Test Case Description Language (TCDL): Test Case Metadata for Conformance Evaluation

Christophe Strobbe[1], Sandor Herramhof[2], Evangelos Vlachogiannis[3] and Carlos A. Velasco[4]

[1] Katholieke Universiteit Leuven, Kasteelpark Arenberg 10, B3001 Heverlee-Leuven (Belgium)
christophe.strobbe@esat.kuleuven.be
[2] University of Linz "integriert studieren - integrated study" (i3s3), Altenbergerstrasse 69, A-4040 Linz (Austria)
Sandor.Herramhof@jku.at
[3] University of the Aegean, Voulgaraktonou 30, GR11472 Exarchia, Athens (Greece)
evlach@aegean.gr
[4] Fraunhofer-Institut für Angewandte Informationstechnik (FIT), Schloss Birlinghoven, D53757 Sankt Augustin (Germany)
Carlos.Velasco@fit.fraunhofer.de

Abstract. Automatic benchmarking of evaluation and repair tools (ERT) has been recently the subject of several studies as there is a growing interest because of legal and commercial issues on Web compliance with different criteria and standards. This paper addresses the development of a description language targeted to formally represent test case metadata. This language was used to develop a WCAG 2.0 test suite that will support the benchmarking of ERT with regard to the aforementioned W3C recommendation.

Introduction

Since the publication of WCAG 1.0 [4], many evaluation and repair tools (ERT) have been produced to check individual web pages or complete web sites against accessibility guidelines. Each tool has its own features and strengths, but comparing these tools is a complex and time-consuming task. There have been several efforts, for example by Brajnik [1], Melody Ivory [10,11,12] and others. Evaluations of ERT software often rely on comparing evaluation reports of existing web sites. Yvory and Chevalier [11] used sections of existing web sites, asked experienced developers to modify the web pages with the help of one or more tools, and finally asked users to perform certain information-seeking tasks on the original and modified sites. These websites are not available for other researcher to repeat the study with other tools. Brajnik [1] used samples from existing websites which are also unavailable for others to repeat the research. However, Brajnik claims that his method should lead to the same results with different web pages. Nevertheless, these evaluation methods introduce an additional unknown value in an equation where the only unknown value should be the ERT software. This unknown value is the combination of (1) which accessibility guidelines are violated, and (2) how often each of these guidelines is vio-

lated. Brajnik [1] uses the union of the sets of violations detected by the tools as a basis for evaluating the completeness of each tool. The disadvantage of this approach is that the false negatives produced by the tools distort the evaluation: the assessment of completeness can only be reliable if the number of false negatives that the tools have in common is low. It is therefore necessary to use a suite of test files for which each violation of an accessibility guideline is known and documented. The evaluation report of an ERT product can then be compared with the test suite documentation to determine which guidelines are covered by the product, which guidelines trigger false positives, etcetera. The test suite also makes the evaluation repeatable, because, unlike web sites, a test suite does not change unexpectedly.

The European project BenToWeb[1] has a goal to create test suites for the upcoming Web Content Accessibility Guidelines (WCAG) 2.0. In this context, a test suite is a collection of test cases; each test cases maps to a WCAG 2.0 success criterion and consists of one or more test files and a metadata file. The metadata are encoded in an XML vocabulary specially created for this purpose: Test Case Description Language (TCDL). This Test Case Description Language should not be confused with the TCDL specification submitted to the W3C by IBM in 2003[2]: IBM's TCDL was designed to describe a set of test materials for a formal specification, while BenToWeb's TCDL describes scenarios and other metadata for individual test cases.

TCDL: Purpose and Description

TCDL serves two purposes. First, it allows test suite developers to save the metadata that are necessary for testing accessibility evaluation tools. Each test cases maps to a specific "rule", for example a WCAG 2.0 success criterion or WCAG 1.0 checkpoint, and either passes or fails that rule at specific locations in the code. These metadata can then be compared to the output of an accessibility evaluation tool to check if the tool covers the "rules" defined in the test cases, and to check for false positives and false negatives. Second, TCDL supports the definition of test scenarios, when needed, to validate the test cases during the development process. Each test case needs to be reviewed for obvious quality assurance reasons, but it is also possible to define scenarios for end-user testing.

TCDL is defined in W3C XML Schema [3] and uses the namespace http://bentoweb.org/refs/TCDL1.1; the XML Schema is available at http://bentoweb.org/refs/schemas/tcdl1.1.xsd and contains detailed documentation.

Each TCDL file has a unique identifier and consists of five major sections: formal metadata, "technologies", the "test case" (test files and test scenarios), "rules" (data related to the relevant guidelines), and namespace mappings.

---

[1] http://www.bentoweb.org/

[2] http://www.w3.org/TR/2005/NOTE-test-metadata-20050914/

[3] XML Schema is defined at http://www.w3.org/TR/2004/REC-xmlschema-1-20041028/ and http://www.w3.org/TR/2004/REC-xmlschema-2-20041028/.

Formal Metadata

Formal metadata are mostly "administrative" information (as opposed to semantic metadata). They consist of a title, a description, creation date and author, rights (usually copyright), status information, and data about sources from which materials were borrowed. Many of these data are optional.

The title contains merely a short title that allows a quick identification of the test case; it is not a unique identifier for machine processing. The description is a summary of the test files and how they are to be used in the test case. It describes to the evaluator or the developer of test scenarios what to expect, and, especially when the test files require user interaction, for example through forms, what will happen when interacting with the file. It can be very simple, for example: "A document with an `img` element that displays a cat. The `img` element does not have an `alt` attribute." It can also be more complex, for example when describing form interaction: "A page with a form that has two `fieldsets`. The first `fieldset` contains a group of radio buttons, and the user is required to make a choice (there is no default). Instructions above the form explain that required fields have labels in red. The red text for the first `fieldset` is created by means of CSS and does not include a textual or character cue to tell the user that the field is required, so it is very difficult for users of current screen readers to find out which form fields are required. The form relies on the user's ability to recognize red text."

For language, creator and rights related to the test materials in the test case, TCDL reuses Dublin Core metadata elements (dc:lang, dc:creator and dc:rights, respectively).

The status of the test case can be tracked as the test case moves from "draft" through evaluation to "accepted". Each test case starts out as a draft and is then reviewed by another accessibility or HCI expert. If any issues are found, the status changes to "pending bugfix" and the test case is sent back to the test case author. The test case author fixes the issue and sets the status back to "draft". It is also possible that the test case contains scenarios for end-user evaluation (explained below); when these scenarios are finalised, the status is set to "accepted for end user evaluation". A test case evaluation framework can select these test cases and present them to end users. (BenToWeb's test case evaluation framework is described by Herramhof et al [7].) After evaluation and when all data are definitive, the status is set to "accepted".

If test materials are borrowed — for example from existing test suites — the source, rights, and any other comments can be recorded in an optional section called "source".

Finally, there is also an optional "version" field, which is not used in BenToWeb, but which may of value to other users of TCDL.

Technologies

TCDL uses the term "technology" in the same sense as WCAG 2.0: "a markup language, programming language, style sheet, data format, or API"[4]. Each test case implements a specific technology feature, for example an HTML element such as `img`, that is relevant to the "rule" for which the test case is defined. TCDL provides the ability to store the following information about the technologies in a test case:

- the formal specification or recommendation that defines the technology (by name and URL; mandatory);
- the technology features (for example, which XHTML elements and/or attributes) that are relevant to the test (optional);
- a reference (URL) to a section in the formal specification, or a quote from the formal specification that describes the technology features that are relevant to the test (optional).

Test Case

The "test case" section consists of four subsections: purpose, required tests, test files and preconditions. The preconditions are an optional section.

Purpose

The "purpose" is a mandatory section that provides a brief explanation why the test was developed. It explains the expected evaluation result ("pass" or "fail") in regard to the relevant "rule" (WCAG checkpoint or success criterion). It is not meant to repeat the "rule", but in complex test cases, it can state that certain aspects of the test file(s) are not meant to be tested. For example, in a test of text alternatives for image map areas, the purpose can state that only the alternative text should be evaluated, and not the colour contrast of the image itself.

Required Tests

This section describes what type of evaluation is necessary to move the test case from "draft" to "accepted". A test case can be evaluated in one or more ways: by end users, one accessibility expert, a group of accessibility experts, or automated tools. These options are called "test modes" in TCDL; the definition of one or more test modes is mandatory.

Depending on the test mode, it is also possible to define test scenarios. Defining test scenarios is optional, except if the test mode is end-user evaluation. A test scenario consists of the following parts:

- user guidance (for example, advice on features of a user agent that should be supported by and enabled in a browser);
- a question that the user is expected to answer (yes/no question, open question, Likert scale, multiple choice, or a yes/no/other question);

---

[4] http://www.w3.org/WAI/GL/WCAG20/WD-WCAG20-20060407/appendixA.html#technologydef

- the experience that the user should have with certain user agents or assistive technologies to evaluate the test, and;
- optionally, disabilities to which the test is relevant.

The third part, experience, defines the types of user agents, assistive technologies and devices with which the test files should be evaluated. User agents are usually browsers and browser plug-ins; devices are usually PCs, but PDAs and cell phones could also be relevant. For each of these softwares or platforms, it is possible to define the minimal experience level that a user should have with them. A test case evaluation framework can then match these software, platforms and experience levels to user profile data and present relevant scenarios to the end user.

For each category (user agent, assistive technology, device) it is possible to specify only the "type" (for example, browser, screen reader, magnification software, speech recognition software), but it is also possible to specify the products and (minimal) product version with which the test case should be evaluated. If several products of the same "type" are specified (for example, the browsers Internet Explorer 6.0, Firefox 1.0 and Opera 7.54), these are in an OR relationship. The different categories, however, have an AND relationship. For example, if a test case requires a browser (for example, one of Internet Explorer 6.0, or Firefox 1.5) and a screen reader (for example, one of JAWS 7.0 or Window-Eyes 5.5), the test case will be presented to users with both Internet Explorer 6.0 and JAWS 7.0, or with both Internet Explorer 6.0 and Window-Eyes 5.5, or with both Firefox 1.5 and Window-Eyes 5.5, etcetera, but not to users who have a older version of these products, or to users who have SuperNova (as a screen reader), or Safari (as a browser) or to users who don't have a screen reader. If the scenario also specifies disabilities; the test case evaluation framework can use this information as additional criteria when matching test scenarios to user profiles.

When a user is presented with a test that matches their setup, it may be necessary to provide extra guidance about certain features that need to be enabled for the evaluation to be useful. Many browsers and assistive technologies can be customised, and some tests may rely on features that the user happens to have disabled. For example, not all users have JAWS set up so that it reads the expansions of acronyms.

Most importantly, the user is asked to interact with the test file (or files) and to answer a question. TCDL supports several types of questions: simple yes/no questions, Likert scales, open-ended questions, yes/no questions with a third option for the user to fill in, and multiple-choice questions. The test case evaluation framework transforms this question into an HTML form; the answer is stored in a database or other persistence mechanism for later analysis.

Test Files

A test case has one or more test files. For most "rules" or WCAG 2.0 success criteria, one file is sufficient. However, certain success criteria only apply to sets of files, for example, success criteria about consistency of navigational mechanisms or about information about a user's location in a website. Usually, it is sufficient to provide only the URL of the test file. In some cases, however, it is useful to test server responses that result from certain specific values in HTTP request headers or the HTTP request body. For this reason, TCDL also supports the definition of specific mediatypes, enctypes, HTTP request headers and HTTP request body. If more than one test file is specified, it is also possible to define if the sequence in which the files are navigated

is meaningful. Supporting files such as included images, external style sheets and dummy pages to which forms are submitted (if only the form is relevant to the test) etcetera, are not documented.

Previous research [9] shows that test cases with multiple files appear to be something new in test suites for Web standards.

Preconditions

Preconditions are "conditions that must be met before the test can be successfully executed" [6]. This kind of conditions is used in the current HTML test suite for WCAG 2.0[5], where they are called "prerequisite tests". BenToWeb does not define preconditions because the BenToWeb test suite relies on a different mapping mechanism than the current HTML test suite for WCAG 2.0: test cases map directly to WCAG 2.0 success criteria, instead of through the intermediate step of techniques.

Rules

TCDL uses "rules" as a generic term for WCAG 1.0 checkpoints, WCAG 2.0 success criteria, BITV provisions, and similar accessibility requirements. The "rules" section defines the mapping between the test case and specific accessibility requirements. The rules do not provide a direct reference to the relevant online source because it may not always be possible to reference "rules" directly (for example, because they are not available in HTML) or because they might move to a different URL. The rules reference a rule ID in a "Rulesets XML" file (see Rulesets XML RDDL document[6]), which in turn references the relevant rule in an online source.

For each rule listed in this section, the following information can be provided:

- whether the rule is the "primary" rule of the test case or only listed for informative purposes (for example, some test cases uses invalid markup in tests that are not about validity, so it is useful to document that checkpoints about validity should not be applied to the test case) (optional);
- the identifier of the rule in "Rulesets XML" (mandatory);
- the locations in the test file that are relevant to the rule, with URL, line number, column number and/or XPath expression [5] (optional);
- whether the test case passes or fails the success criterion or rule (mandatory);
- the functional outcome of the test: a description of why the test case passes or fails, in terms that relate to a user's experience (as opposed to technical comments about source code) (mandatory);
- technical comments on the test: a technical description of why the test case passes or fails, and other technical information (for example, issues in certain browsers or assistive technologies).

[5] http://www.w3.org/WAI/GL/WCAG20/tests/

[6] http://bentoweb.org/refs/rulesets

Namespace Mappings

Finally, there is an optional section for "namespace mappings", which is important for test cases based on test files that use XML-based technologies. The technology features (see the section on Technologies above) used in a test case exist in an XML namespace [2] and should be referenceable with an XPath expresssion. The normal mechanism for mapping namespaces to URIs by declaring them with `xmlns:prefix="namespaceURI"` cannot be used for this, because it is necessary to be able to define empty prefixes and to extract the namespace mappings with a common XML API.

Differences with IBM's TCDL Submission to W3C QA

BenToWeb's TCDL was developed completely independently from the TCDL proposal submitted by IBM [8] to the W3C Quality Assurance Working Group (QA WG) in 2003. The goal of IBM's TCDL is to catalogue most of the test materials that a W3C Working Group would provide. It is intended to be used in a test lab "to set up and run the test materials against one or more test subjects" and "supports automated setup of the system(s) used for testing, automated running of test cases, automated comparison of results, and automated cleanup of the system(s)". Like BenToWeb's TCDL, IBM's TCDL is an XML vocabulary, so metadata can be transformed into other formats for human or machine consumption.

An IBM TCDL catalogue contains a list a test cases, whereas a BenToWeb TCDL contains metadata for an individual test case only. BenToWeb's TCDL is also different because each test case references at least two specifications (WCAG 2.0 or another ruleset, and a specification of a technology such as XHTML or CSS), and one of these (WCAG 2.0) cannot be tested fully automatically.

Unfortunately, it is not possible to make a detailed comparison of the two languages because the XML Schema for IBM's TCDL submission is not available.

Conclusions and Future Work

Even though the BenToWeb consortium is not the first organisation that develops accessibility test suites, the Test Case Description Language appears to have no predecessors that fulfil the same requirements. TCDL has undergone almost a year's implementation experience through the creation of test case descriptions, a desktop TCDL editor, transformations into XHTML, and implementation in the test case evaluation framework (see Herramhof et al [7]). TCDL implements the test metadata defined by the W3C's Quality Assurance Working Group [6] in order to support use cases outside BenToWeb. Through informal contacts, the W3C's ERT Working Group has expressed interest in this language, and there are plans to submit to it to the W3C.

Acknowledgements

This work has been undertaken in the framework of the project BenToWeb — IST-2-004275-STP — funded by the IST Programme of the European Commission.

References

1. Brajnik, G (2004). Comparing accessibility evaluation tools: a method for tool effectiveness. Univ Access Inf Soc 3: pp. 252-263. DOI: 10.1007/s10209-004-0105-y
2. Bray, T, Hollander, D, Layman, A (1999). Namespaces in XML — World Wide Web Consortium 14-January-1999. Available at: http://www.w3.org/TR/REC-xml-names/
3. Caldwell, B, Chisholm, W, Slatin, J, Vanderheiden, G, White, J (eds) (2005). Web Content Accessibility Guidelines 2.0, W3C Working Draft 30 June 2005. World Wide Web Consortium. Available at: http://www.w3.org/TR/WCAG20/
4. Chisholm, W, Vanderheiden, G, and Jacobs, I (eds) (1999). Web Content Accessibility Guidelines 1.0, W3C Recommendation 5-May-1999. World Wide Web Consortium. Available at: http://www.w3.org/TR/WCAG10/
5. Clark, J, DeRose, S (1999). XML Path Language (XPath) 1.0 — W3C Recommendation 16 November 1999. Available at: http://www.w3.org/TR/xpath
6. Curran, P, Dubost, K (2005). Test Metadata — W3C Working Group Note 14 September 2005. Available at: http://www.w3.org/TR/2005/NOTE-test-metadata-20050914/
7. Herramhof, S, Petrie, H, Strobbe, S, Vlachogiannis, E, Weimann, K, Weber, G, Velasco C A (2006). Test Case Management Tools for Accessibility Testing. Computers Helping People with Special Needs. 10th International Conference, ICCHP 2006, Linz, Austria, 12-14 July 2006, Proceedings.
8. Marston, D (2003). Test Case Description Language 1.0 — Submisson to QA Working Group 12 October 2003. Available at: http://www.w3.org/QA/WG/2003/10/tcdl-20031012.html
9. Strobbe, C (2005). Test-suites' State of the Art and Quality Assurance Methods for W3C Recommendations. BenToWeb deliverable D 4.1. Available at: http://www.bentoweb.org/html/BenToWeb_D4.1.html.
10. Ivory, M Y, Sinh, R R, and Hearst, M A (2001). Empirically validated web page design metrics. Proceedings of the Conference on Human Factors in Computing Systems (Seattle, WA, March), pp. 53—60. New York, NY: ACM Press.
11. Ivory, M Y, and Chevalier, A (2002). A Study of Automated Web Site Evaluation Tools. Technical Report UW-CSE-02-10-01. University of Washington, Department of Computer Science and Engineering. http://ubit.ischool.washington.edu/pubs/tr02/toolstudy.pdf.
12. Ivory-Ndiaye, M Y (2003). An Empirical Approach to Automated Web Site Evaluation. Journal of Digital Information Management, 1(2), June 2003, pp. 75—102.