Towards a Service-based Architecture for Web Accessibility Federated Evaluation

José R. Hilera, Salvador Otón, Carlos I. Martín-Amor
Dept. Computer Science
University of Alcalá
Alcalá de Henares, Spain
e-mail: jose.hilera@uah.es, salvador.oton@uah.es, carlosivan.martin@edu.uah.es

Cristian F. Timbi-Sisalima
Grupo de Investigación en IA y Tecnologías de Asistencia
Universidad Politécnica Salesiana
Cuenca, Ecuador
e-mail: ctimbi@ups.edu.ec

Abstract— **This paper presents a work in progress aimed to develop a universal architecture based on Web services and semantic Web technologies, for evaluating Web accessibility by using a federation of multiple evaluation tools, and compounding a unique result report combining semantically the reports obtained by each tool. The definition of a standard interface for evaluation services is proposed, and its implementation using RESTful Web API (Application Programming Interface) is described. Services for automatic semantic composition of reports are described using W3C (World Wide Web Consortium) standards as RDF(S) (Resource Description Framework Schema), OWL (Web Ontology Language), SPARQL (Sparql Protocol and RDF Query Language) and EARL (Evaluation and Report Language).**

Keywords-Web accessibility; a11y; Web service; federated evaluation; semantic Web; Web api; metatool.

I. INTRODUCTION

The accessibility of a Web site is essential to make it understandable, usable and practical for all users, including disabled people. To help determine the accessibility of a Web site, the World Wide Web Consortium has published the Web Content Accessibility Guidelines (WCAG) [1] that have been adopted as an international ISO (International Organization for Standardization) standard [2]. This standard establishes the minimum requirements for a Website to be accessible, overcoming barriers of access to any type of user. Other organizations have published their own Web accessibility requirements, such as Section 508, by the United States government; BITV (Barrierefreie Informations Technik Verordnung) by the Germany Government; or Stanca Act, by Italy Government. In general all these have much in common with WCAG that defines 61 accessibility success criteria to be satisfied by Web applications or Websites. To quantify the accessibility of a Website, the standard has created three levels of compliance: level A is reached by a Web site that accomplishes 25 specific success criteria, AA level requires meeting other 13 criteria, and AAA level is obtained when all criteria (61) are satisfied.

Developers and testers of Websites can verify the success criteria using accessibility evaluation tools. The W3C maintains a Web page with the list of the most important tools [3], including online tools. In general, an online evaluation tool is a Web application that allows the user to enter the URL (Uniform Resource Locator) of the Website to be tested obtaining an assessment report, which includes the accessibility requirements verified, those non-verified (requiring manual assessment), errors found and warnings. However, not all the tools have the same efficiency making essential the execution of different tools to complement the results. This procedure is tedious, as each tool uses a different user interface with different options and various formats for the results. To solve this problem, this paper presents a work in progress aimed to develop a universal architecture for evaluating Web accessibility using a federation of multiple evaluation tools, and compounding a unique results report combining semantically the reports obtained by each tool.

The following section describes the proposed architecture. In Section III, the definition of a standard interface for the accessibility evaluation services included in the architecture is presented. Section IV describes the basic services proposed for semantic composition of results reports. In the final section, some conclusions and other related works are presented.

II. SERVICE-BASED ARCHITECTURE

The proposal architecture is for any accessibility evaluation meta-tool system, which uses other evaluation tools in a federated way, receiving an evaluation request from a user, launching calls to a pool of federated tools, and finally compounding a single assessment report for the user, based on the results of each of the remote tools invoked. To implement this system, a service-based architecture with three layers is proposed (Figure 1). The choice of an architecture based on Web services is mainly given by the independence of the implementation details of each of the assessment tools available online. Each tool is independent from the rest, establishing its own decisions, and acting under its own autonomy. The levels proposed are as follows:

Layer 3. It represents the user (person or software) who wants to perform a simultaneous evaluation of the accessibility of a Website using multiple assessment tools. To do this, the user must provide the URL, or HTML (Hyper Text Markup Language) code, of the page to analyze, the accessibility evaluation standard to apply (WCAG 2.0, Section 508, etc.), and data about the federation of tools (at least, the selection of tools to participate in the federated evaluation).

Layer 2. At this level there are the services that manage user interface (*Front-end management*), and process the user data (*Back-end management*). From these data, a federation service module is responsible for determining how to connect with the final assessment tools provided by the user. Then, launches the requests for evaluation to those involved, and receives reports with the results. At this level, there are other support services, explained in Section IV, for filtering and adapting the format of reports, and the semantic composition of a single overall evaluation report.

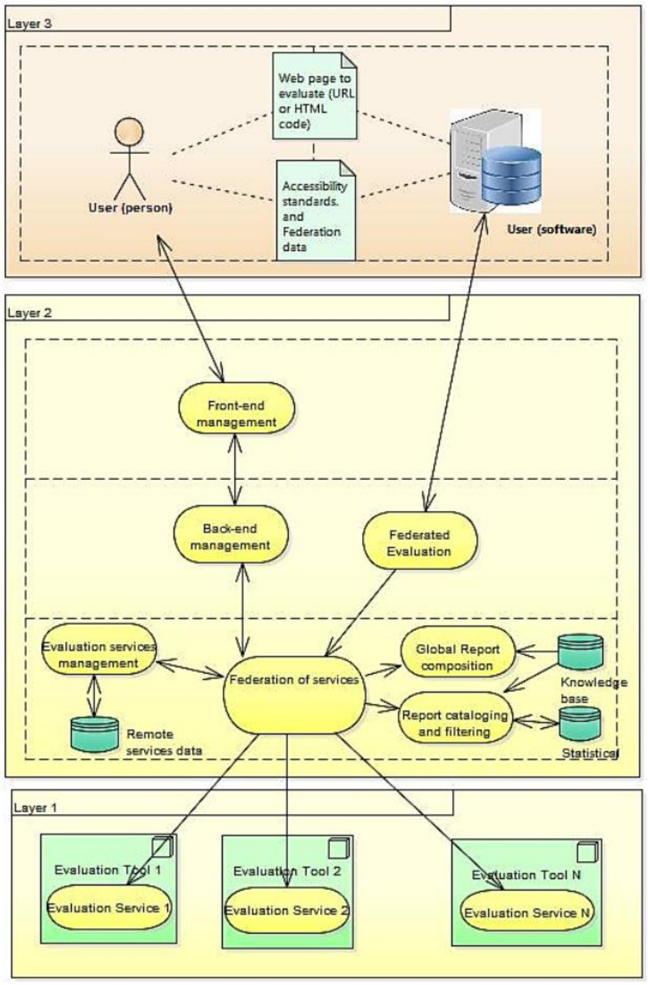


Figure 1. Service based architecture for accessibility evaluation.

Layer 1. It includes remote tools that perform the evaluation of the accessibility of Web sites, and should expose its functionality as a Web service *(Evaluation Service*) with a standard interface, as it is explained in the following Section.

III. STANDARD INTERFACE FOR EVALUATION SERVICES

There is a growing number of online Web accessibility evaluation tools. And some of them expose their functionality through Web services using RESTful Web API technology [4]. This is the case of the following free use tools: AChecker [5], OWA [6], Tenon [7] and WAVE [8].

Surely, the tendency to offer functionality through Web services will be extended to other existing online tools, which are now Web applications with user interface to be used with a Web browser. The problem is that every tool creates its own services, and its own input and output parameters, being different in all cases. As an example, Table I shows the input parameters of the evaluation service exposed by the four cited tools. It can be seen that, in general, at least the address of the Website to evaluate and a user ID or password are required. But there are tools offering other optional parameters, such as the evaluation standard to be applied. This is the case of AChecker with an optional "*guide*" parameter, whose default value is "WCAG2-AA", but that can refer to other standards such as the American "Section 508", the Italian "Stanca-act" or the German "BITV1". The other tools do not offer this possibility, because they only evaluate according to the WCAG standard.

TABLE I. API REQUEST PARAMETERS OF EVALUATION TOOLS

AChecker	Required: uri, id. Optional: guide, output, offset.
OWA	Required: format, url, key. Optional: level, resolution.
Tenon	Required: key, url (or src). Optional: appID, certainty, waitFor, fragment, importance, level, priority, ref, store, projectID, uaString, viewPortHeight, viewPortWidth.
WAVE	Required: key, url. Optional: format, reporttype.

Also, the information that the tools include in response to the request made after the evaluation of accessibility is different in each case. Table II shows the data in response.

TABLE II. API VALIDATION RESPONSE OF EVALUATION TOOLS

AChecker	**resultset**, **summary**, **status**, **sessionID**, **NumOfErrors**, **NumOfLikelyProblems**, **NumOfPotentialProblems**, **guidelines** (guideline, results (result (resultType, lineNum, columnNum, errorMsg, errorSurceCode, repair, sequenceID, decisionPass, decisionFail, decisionMade, decisionMadeDate))), **errors** (totalCount, error (message)).
OWA	**Date, message, result** (elements (forms, iframes, images, links, linksImages, tables, total), image, level, principles, resolution, summary, url).
Tenon	**status**, **message**, **documentSize**, **responseExecTime**, **responseTime**, **sourceHash**, **request** (appID, certainty, docID, importance, key, level, priority, priorityWeightissueLocation, ref, responseID, projectID, uaString, url, viewport, fragment, store), **clientScriptErrors** (message, stacktrace), **globalStats** (allDensity, errorDensity, warningDensity), **resultSummary** (density, issues, issuesByLevel, tests), **resultSet** (bpID, certainty, priority, errorDescription, errorSnippet, errorTitle, position, ref, resultTitle, signature, standards, tID, xpat), **apiErrors** (line, message, sourceId, tID).
WAVE	**status** (error (code, description)), **categories** (description, count, items (count, id, description)), **statistics** (creditsremaining, pageurl, pagetitle, waveurl, time, allitemcount, totalelements).

Another problem is the format of the result. AChecker, OWA and Tenon allow to optionally specify the format as an input parameter: HTML, XML (eXtensible Markup Language), JSON (JavaScript Object Notation); while Tenon always returns the information in JSON format.

In short, it can be seen that it is difficult to combine all the results and unify the input parameters, so it would be necessary to create a standard interface to be met by tools interested in participating in the federation of their services. Authors of this paper are working on creating a universal interface proposal, covering all possible input data, and producing a result with self-descriptive structure, based on the standard language EARL [9].

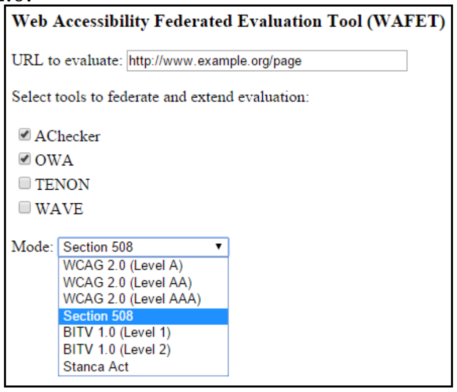
TABLE III. EXTRACT OF AN EVALUATION REPORT USING OWA API

```
@prefix earl: <http://www.w3.org/nss/earl#> .
@prefix ptr: <http://www.w3.org/2009/pointers#>.
@prefix doap: <http://usefulinc.com/ns/doap#> .
@prefix a11y: <http://example.org/a11yResources.owl#> .
@prefix wcag2: <http://www.AccessibleOntology.com/WCAG2.owl#>.

ex:assertion_OWA a earl:Assertion ;
earl:assertedBy a11y:OWA_API ;
earl:subject <http://www.example.org/page.html> ;
earl:test wcag2:SuccessCriterion_111;
earl:result ex:OWAResult .

wcag2:SuccessCriterion_111 a earl:TestRequirement .

a11y:OWA_API a earl:Software;
  doap:name "OWA Web Service API";
  doap:homepage <http://observatorioWeb.ups.edu.ec/oaw/apirest.jsf> .

ex:OWAResult a earl:TestResult;
 earl:outcome earl:failed;
 earl:pointer ex:ptr1_OWAResult .

ex:ptr1_OWAResult a earl:Pointer, ptr:LineCharPointer;
ptr:lineNumber  "37";
ptr:charNumber  "8" .
```

Tables III and IV show EARL extracts of reports to be used in Section IV. In both cases, for simplicity, the reports are represented using the Turtle RDF serialization syntax [10], although they could be obtained in other formats such as RDF/XML or JSON-LD (JSON for Linked Data). The reports consist of a list of triplets *"subject predicate object"* to describe the results. For example, the report in Table III indicates that a fail has been found, because of the noncompliance with the success criteria 1.1.1 of WCAG 2.0 on line 37 in the Web page www.example.org/page.html, using as evaluation tool the API provided by OWA.

IV. SERVICES FOR SEMANTIC COMPOSITION OF REPORTS

A problem to be solved when trying to do a joint or federated evaluation of the same Website, is how to combine the results obtained by each tool. It may be the case shown in Figure 2, in which a user wants to use two tools, and to evaluate the Web page according to the rule Section 508 existing in the US. In this case, only the AChecker tool can evaluate Section 508. However, the other can make an evaluation according to a similar standard as is WCAG 2.0.


Web Accessibility Federated Evaluation Tool (WAFET)

URL to evaluate: http://www.example.org/page

Select tools to federate and extend evaluation:

- [✓] AChecker
- [✓] OWA
- [ ] TENON
- [ ] WAVE

Mode: Section 508
- WCAG 2.0 (Level A)
- WCAG 2.0 (Level AA)
- WCAG 2.0 (Level AAA)
- Section 508
- BITV 1.0 (Level 1)
- BITV 1.0 (Level 2)
- Stanca Act

Figure 2. Basic user interface of a federation based meta-tool.

The solution proposed here is to include in the architecture a service and a knowledge base, both based on semantic technologies such as ontologies, able to help determine the equivalence between both standards. In this context, semantic technologies refers to technologies that facilitate the description of the meaning of information, so that this information can be compared or combined with other information with similar meaning without the need of the intervention of a person, using for that a specific software, such as intelligent agents.

Ontologies are implemented using semantic Web techniques, such as OWL, RDF (S) and SPARQL [11]. And they are compatible with the EARL language to describe individual reports returned by each tool. There are ontologies to model the WCAG standard, as the one referenced in Table III, but new ontologies must be created to conceptualize other standards, as Section 508, and apply techniques for mapping ontologies to align all involved. The authors of this paper are collaborating in the creation of ontologies about accessibility to model standards (section508.owl in Table IV), but also to model and classify evaluation tools (a11yResources.owl in Table IV).

TABLE IV. EXTRACT OF AN EVALUATION REPORT USING ACHECKER API

```
@prefix s508: <http://example.org/section508.owl#> .

ex:assertion_AChecker a earl:Assertion ;
earl:assertedBy a11y:AChecker_API ;
earl:subject <http://www.example.org/page.html> ;
earl:test s508:req_1194_22_a;
earl:result ex:ACheckerResult .

s508:req_1194_22_a a earl:TestRequirement .

a11y:AChecker_API a earl:Software;
  doap:name "AChecker Web Service API";
  doap:homepage <http://achecker.ca/checkacc.php> .

ex:ACheckerResult a earl:TestResult;
 earl:outcome earl:failed;
 earl:pointer ex:ptr1_ACheckerResult .

ex:ptr1_ACheckerResult a earl:Pointer, ptr:LineCharPointer;
ptr:lineNumber  "25";
ptr:charNumber  "10" .
```

Table V shows Turtle RDF code for declaring in the knowledge base that the success criterion 1.1.1 of WCAG 2.0 is semantically equivalent to the requirement 1194.22 (a) of Section 508. Both standards require that in an accessible Web page, a text equivalent for every non-text element shall be provided. Thus, a combined report can be composed using SPARQL [12] and a software reasoner that can make inferences from rules in ontologies. The authors are using for this Jena, an open source Java framework that allows programming SPARQL queries and has integrated reasoning, but also it allows us embed external reasoners as Pellet, or FaCT Racer [13].

As an example, Table VI shows a possible query and Table VII the result. It can be seen that through the mechanism of reasoning in the knowledge base, it has been inferred that the requirement 1.1.1 detected by OWA is equivalent to 1194.22 (a) of Section 508, and appears as such in the report, since the user had indicated (Figure 2) that he/she wanted to evaluate accessibility according to Section 508.

TABLE V. EXTRACT OF THE KNOWLEDGE BASE

```
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix s508: <http://example.org/section508.owl#> .
@prefix wcag2: <http://www.AccessibleOntology.com/WCAG2.owl#> .

s508:req_1194_22_a   a s508:Requirement;
s508:hasDescription "1194.22(a) A text equivalent for every non-text
element shall be provided"@en .

s508:req_1194_22_a owl:sameAs wcag2:SuccessCriterion_111 .

wcag2: SuccessCriterion_111 a wcag2:SuccessCriterion;
wcag2:hasDescription "Non-text Content: All non-text content that is
presented to the user has a text alternative that serves the equivalent
purpose, except for the situations listed below."^^xsd:string .

wcag2:SuccessCriterion_111 owl:sameAs s508:req_1194_22_a .
```

TABLE VI. SPARQL SENTENCE TO OBTAIN A COMBINED REPORT

```
prefix earl: <http://www.w3.org/nss/earl#>
prefix ptr: <http://www.w3.org/2009/pointers#>
prefix doap: <http://usefulinc.com/ns/doap#>
prefix s508: <http://example.org/section508.owl#>
SELECT ?tool ?desc ?line ?char
WHERE { ?a a earl:Assertion .
        ?a earl:assertedBy ?tool .
        ?a earl:test ?req .
        ?req a s508:Requirement .
        ?req s508:hasDescription ?desc .
        ?a earl:result ?res .
        ?res earl:outcome earl:failed .
        ?res earl:pointer ?pt .
        ?pt ptr:lineNumber ?line .
        ?pt ptr:charNumber ?char . }
```

TABLE VII. RESULT OF THE SPARQL QUERY

tool	desc	line	char
AChecker_API	"1194.22(a) A text equivalent . . ."	25	10
OWA_API	"1194.22(a) A text equivalent . . ."	37	8

Another problem that arises by combining results is the possibility of inconsistencies between the results of different tools for the same accessibility requirement. For example, a tool can determine that the success criterion 1.1.1 is satisfied because all images have an alternative text, whereas another more advanced tool could determine that the goal was not met because it has detected as alternative text images the filename of the image archive, which is not suitable to describe an image. The proposed architecture includes a filtering service to help resolve these cases. This service must also manage preferences expressed by the user on how to resolve conflicts, but can also access the knowledge base and a database with statistical results of previous assessments that help determine the reliability of each tool for each requirement. This is in line with what other experts proposed [14].

V. CONCLUSIONS

This is a work in progress that provides a solution to the problem of interoperability between tools for evaluating Web accessibility, and allows automatic composition of evaluation reports from different sources. No other proposals have been found to solve this problem. There are studies that address part of the problem, as is the case of [15], which also proposed an architecture and use of EARL format to compare results from different tools, but neither combining results nor using Web services. There are also tools that reuse open source available from other evaluation tools, is the case of QuickCheck [16], reusing specific functionality of the tools Chrome Developer Tool, Axe Engine and HTML Code Sniffer, but without the possibility of combining the results or federation, as it is not based on Web services.

The main contributions of this work are mainly two: first, a new standard API for online accessibility evaluation tools that serves as a single and common interface through which it can access all the functionality offered by any of the current tools, and flexible enough to support new functionality to appear in the future. A second contribution is a new mechanism based on this interface that, using federation of services and semantic technologies, allows combining the results of evaluations of the same Website by different tools, applying different criteria or preferences established by the user.

The idea of federating services and sematic combination has been proposed and applied previously by the authors of this paper, having implemented similar architectures in other application areas, such as the federation of search results in distributed learning objects repositories [17]. This previous experience is now been applied to implement the proposed architecture for the case of Web accessibility evaluation.

We are currently working on implementing a first version of a prototype that meets the architecture and basic functional requirements that reflect the main ideas described in this paper. The goal is that it can be useful especially for evaluators who know standards such as WCAG or Section 508. In the future, additional requirements will be considered, such as those relating to the usability of the tool and ease of interpretation of the evaluation results, so that it can be used even by people who do not have a thorough understanding of Web accessibility standards.

ACKNOWLEDGMENT

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

REFERENCES

[1] Web content accessibility guidelines (WCAG) 2.0. World Wide Web Consortium, 2008.

[2] ISO/IEC 40500:2012, information technology -- W3C Web content accessibility guidelines (WCAG) 2.0. International Organization for Standardization, 2012.

[3] Web Accessibility Evaluation Tools List. Wide Web Consortium, 2014. http://www.w3.org/WAI/ER/tools/. [retrieved: 03, 2016]

[4] L. Richardson and M. Amundsen, RESTful Web APIs, O'Reilly Media, 2013.

[5] AChecker Web Service API. Inclusive Design Research Centre, 2011. http://achecker.ca/documentation/web_service_api.php. [retrieved: 03, 2016]

[6] Observatorio de Accesibilidad Web (OWA). Universidad Politécnica Salesiana, 2016. http://observatorioweb.ups.edu.ec/oaw/apirest.jsf. [retrieved: 03, 2016]

[7] Tenon API Documentation. Tenon, 2015. http://www.tenon.io/documentation/. [retrieved: 03, 2016]

[8] WAVE API. WebAIM, 2015. http://wave.webaim.org/api/. [retrieved: 03, 2016]

[9] Evaluation and Report Language (EARL) 1.0. World Wide Web Consortium, 2011.

[10] RDF 1.1 Turtle. World Wide Web Consortium, 2014.

[11] Semantic Web. World Wide Web Consortium, 2015. http://www.w3.org/standards/semanticWeb/. [retrieved: 03, 2016]

[12] SPARQL 1.1 Query Language. World Wide Web Consortium, 2011.

[13] Reasoners and rule engines: Jena inference support. Apache Software Foundation, 2015. https://jena.apache.org/documentation/inference/. [retrieved: 03, 2016]

[14] Automated WCAG Monitoring Community Group. World Wide Web Consortium, 2015. https://www.w3.org/community/auto-wcag/. [retrieved: 03, 2016]

[15] N. Fernandes, R. Lopes, and L. Carriço, “Architecture for Multiple Web Accessibility Evaluation Environments”. 6th Int. Conf. UAHCI 2011, July 2011, pp 206-214, ISBN: 978-3-642-21671-8.

[16] P. Nawaz. QuickCheck, 2015. https://github.com/mpnkhan/quickcheck/. [retrieved: 03, 2016]

[17] S. Otón et al. Service oriented architecture for the implementation of distributed repositories of learning objects. Int. Journal of Innovative Computing, Information and Control. Vol. 6(3A), 2010, pp. 843-854.