TEST REPORT PRESENTATION USING EARL RDF/XML NOTATION

Author: Olga CASIAN
Scientific advisor: lect. sup. Dumitru CIORBĂ
Technical University of Moldova
Email: dae.eklen@gmail.com, dumitru.ciorba@ati.utm.md

Abstract: *Validation and testing tools generate a large variety of kinds of test reports. Evaluation and Report Language (EARL) is a format that provides the possibility of storing the information about who or what performed the test, what was tested, what were rules tested against and what the outcome of the test is. Inheriting from Resource Description Framework (RDF) and from Extensible Markup Language (XML) constitutes the best option for storing data for further machine processing in platform and vendor independent way. This article describes the implementation of an EARL exporting component of the accessibility validation tool AChecker.*

Index Terms: *EARL, RDF/XML, validation, test report, machine-readable.*

1. Introduction

Validation and testing tools generate a large variety of kinds of test reports – from text-based styles to graphical or icon-base styles that are used to highlight validation issues in web content. There is no universal solution for a report format that will suit everyone, the best choice should be made taking into account the target audience, user requirements and skills.

AChecker, the tool described in this article, uses Evaluation and Report Export Language as one of export options for representing report data for further machine processing.

Evaluation and Report Export Language (EARL) is a vocabulary for expressing various test and validation results in a machine-readable format. The EARL vocabulary is used to describe information about who or what runs the test, the test cases, test criteria and the result of the test. Moreover, the format provides the possibility to record these and others elements in semantically rich testing reports that can contain detailed descriptions, links to resources, the exact error position, the date and time of evaluation and the test result.

In order to avoid reinventing already existing metadata definitions, EARL was built on top of already existing ones, having inherited the normative namespaces and pointers from the Resource Description Framework (RDF) and combined them with the presentation in the Extensible Markup Language (XML).

The latest version of EARL is 1.0 (released on 10 May 2011). It is considered to be stable despite of the fact that the World Wide Web Consortium (W3C) refers to its specification document [1] as to a Working Draft, "with expectation to become a W3C Recommendation".

Official W3C documents [1, 3] define the following main use cases of the format:

- ***Combining results from different tools.*** In quality assurance testing it often happens that different parts of the resource are tested by different tools (some of them may require human intervention) or the whole resource is tested by more than one tool. In this case the total report can be combined from partial reports using the EARL format.
- ***Exchanging results between tools.*** In this case EARL provides the ability to integrate various validators in different tools, such as content management systems providing different report types for different audiences.
- ***Benchmarking of testing tools.*** EARL may be used to describe the difference between two test reports, provided, for example, by accessibility evaluation tools.
- ***Evaluating dynamic and multilingual websites.*** The EARL vocabulary allows describing different web resources, including any parts or entire HTTP requests between a client and a server. This is useful in recording HTTP headers without taking into account the actual content. User interaction with the website can be recorded as well in order to describe the particular context of the test execution.
- ***Analyzing test reports.*** RDF facilitates advanced data mining by semantic inference, which can be used for gaining a high-level view on customized reports, so that project managers and developers can detect

and fix bugs. This is exactly the use case of the tool described in this article. The tool provides the possibility of exporting the results of accessibility tests to the EARL format.

2. Structure of EARL

RDF can be serialized in many equivalent ways; thus, the syntax is not limited to RDF/XML representation and makes certain plain text forms possible, as well as the Notation3 form [2]. However, RDF/XML is recommended by EARL specification.

The core construct of EARL is Assertion, which describes the resources relevant to test reporting and contains the following sections [3]:

- ***Assertor*** – contains information about who or what ran the test (ex.: automated validators, human evaluators or combinations of these);
- ***Test Subject*** – describes the things being tested (ex.: webpage, applet, image, software, etc.);
- ***Test Criterion*** – represents the specification (a set of guidelines or some other testable statements against which the evaluation is performed);
- ***Test Result*** – the outcome of the test that contains contextual information relevant to the test subject in both machine-readable and human-readable forms.

3. Implementation details

The EARL report export system presented in this article was a part of the "Google Summer of Code 2011" project for AChecker, the open source web accessibility evaluation tool, which can be used to review accessibility of web content conforming to different guidelines (WCAG 1.0, WCAG 2.0, BITV 1.0, Section 508, Stanca Act). AChecker is different from other accessibility tools in the fact that it provides the possibility of manual checks, making the evaluation process interactive by categorizing problems. In this way it avoids the situations when a fully automated tool cannot identify potential problems. Another important feature is that AChecker allows users to create custom accessibility guidelines against which to validate, thus dealing with particular environments that require a specific level of accessibility.
Moreover, AChecker is an open system which means that one can have full access to all the inner workings of the system, including guidelines, checks and solutions for fixing problems [4].

The goal of the EARL report system was to provide the following information:

- ***Assertor*** – who performed the test; can be a *foaf:Group* class containing in the *earl:mainAssertor* subclass the details about the validation tool, and in the *foaf:member* subclass the details about the logged in user; or, for anonymous validation, an *earl:Software* class that only includes information about AChecker.
- ***Test Subject*** – what was tested; the tool can work with web resources, files from the local disk or just a fragment of code; this information needs to be reflected in *earl:TestSubject* class in the given section.
- ***Test Criterion*** – what accessibility guidelines were selected as test criteria; can contain one or more *earl:TestRequirement* classes, each of them describing the guideline selected by the user to test against.
- ***Test Result*** – the outcome of the performed test; contain at least one *earl:TestResult* class and one *ptr:ExpressionPointer* class, where the former represents a list of pointers to the second class. In the case when the test passed successfully, the *earl:TestResult* class contains a single pointer to a *ptr:ExpressionPointer* that notifies of success. If the test failed, each *earl:TestResult* includes six pointers to six *ptr:ExpressionPointer* classes that contain the details about the error, such as the exact location, the error description, the corresponding HTML and CSS code, image source (if it exists) and the user's decision about the error.

Currently, there is not a single EARL library for any programming language, and the only examples of implementation could be found in the documents describing the vocabulary. This makes understanding the vocabulary a more challenging process. However, the Evaluation and Repair Tools Working Group (ERT WG) highly encourages feedback from developers and researchers who have interest in Semantic Web technologies about the standard. This makes the process of vocabulary development faster and more effective.

Taking into account the rules defined above and the W3C EARL schema [3], the following minimal example represents the case when an anonymous user tested a web resource that has URL http://atutor.ca/ against the WCAG 2.0 guidelines, level of conformance AA, and the outcome showed that no known errors were found:

```
<rdf:RDF
         xmlns:earl="http://www.w3.org/ns/earl#"
         xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">

         <!-- Assertor -->
        <foaf:Group rdf:ID="assertor01">
            <dct:title>FirstName LastName (Username) and AChecker – Web Accessibility
Checker</dct:title>
            <dct:hasVersion>1.2</dct:hasVersion>
            <dct:description xml:lang="en">
                AChecker is an open source Web accessibility validation tool.
            </dct:description>
            <earl:mainAssertor rdf:resource="http://atutor.ca/achecker/"/>
            <foaf:member>
                <foaf:Person>
                    <foaf:mbox rdf:resource="mailto:example@domain.com"/>
                    <foaf:name> FirstName LastName (Username)</foaf:name>
                </foaf:Person>
            </foaf:member>
        </foaf:Group>


        <!-- Test Subject -->
        <earl:TestSubject rdf:about="http://atutor.ca">
            <dct:title xml:lang="en">ATutor Learning Management System</dct:title>
            <dct:date>2011-11-10</dct:date>
        </earl:TestSubject>

        <!-- Test Criterion -->
        <earl:TestRequirement rdf:about="http://www.w3.org/TR/WCAG20/#a">
            <dct:title xml:lang="en">WCAG2-AA</dct:title>
            <dct:description xml:lang="en">Web Content Accessibility Guidelines (WCAG),
Version 2.0, Level AA</dct:description>
        </earl:TestRequirement>

        <!-- Test Result -->
        <earl:TestResult rdf:ID="result_known1">
            <earl:pointer rdf:resource="#pointer_known1_message" />
            <earl:outcome rdf:resource="http://www.w3.org/ns/earl#passed" />
        </earl:TestResult>

        <ptr:ExpressionPointer rdf:ID="pointer_known1_message">
            <ptr:expression rdf:parseType="Literal" xml:lang="en">
                Congratulations! No known problems.
            </ptr:expression>
        </ptr:ExpressionPointer>
</rdf:RDF>
```

Being similar to XML presentation and to RDF namespaces, the format has the following root element that should present in any report:

```
<rdf:RDF
         xmlns:earl="http://www.w3.org/ns/earl#"
         xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">

    <!-- here goes report -->

</rdf:RDF>
```

The *earl:Group* class from the Assertor component has self-describing *dct:title*, *dct:hasVersion* and *dct:description* subclasses that include information about the tool that performed the test. The *earl:mainAssertor* contains obligatory reference to the resource. The *foaf:member* subclass contains information about the person who performed the test. The Test Subject component contains a *dct:title* subclass for the title of the tested resource and a *dct:date* subclass for the date when the test was performed.

The Test Criterion component has the following subclasses: *dct:title* for the acronym and *dct:description* for the detailed description of the guidelines. Because in given example outcome is positive, there is only one *earl:TestResult* class and one *ptr:ExpressionPointer* class in the Test Result component. The *earl:TestResult* class contains an *earl:pointer* subclass that links to the *ptr:ExpressionPointer* and an *earl:outcome* subclass that indicates if the test failed or succeeded by referencing to EARL specification. The *ptr:ExpressionPointer* class has a *ptr:expression* subclass that, in this case, contains a congratulation message.

4. Conclusion

Considering the alternatives, EARL allows processing test results for quality assurance, validation, etc. purposes in a platform and vendor independent way, and provides resulting the test report as a set of machine-processable statements. Describing the report result using the RDF/XML representation provided by the format has a great advantage over the usual XML representation because it is better standardized and encourages introduction of as much information about the testing process as possible.

However, it is important to consider some limitations. Being a text-based format, using the EARL format may cause security and privacy issues for resources which are not protected enough. The test report can contain information from restricted web pages, the internal directory structure of the server or even passwords. Another aspect is the official status of Working Draft which still implies a certain degree of uncertainty before the describing document will have become a W3C Recommendation. ERT WG currently encourages interested developers to provide feedback about [1] and [3], and, particularly, about the semantics they would like to see added or removed from the format.

References

1. Carlos A Velasco, Shadi Abou-Zahra, *Developer Guide for Evaluation and Report Language (EARL) 1.0*, http://www.w3.org/TR/EARL10-Guide/ , 2011.
2. Daniel Dardailler, Sean B. Palmer, *Evaluation and Report Language*, http://www.w3.org/2001/03/earl/, 2001.
3. Shadi Abou-Zahra, *Evaluation and Report Language (EARL) 1.0 Schema,* http://www.w3.org/TR/EARL10-Schema/, 2011.
4. *AChecker community,* http://atutor.ca/achecker/, 2011.