ("Button") executes Javascript code that injects more elements in the page with new elements that should be evaluated. Ultimately the AT a user is utilizing must cope with this new content. Note that the URI is the same and the access to the new elements can only be done if the Javascript is interpreted.

2.2 Research Hypothesis

Considering the way Web browsers interpret Web pages, as detailed above, and taking into account that users interact with these Web resources through browsers, and possibly ATs, we post that:

When evaluating RIAs, Web accessibility tech-
niques should be applied to what is ultimately presented and interacted with by the end-users, to its full extent (i.e. covering all states).

Of course, alternatives may be envisaged, among which, for example, a full source code analysis could be done including scripting interpretation. Also, we recognise that other aspects of RIAs should be taken in consideration, like new complex elements (e.g. canvas) or WAI-ARIA (Web Accessibility Initiative - Accessible Rich Internet Applications) properties [15]. We believe that these should be considering in addition to covering the whole RIA states’ graph.

From that rationale, we devised the following research hypothesis that serves as the basis for our experimental study:

Evaluating Web content that considers fully processed RIA states will provide different accessibility evaluation outcomes.

3. RELATED WORK

Web Accessibility Evaluation (WAE) is a procedure to analyse how well the Web can be used by people with different levels of disabilities [14]. Unfortunately, current studies show that many Web sites still cannot be accessed in the same conditions, by a large number of people [14, 17].

WAE can be performed in two ways: users’ based or experts’ based. The users’ based evaluation is carried-out by real users; they can verify how well the accessibility solutions present in the Web sites match their own needs. However, assessment by users is often subjective. Furthermore, user testing is necessarily limited in scale, thus leaving a substantial number of problems out.

Experts’ based evaluation can be performed manually or automatically. The first is focused on the manual inspection of Web pages. Contrarily to the one above, it is performed by experts and it can provide a more in-depth answer to the accessibility quality of a Web page. This type of evaluation should be considered as a complementary evaluation rather than a replacement of user-evaluation. However, it is a time-consuming process too, and it can bring potential bias in the comparison of the different Web pages’ accessibility quality [14, 17].

The automatic evaluation is performed by software. The expertise is embedded in a software framework/tool. The evaluation is carried out by the tool without the need of direct human intervention. The main benefits are scalability and objectivity [17], enabling the conduction of large-scale studies, like [18] or [8] for example. However, it has limitations that both direct or user’s evaluations do not have, e.g., the depth and completeness of analysis. Again, it is a trade-off and often constitutes a complement to manual evaluations.

Experts’ evaluations rely on knowledge. Especially for the automatic version, the focus of this work, that knowledge is expressed in a set of guidelines, preferably in a way that can be automated.

Web Content Accessibility Guidelines (WCAG) [3] are one of the most used technical standards for accessibility evaluations, encouraging creators (e.g., designers, developers) in constructing Web pages according to a set of best practices. If this happens, a good level of accessibility can be guaranteed [14, 17]. These guidelines can be used in automatic evaluations.

The results of an accessibility evaluation can be used to measure quantitatively the level of accessibility of a Web page. Metrics are important to facilitate understanding, control, and improvement of products and processes in software development [11]. In terms of accessibility, metrics can also help the user to understand if a Web page/site can be used by them. In addition, they provide an important tool to understand whether companies may improve the accessibility of final products or companies beginning the software development can introduce accessibility issues in the development process.

The metrics results can be obtained with the aid from automatic evaluations. Some examples of metrics include: Failure Rate [21], UWEM [22], and WAQM [23]. Some authors say that the metrics should not be dissociated from the users, and consider the necessary effort to perform the repair of the accessibility problems of the pages [6, 20]. We concur. However, metrics can also be used to understand the Web accessibility behaviour through large-scale evaluation with hundreds or thousands of evaluations. That does not mean dissociating metrics from users. It simply means that some users view them macroscopically.

3.1 In Browser Evaluation

The predominant technologies in the Web were HTML
and CSS, which resulted in static Web pages. However, current Web pages, by means of user or automated events, can change their content. Thus, the final presented content can be different from the initially loaded by the Web browser [10].

Unfortunately, most of the current automated evaluators [1] are still not capable of detecting those changes. Nevertheless, some can be pointed that already work on after browser processed DOM trees, or in Browser Context. Examples are Fuzzybility, Mozilla/Firefox Accessibility Extension and WAVE Firefox toolbar [12]. Unfortunately they are still not using WCAG 2.0. Another example is Hera-FFX 2.0 [13]. Being a semi-automatic evaluator, the evaluation outcome can be improved and more accurate through human intervention. However, this approach is hardly compatible with large scale evaluations. In general, most of the existing tools aim at interactive evaluation, on a page by page, or small scale basis.

Table 1 presents several Web accessibility evaluators currently used, considering: 1) if they perform the evaluation before processing, 2) the version of WCAG used, 3) if they perform the EARL reporting (standard format for Web accessibility report), and 4) some notes that we considered appropriate. Through the analysis of this table, we can conclude that none of these evaluators assess RIAs, nor meet the requirements for the characteristics we have identified as relevant.

In RIAs this problem is aggravated, because of the states that can be triggered in the same page. Current evaluators would not be able to identify this changes and consequently not able to evaluate their accessibility problems. Therefore, these technologies are used and combined in new ways that threaten accessibility [4].

Besides, the increasing usage of video components on the web is becoming a relevant problem when considering, for instance, deaf or blind people. Text alternatives should be made available. Dynamically created content and AJAX based Web pages are growing and most of them are not considered accessible as screen readers, such as JAWS, do not seem to work satisfactorily [26]. Dynamic Web enables the change of the Web page’s content and structure, usually by displaying or hiding information and HTML elements, in- injecting new HTML code or even removing it [30]. As can be seen, guaranteeing that the advantages of this new “Web of applications” are available to everyone, demands for a complete and rigorous accessibility evaluation process capable of handling the characteristics of these type of Web pages.

### 3.2 Rich Internet Application coverage

A fundamental problem in evaluating RIAs’ accessibility is identifying the complete states-graph of the application. Watanabe et al [24] tested the accessibility requirements in RIAs. Their proposal simulates keyboard events to change the DOM tree of the Web pages. The objective was to produce AT scenarios and to know if it was possible to navigate through the page with those events. However, their system did not allow them to simulate all the possible user interactions with the page.

Mesbah et al [19] presents a tool to perform automated tests of AJAX applications, taking into account the dynamics of these applications. The strategy was to access event changes by adopting a Web crawler capable of triggering the events on, for example, the clickable elements of the user interface. Still, the tool confines its crawling to AJAX requests and does not provide accessibility evaluation means.

Recently, Doush et al [7] designed a conceptual RIA accessibility evaluation tool. The idea of this framework is to check the accessibility of the visible content of a Web page. Also it provides a report with the content that should appear in the Web page (considering ARIA specifications). However, this framework is not yet implemented.

### 4. QUALWEB

To perform the accessibility evaluation, we used the QualWeb evaluation framework [9]. The architecture (depicted in Figure 4) is composed by three major modules: QualWeb evaluator core, Browser Processing Simulator, and Interaction Simulator.

![Figure 4: Architecture of QualWeb.](image)

#### 4.1 QualWeb core

The Browser Processing Simulator (BPS) receives the URI of the page to evaluate and process it. If the page has states, it sends the DOM of the page to the Interaction Simulator, which simulates the several states. Then, the Interaction Simulator send them to the BPS to be processed, which posteriorly forwards the processed DOM of the pages to the QualWeb core to be evaluated.

This way, the DOMs are then fed to the QualWeb evaluator core that cumulatively assesses the quality of the RIA.

To extract the CSS from the DOM tree is used the CSS Pre-Processor which obtains all the CSS of the page (i.e., internal, external and in-line). Next, to perform the evaluation QualWeb uses the features provided by the Techniques component. It uses the Formatters component to tailor the results into specific serialisation formats, such as: EARL reporting [2], since EARL is a standard format for accessibility reporting; comma-separated-values (CSV) for statistical proposes; or JSON if we want to use QualWeb as a Web service. If the results report are formatted in EARL, it can be interpreted by any tool that understands this format, and
Table 1: Web Accessibility evaluation tools. BBP - evaluation before Browser Processing, ABP - evaluation after Browser Processing.

<table>
<thead>
<tr>
<th>Name</th>
<th>BBP</th>
<th>ABP</th>
<th>WCAG 1.0</th>
<th>WCAG 2.0</th>
<th>RIA</th>
<th>EARL Reports</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>A-Checker</td>
<td>√</td>
<td>√</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TAW Standalone</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Foxability</td>
<td>√</td>
<td>√</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Functional Accessibility Evaluator</td>
<td>√</td>
<td>√</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>In upgrade to WCAG 2.0. It used alternative guidelines to WCAG.</td>
</tr>
<tr>
<td>WAVE</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td></td>
<td></td>
<td></td>
<td>Semi-automatic tool</td>
</tr>
<tr>
<td>Hera-Yx 2.0</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

even allow comparing the results with other tools.

In this version of QuaWeb, we added more techniques. We currently implement 31 HTML and 13 CSS WCAG 2.0 techniques (i.e., the basic practices of creation of accessible content as basis of testable success criteria). Thus, a total of 44 accessibility techniques were used for the evaluation.

4.2 Browser Processing Simulator

This module simulates the processing of the Web browser using PhantomJS\(^1\). PhantomJS is a command-line tool that uses WebKit. WebKit is an open source web browser engine used in some of the most disseminated web browsers [25]. PhantomJS works like a WebKit-based web browser, interpreting web applications, without rendering them on a display a headless WebKit browser with a JavaScript API.

The absence of display rendering, still interpreting the applications, enables this module to perform sequential evaluations in browser context, in a fast and thought-out way. This favours large-scale evaluations. The access to after the Web browser processing is achieved through the onLoadFinished event available in the PhantomJS API.

4.3 Interaction Simulator

To cope with the challenges of the RIAs, we have integrated an Interaction Simulator. This component is responsible for simulating user actions and triggering the interactive elements of the interface. As a result we have access to all the different states (in DOM format) of RIAs.

To perform the simulation, we use JQuery to interact with clickable elements (e.g., button, input, div). For every state we get the children states (those reachable through clickable elements) and processed them in the BPS. Then, for each child we verify if it was already visited. If not we repeat the process. At the end we obtain the full interaction state graph of the RIA. Figures 5 and 6 show the DOM trees of the initial and state resulting of click in the link.

4.4 Validation and Testing

4.4.1 WCAG Techniques

A test-bed was developed, in order to verify that all the WCAG 2.0 implemented techniques provide the expected results. The test-bed is constituted by a set of HTML and CSS test documents, based on documented WCAG 2.0 techniques and ancillary WCAG 20 documents. Besides, each HTML test document was carefully hand-crafted and peer-reviewed (within the research team), in order to guarantee a

\[^1\]http://phantomjs.org/

\[^2\]http://www.alexa.com/topsites

Figure 5: DOM tree of a initial state of a page.

Figure 6: DOM tree of the second state of a page.

high level of confidence on the truthfulness of implementation. Success or failure cases were performed for each technique, to test the possible techniques' outcomes (unit tests). First static examples were implemented, e.g. static html code that produces fail/warn/pass according to the guidelines. Then, the test-bed also considers dynamic cases. For each technique a corresponding test that dynamically generates the same code was created. That code can be generated on the page load or as a result of a user interaction. The evaluation outcomes (warn/pass/fail by technique) for all HTML/CSS test documents were compared and checked for consistency.

Additionally, we performed an expert analysis and compared its results with the ones of QualWeb. We performed it on some of the pages the more used Web sites (from Alexa Top 100 Web sites\(^2\)), using WCAG 2.0 also. We inspected before browser processing, after browser processing, and after processing considering the states of the pages. For the implemented techniques the results were consistent, with the exception of the warnings. Those were considered fails by the experts in some cases and passes in others.
5. EXPERIMENTAL STUDY

Three outcomes are gathered:

We used the QualWeb to compare Web accessibility evaluations under the following three conditions:

- **Evaluation 1 - E1**: evaluation performed before Web browser processing – Interaction Simulator and Browser Processing Simulator turned off.

- **Evaluation 2 - E2**: evaluation performed after browser processing, without consider states of the pages – Interaction Simulator turned off and Browser Processing Simulator active.

- **Evaluation 3 - E3**: evaluation performed after browser processing, considering the different states of the pages – both modules active.

Figure 7 present a schematic representation of the evaluations.

5.1 Setup and Measurement

We gathered 14000 URI by crawling the Web, in middle of January 2013. The seed was a Portuguese portal, although the obtained URI were not confined to the Portuguese domain. The list of URI was split and feed into 20 instances of the QualWeb evaluator running in a corresponding number of PCs. The evaluation took approximately 10 hours.

The results of the evaluation are presented in terms of **PASS, WARN and FAIL**: pass or fail, if the elements (or certain values/characteristics of the elements) verified by the techniques are in agreement or disagreement with the W3C recommendations for the techniques, respectively; and **Warning** – if it is not possible to identify certain values/characteristic of an element as right or wrong, according to the W3C recommendations for the techniques (without a need of an expert intervention).

We used **strict rate** metric [18], where WARN results are not considered \(\text{(strict rate } = \text{pass/(pass + fail)})\). It is normalized into a percentage, where the results are between **accessible** (100%) and not accessible (0%). This is of course not an absolute value of accessibility. Amongst others, warnings should be disambiguated and the techniques should be integrated in success criteria, for example. However, since we are comparing the results in the different conditions, the comparison between this metric’s results provides a sufficient indication of the relative accessibility quality.

5.2 Results

Our evaluations observed a total of 11860 viable URIs, and 20869 CSS files. A total of 2140 URIs were no longer available. From these pages, we withdrew the ones with only one state. This way we focused on understanding the states influence on the evaluation results. Finally, we proceeded with a total of 8282 URIs.

The average number of evaluated elements per RIA was: 1152.21 elements in E1; 1665.7 elements in E2; and 19964.00 elements in E3 (Figure 8). The number of elements in E2 is higher than in E1 \(\left(\frac{\text{ratio}_{E2/E1}}{1.45}\right)\), and the number of elements is higher in E3 then in E2 \(\left(\frac{\text{ratio}_{E3/E2}}{11.99}\right)\).

Figure 8: Average of number of HTML elements by type of evaluation.

Considering the states ratio in E3 we get 1715.10 elements per RIA, per state: a number similar to the average number of elements in E2. This means that the states reached by the interaction events have a similar number of states that the main one. A closer look shows a standard deviation of 4.13.

5.2.1 Evaluation Outcomes

Figure 9 presents the outcomes of the evaluations. It can be observed that the outcomes increase in E2 relatively to E1 and in E3 relatively to E2. The average numbers of outcomes per RIA are relevant for the analysis as they show what has been ignored if states are not considered.

We also obtained the average number of outcomes per RIA, per state, to understand if states behave similarly to the main URI. The figure 10 shows these values. In this case the average numbers are similar. We notice a slight decrease in the number of outcomes per state.
on the number of passes and fails from E2 to E3 per state, and a growth of warnings.

Table 2 show the ratios from the evaluations, including the E3 per state values. Most values show an increase from E1 to E2 and E2 to E3, especially in the latter. Pertaining to the values per state, the ratios of E3 per state and E2 shown a slight decrease in the number passes and fails. Nevertheless, it is worth noting that the fails ration is bigger than the pass.

Table 2: Ratios per outcome.

<table>
<thead>
<tr>
<th>Ratio</th>
<th>Pass</th>
<th>Fail</th>
<th>Warn</th>
</tr>
</thead>
<tbody>
<tr>
<td>E2/E1</td>
<td>1.21</td>
<td>1.17</td>
<td>1.48</td>
</tr>
<tr>
<td>E3/E2</td>
<td>10.49</td>
<td>11.40</td>
<td>12.35</td>
</tr>
<tr>
<td>E3 states/E2</td>
<td>0.90</td>
<td>0.98</td>
<td>1.06</td>
</tr>
</tbody>
</table>

Considering states, we detected on average 12.51 possible states per URI, in E3. This means that, on average, each URI can have approximately 12 different states, triggered by users’ interaction that will be presented to the AT they use. The average of the standard deviations of the outcomes per RIA shows that the accessibility differences between main URI and the remainder assessed states in small: average pass per RIA per state is 3.32 (SD 0.02); average fail per RIA per state is 88.12 (SD 0.30); average warning per RIA per state is 316.57 (SD 0.65).

Regarding the use of the strict metric, we have found a small decrease in accessibility quality from E2 to E3. However, in both cases the quality is really low (0.041 for E2 and 0.039 for E3) taking into to account that 0 (zero) is the lower value (no passes) and 1 is the higher value (no fails) of accessibility quality.

Figure 11 shows the results of the strict metric for RIA with increasing number of states. A trend line was included revealing a tendency on accessibility quality with the increasing number of states. We excluded the number of states that had less that 5 represented RIA. For example, there was a single RIA with 16 states and two with 25 that were therefore excluded from the figure. An interesting characteristic of the URI with 16 states (www.radicim.sapo.pt) was that the strict metric result was a fairly high comparing with all the others (0.17). A closer look revealed a radio web site.

6. DISCUSSION

Our study has yielded interesting insights respecting the automated Web accessibility evaluation practices. Revisiting our initial research hypothesis we in fact can confirm that:

Evaluating Web content that considers fully processed RIA states will provide different accessibility evaluation outcomes

According to the posted rational that Web accessibility techniques should be applied to what is ultimately presented and interacted with by the end-users, then we may conclude that accessibility evaluation should be applied to whole set of RIAs states.

The results obtained show that E3 presents a higher average of outcomes, relatively with E2 and E1. The case for E1 and E2 was discussed elsewhere [10]. The case for E3 derives directly from the number of evaluated elements as each interaction provides a new state.

A closer look revealed that in a majority of RIAs the new states are comparable, but not equal, to the main URI. Interaction merely changes some data on the destination state or a small number of attributes. As a consequence the difference of the evaluation outcome are also small. Also shown by the small standard deviation.

Although this is true, however, the errors introduced by these changes would not be detected by a classic automatic
evaluation method. That means that a user may find a barrier on an otherwise barrier-free RIA, just because he/she interacted with it. Dynamic content has shown by these results, and those discussed in [10], does in fact hinder accessibility barriers.

Another interesting aspect of the above results emerges from the relation between the number of states of a RIA and the perceived accessibility quality. These may reveal that increasingly complex RIAs are prone to have more accessibility problems. This conclusion needs to be further investigated.

The cost/benefit has also to be considered. For instance, E3 outcomes do provide a more exhaustive accessibility evaluation of RIAs, and in the end it will provide a more detailed view of the RIA accessibility issues. Nevertheless, for a large number of states (and of pages) it can be overwhelming.

Designers can develop more accessible content, if they use evaluation procedures that consider all the possible states of the Web application. This happens because they have access to a more complete page/applications analysis, which they may use to improve the accessibility quality of the page/application.

6.1 Limitations
Our experiment has faced some limitations on the type of results that can be extrapolated, including:

- **Techniques coverage**: it would be important to have ARIA techniques implemented and adopt new emerging techniques that will, for sure, appear from the conformance to HTML 5 new features;

- **States flow-graph**: the states detection algorithm has to be improved so that it becomes able to detect the complete flow-graph of the RIA. For now, we were more focused in clickable events, but events such as `onFocus` may, for instance, as well originate changes in the content of the page;

- **Duration of evaluation**: to minimize the duration, instead of evaluating a new state as a new page when we find it, we should ponder to compare its DOM with the original page DOM. If this operation takes less time than evaluating all the new states of the page, we should only evaluate the new elements. This is particularly important in large-scale evaluations;

- **Cross-Site scripting**: in some Web pages the injection scripts are blocked with cross-site scripting (XSS) dismissal techniques. In these cases, we are not able to inject JQuery (if necessary) to simulate the interaction with the pages;

- **Automated evaluation**: since this experiment is centered on automated evaluation, it shares all of the inherent pitfalls.

Next we present the conclusions of the experiment, and try to synthesize the important points.

7. CONCLUSIONS AND FUTURE WORK
This article presented a large-scale study of accessibility on over 8000 RIAs. The results of this study allowed us to compare three accessibility evaluation approaches: before and after Web browser processing, and after browser processing with states.

Web pages have become more complex and evolved from simple information display into RIAs. We can conclude that there are, in fact, differences between these three approaches. Thus, we were able to verify that Web Accessibility Evaluations after browser processing, considering all the possible states of the page, really provide a more accurate and in-depth analysis of page accessibility.

In conclusion, regular Web accessibility evaluations or even evaluators which consider browser processing but without states, overlook 92% of the states of the pages (for 11860 Web pages). Consequently, the accessibility problems on those states are ignored. This is a reality that has to be changed, so that Web pages accessibility may be improved.

As for the future, we are aiming to improve the states detection algorithm; and detect the actual differences between states of a RIA. With that we can not only find the impact of states for end user accessibility, but also have a measure of the effort needed to correct the page.

8. ACKNOWLEDGEMENTS
This work was funded by Fundação para a Ciência e Tecnologia (FCT) through the QualWeb national research project PTDC/EIA-EIA/117058/2008 and through the FCT national research project PTDC/EIA-EIA/117058/2010.