An Environment for Defining and Handling Guidelines for the Web

Barbara Leporini, Fabio Paternò, Antonio Scorcia

ISTI – CNR
56124 Pisa, Italy
{barbara.leporini, fabio.paterno, antonio.scorcia}@isti.cnr.it

Abstract
Several accessibility and usability guidelines are proposed more and more. In this paper we present an environment for defining, handling and checking guidelines for the Web. The goal of such a tool is to support developers and evaluators in flexibly handling multiple sets of guidelines, which can be dynamically considered in the evaluation process. In particular, an interactive editor has been designed to assist the evaluators in abstracting and specifying new and existing guidelines in our XML-based Guideline Abstraction Language (GAL), which are then stored in external files. Our tool is able to check any guidelines specified in this language without requiring changes in its implementation.

Keywords: guidelines, editor, accessibility, usability, abstraction language

1. Introduction

Design guidelines are increasingly proposed in order to improve and make consistent user interfaces, in particular for better supporting accessibility and usability. Indeed, several guidelines have been proposed in literature for general user interfaces (e.g., [5], [6] and [7]) as well as for Web interfaces (e.g. [4], [8] and [9]). Thus, in order to assure a certain level of accessibility and usability, developers have to orientate themselves among several sets of guidelines. To this end, we have developed a tool, MAGENTA (Multi-Analysis of guidelines by an Enhanced Tool for Accessibility), which supports various sets of guidelines. One of its mean features is to be independent of the guidelines specification. This means that if the guidelines are expressed by an abstract language that we propose and are stored in external files then the tool can check any of them without modifications in its implementation. In this way, it is possible to:

- Specify any kind of guideline based on X/HTML tags or CSS attributes;
- Change any existing guideline simply by handling the corresponding XML-based specification;
- Group and separate several guidelines into various sets.

In our first experiences with the tool, we soon realised that if files are handled manually by developers or evaluators then many problems could occur:

- File editing should be made manually by an expert and it can be rather tedious;
- Editing could introduce some syntactical errors, unless the XML file is not parsed by an XML validator;
- Adding new guidelines as well as changing existing ones could require a lot of time and effort;
- The experts who handle the XML file and guidelines should know tags, attributes and properties very well.

For all these reasons, we have extended our tool with a graphical environment thought for assisting developers and evaluators in proposing, specifying and modifying guidelines. In this paper we present an environment, which includes a Guideline Editor (GE) developed for assisting developers in defining new guidelines and in handling those already existing (such as those for accessibility and usability for the Web). In particular, such an editor has been integrated into the tool MAGENTA in order to improve its flexibility.

The paper is structured as follows: firstly we introduce a short description of the tool MAGENTA; next, the Guideline Abstraction Language (GAL) is described. In section 4, an interactive editor for defining guidelines according to the GAL is proposed and discussed.

2. The MAGENTA Tool

The MAGENTA tool has been developed with the intent to check whether a Web site is accessible and usable and provide support to improve it. To this end, the tool checks how satisfactorily the selected guidelines are applied to the indicated Web pages. This is obtained through automatic identification of the checkpoints associated with each guideline and analysis of the associated constructs and attributes to check whether they provide the necessary information. Currently, MAGENTA supports three sets of guidelines for the Web: a set of guidelines for visually-impaired users [2], WCAG 1.0 [8], and the guidelines associated with the Italian accessibility law [1]. Firstly, the guideline set to be considered can be selected. Then, through a list of checkboxes, the evaluator can decide which guidelines of the selected set(s) have to be checked.

The tool is not limited to checking whether the guidelines are supported but, in case of failure, it also provides support for modifying the implementation in order to make the resulting Web site more usable and accessible. Thus, when an error is detected, the tool points out what parts of the code are problematic and provides support for corrections indicating what elements have to be modified or added. The process is not completely automatic because in some cases the tool requires designers to provide some information that cannot be generated automatically. The tool indicates the parts that must be corrected (i.e. tags or attributes) and developers write those parts or values that can not be added automatically (e.g. specification of proper link contents or proper name for frames).

MAGENTA has been developed considering the limitations of most current tools, in which the guidelines supported are specified in the tool implementation, with all the associated issues mentioned in the introduction (e.g., having to change the tool implementation each time a guideline is modified or added). In our work we aim to provide a tool (MAGENTA) independent of the guidelines to check. Practically, the tool is able to deal with guidelines which are defined externally from the tool. The guidelines are specified in an XML file in a specific abstract language whose basic constructs can be interpreted by our tool at run-time. In this way, the tool is able to adapt the checking procedure to various guideline sets as well as new set of guidelines, which are specified by the evaluators and it does not require to be modified and recompiled. This opens up many interesting opportunities to developers and evaluators who can indicate flexibly what guidelines they want to check even if they do not belong to those already existing and available. However, as shortly described in the introduction, handling manually XML-based guideline specification files could generate several difficulties and errors. Thus, we decided to extend the guideline definition support by adding a new module: the Guideline Editor, which has been added to the MAGENTA functionalities in order to improve its flexibility.

3. The Guideline Abstraction Language

Our solution is based on the definition of a language for specifying guidelines that are stored externally to the tool. In general, a guideline, is defined as a rule or principle that provides guidance to appropriate behaviour. Generally speaking, a guideline is expressed in natural language, which automatic tools cannot handle. Hence, in order to support developers in using and customizing various sets of guidelines, our approach is aimed at expressing guidelines so that they can be dealt with automatically. In practice, the solution consists of abstracting guideline statements in a well-defined language. For example, a statement such as “all images in a Web page must have an alternative description” or “Tables should have a short summary description” have to be formalised in some manner that can be handled by the automatic tool. Thus, we require that the guideline statement be abstracted and appropriately specified. For the guidelines specification we use an XML-based language. This implies that the guideline is previously structured and then specified in our XML-based language.

In defining a guideline abstraction, the main features and elements characterizing the guideline itself must be defined. Hence, general features, objects involved in the checking process, and conditions referred to objects must be specified through the XML-based Guideline Abstraction Language (GAL) that we propose. For example, a guideline such as ”All links must have clear and significant content” could be abstracted as follows:

<a> with href, XHTML, (content not_belong not_proper_list_values)

and

<a> with href, XHTML, (for <img> content: Exist alt & (alt not_belong not_proper_list_values | alt+content not_belong not_proper_list_values))

In this case, checking involves both textual and graphical links, i.e. the criterion has two checkpoints: (1) textual and (2) graphical links. In the former case the content should not belong to a list of non appropriate terms, in the latter case the alt attribute should exist with values not belonging to inappropriate terms.

In brief, each guideline expresses a principle to be complied with by applying one or more conditions at checkpoints. All this information and these conditions have to be structured by a specific guideline abstraction. More details can be found in [3].

Summarising, we define a guideline in terms of a number of elements: checkpoints expressed with objects, operations (such as Exist, Count, Check, Execute) and conditions to be verified. In specific, an XML-Schema [10] has been defined in order to define the general structure of our Guideline Abstraction Language (GAL). The next table reports an excerpt of an example of a guideline expressed with XML tags according to the proposed language for abstracting guideline definition.

```
….
      <name> Italian accessibility law</name>
              ...
      <criterion type="accessibility" target="page">
      <description>Links should have an appropriate content</description>
      <checkpoints rel="and">
      <cp id="19.1" summary="Textual links" priority="1">
        <cp_descript>...</cp_descript>
        <eval_object code="html" mandatory="yes">
          <object type="tag">a</object>
          <select type="attrib">href</select>
        </eval_object>
        <conditions rel="and">
        <evaluate operator="check" cond="not_belong" iderr="19.1.1">
          <el type="tag">a</el>
          <el type="file">"nolinks.dic"</el>
        </evaluate></conditions>
        </cp>
      <cp id="19.2" summary="Graphical links" priority="1">
        <cp_descript>...</cp_descript>
        <eval_object code="html" mandatory="no">
          <object type="tag">a</object>
          <select type="attrib">href</select>
          <select type="tag">img</select>
        </eval_object>
      <conditions rel="and">
      ...
      </guideline></gdl_set>
```

Table 1 - Excerpt of Example Guideline Specification

4. The Guideline Editor

When a guideline must be defined so that a tool can deal with it automatically, the problem is to transform its description in natural language into technical terms. To this end, we have defined a XML abstract language for describing guidelines. The abstraction process is not easy and requires some effort when implementation languages for the Web are considered. Furthermore, according to the proposed GAL, to define a guideline several XML tags and attributes must be specified. In order to facilitate this process, a graphical editor has been designed and added to the MAGENTA tool, thus enabling even people non particularly expert in handling languages such as X/HTML and CSS to specify the desired guidelines.

The Guideline Editor (GE) has been designed for assisting developers in handling single as well as groups of guidelines. The tool supports new guideline definition and various types of editing. Summarising, the editor offers various useful features in order to facilitate multiple guideline sets management and general utilities. In particular, the guideline editor provides support for:

- Defining or modifying single guidelines;
- Importing guidelines from various sets in order to create customised groups;
- Organising, classifying and browsing guidelines into groups;
- Handling several sets of guidelines simultaneously;
- Writing and repairing external dictionaries used with some guidelines (e.g. proper link content, proper name for frames, specific sections, and so on).

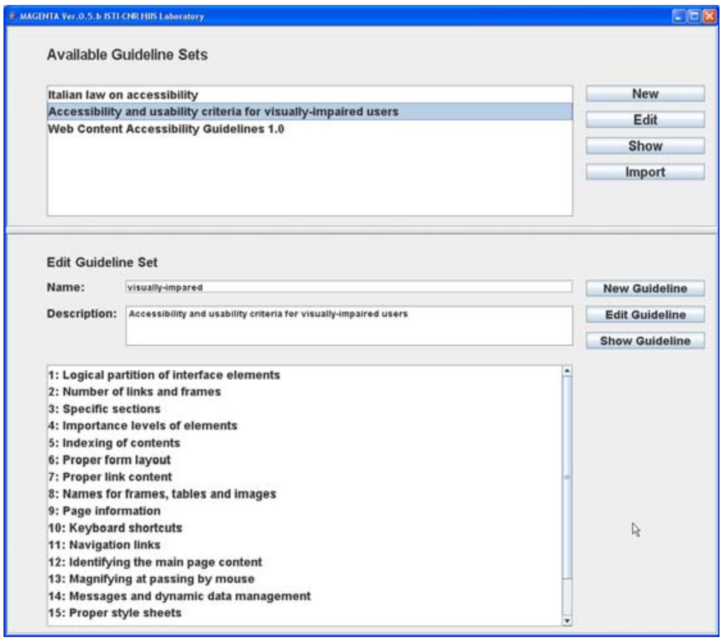
Indeed, the last feature has been thought for some kind of guidelines that need some support in identifying appropriate or not appropriate terms. For instance, if a guideline states “frames should have the attributes ’title’ appropriate”, then some not appropriate values could be stored in an external file called “dictionary”. Such dictionary is used during the evaluation process in order to check that frames have a title attribute which does not belong to the non-appropriate term list. In practise, we verify that the value is not inappropriate (negative control). Other guidelines can be checked with a similar approach.

Regarding manipulation of single guidelines, the tool allows developers to add new ones or modify/delete existing ones. In practice, the graphical editor guides in specifying all the XML tags necessary for defining a guideline (see an example in Table 1). Furthermore, it is also possible to add new guidelines by importing them from other guideline set(s). This allows designers to create customised set(s) of guidelines by reusing already specified ones. More precisely, through the GE for a guideline it is possible to:

- Insert general information on the guideline (short name, description, etc.);
- Select the object to be considered in the guideline (i.e., language, tags and restriction on the object itself).
- Indicate the number of checkpoints needed and their relationship (and | or);
- Insert general information for each checkpoint, such as short name, description and the number of the conditions composing the checkpoint;

- Define the conditions to be verified, by selecting operators and conditions to be applied at a runtime (i.e. when the tool will carry out the evaluation).

The tool user interface has been designed in order to be easy for any user, including those who are not particularly expert with XML and X/HTML languages. In fact, since it is not possible to specify a guideline precisely in a natural language, then the XML elements involved in the guideline must be edited. To this end, when selecting a specific object (tag, attribute or property) a short description is given to the developers. In practise, the tool guides the user in editing all the elements necessary to define a guideline in a technical format. Thus, the tool assists developers in abstracting a guideline and then generates its specification into the XML external file.


MAGENTA Ver.0.5.b ISTI-CNR HIIS Laboratory

Available Guideline Sets

Italian law on accessibility
Accessibility and usability criteria for visually-impaired users
Web Content Accessibility Guidelines 1.0

Edit Guideline Set

Name: visually-impared

Description: Accessibility and usability criteria for visually-impaired users

1: Logical partition of interface elements
2: Number of links and frames
3: Specific sections
4: Importance levels of elements
5: Indexing of contents
6: Proper form layout
7: Proper link content
8: Names for frames, tables and images
9: Page information
10: Keyboard shortcuts
11: Navigation links
12: Identifying the main page content
13: Magnifying at passing by mouse
14: Messages and dynamic data management
15: Proper style sheets

Fig. 1. The interface supporting the creation of a new set of guidelines to check.

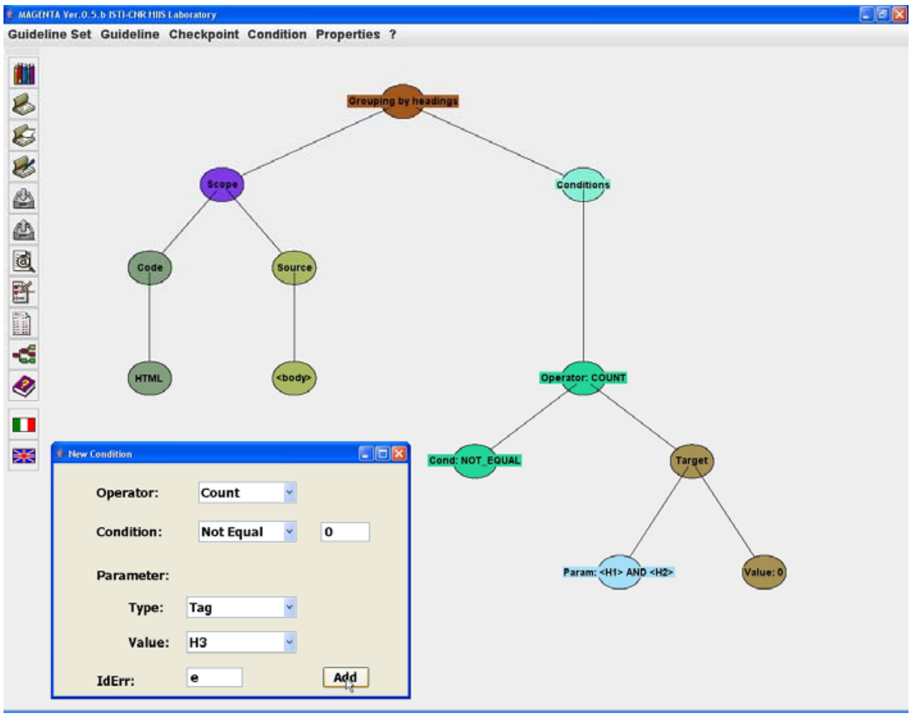
The Guideline Editor UI is mainly organized in various parts shown in cascade. Three main functionalities are available:

- *Main Guideline Management*, where all groups of guidelines can be handled (see Figure 1, top); it is possible to create a new set, modify or delete an existing one. The available guideline sets are listed; through the buttons “New”, “Edit”, “Show” and “Import” all the corresponding activities can be performed.
- *Guideline set*, which is the environment in which a new specific group of guidelines can be defined. General information of the set as well as the list of guidelines already belonging to the group are presented (see Figure 1, bottom).

- *Guideline specification*, where developers can structure and define a single guideline. Firstly, general data such as the short name and description of the current guideline as well as the main object – tag, attribute or property - involved by the guideline can be written and modified; next, all checkpoints and their conditions to be applied or evaluated can be edited through a direct manipulation graphical editor (see Figure 2).

Some elements can be selected from a list or typed by a combo-box. In practice, the elements to be selected are listed in order to make them available for the evaluator and to avoid typing errors. In this way, the user is guided during the abstraction and definition process. For instance, when an object to be checked has been selected, in the list-box for selecting the attributes related to the element, just its attributes are available. For example, if the element chosen is the tag <a>, all the attributes available in the other list-box are only those related to the tag <a> (e.g., href, tabindex, accesskey, etc.). If no restriction must be applied the “Anyone” item can be chosen.

When the user has selected all the elements which express the objects to be checked, including its properties, checkpoints and conditions, the editor saves them in the XML-based file associated to the current guideline set by inserting all the elements in the correct tags defined for each guideline.



Operator: Count

Condition: Not Equal 0

Parameter:

Type: Tag

Value: H3

IdErr: e

Fig. 2. The interface supporting the definition of a new guideline.

5. Conclusions

The increasing number of design guidelines proposed for the Web, in particular for accessibility evaluation, makes the implementation of automatic tool working with guidelines more and more complex. In order to develop a tool independent of guideline definition, guidelines should be specified separately and interpreted at runtime. To this end, we propose a Guideline Abstraction Language to define guidelines using an XML structure. These XML-based files can be managed by our MAGENTA tool, which loads and check the guidelines indicated by the developer without requiring modifications to its implementation. Thus, designers can dynamically select and edit the guidelines to check in their Web site. Since abstracting and defining guidelines requires some effort, a graphical Guideline Editor has been integrated in our environment for assisting evaluators in these activities. The user interfaces has been designed to guide developers as much as possible in order to facilitate their task. Currently, the editor supports guidelines for documents implemented in X/HTML and CSS languages, future work is planned to support other Web languages such as SMIL, XForm, and so on.

References

1. Italian Accessibility Law, 2004, Law n. 4, January 9, 2004: Provisions to support the access to information technologies for the disabled http://www.pubbliaccesso.it/normative/law_20040109_n4.htm _20040109_n4.htm
2. Leporini, B., Paternò, F. (2004). Increasing Usability when Interacting through Screen Readers, International Journal Universal Access in the Information Society (UAIS), special Issue on "Guidelines, standards, methods and processes for software accessibility", Springer Verlag, Vol.3, N.1, pp. 57-70.
3. Leporini, B., Paternò, F., Scorcia, A., (2006). Flexible Tool Support for Accessibility Evaluation. Interacting with Computers, Elsevier, to appear, 2006.
4. Mariage, C., Vanderdonckt, J., Pribeanu, C. State of the Art of Web Usability Guidelines, Chapter 41, in R.W. Proctor, K.-Ph.L. Vu (Eds), "The Handbook of Human Factors in Web Design", Lawrence Erlbaum Associates, Mahwah, 2005.
5. Mayhew D. J.. Principles and guidelines in software and user interface design. Prentice Hall, Englewood Cliffs, New Jersey, 1992.
6. Nicolle, C., Abascal J. Inclusive design guidelines for HCI, p. 285, Taylor & Francis, 2001.
7. Smith S. L. and Mosier J. N.. Guidelines for designing user interface software. Mitre Corporation Report MTR-9420, Mitre Corporation, 1986.
8. Web Accessibility Guidelines 1.0. Web Accessibility Initiative, W3C Recommendation 5-May-1999. Accessible at http://www.w3.org/WAI/GL/WCAG10/
9. Web Content Accessibility Guidelines 2.0, W3C Working Draft 17 January 2006, available at http://www.w3.org/WAI/GL/WCAG20/
10. XML Schema Part 0: Primer Second Edition. W3C Recommendation 28 October 2004 available at http://www.w3.org/TR/xmlschema-0/