Chapter 22

AUTOMATED REPAIR TOOL FOR USABILITY AND ACCESSIBILITY OF WEB SITES

Arnaud Jasselette[1], Marc Keita[1], Monique Noirhomme-Fraiture[1], Frédéric Randolet[1], Jean Vanderdonckt[2], Christian Van Brussel[2], and Donatien Grolaux[2]
[1]Institut d'Informatique, Facultés Universitaires Notre-Dame de la Paix,,
rue Grandgagnage, 21 – B-5000 Namur (Belgium)
E-mail: (aja,mke,mno,fra)@info.fundp.ac.be – Web: http://www.info.fundp.ac.be/DESTINE
[2]School of Management (IAG), Université catholique de Louvain
Place des Doyens, 1 – B-1348 Louvain-la-Neuve (Belgium)
E-mail:{vanderdonckt,vanbrussel,grolaux}@isys.ucl.ac.be – Web: http://www.isys.ucl.ac.be/bchi
Tel: +32 10 47{8525, 8391} – Fax: +32 10 478324

Abstract The need for checking both usability and accessibility of Web sites is widely recognized, approved and recommended by several official organizations. What should really be more recognized or addressed is an equal need for repairing the usability and accessibility defects that have been detected. Within the DESTINE suite, we developed a tool allowing to repair the HTML source code of a page with user interaction. Thanks to an improved version of the Guideline Definition Language (GDL), the accessibility guidelines are not hard-coded, so that our tool can deal with any existing or future standards.

Keywords: Accessibility, Automated repair tool, DESTINE, Usability engineering.

1. INTRODUCTION

In our modern world, Internet plays a prevailing role in communication, knowledge and information exchange. Internet expanded almost thanks to the World Wide Web who knew a great expansion these last years. Nowadays, the Web has to be considered as one of the most used and expanded source of information. However, this great amount of data is not always accessible to all users, especially people suffering from a handicap. The awakening of this problem gave birth to the creation of standards and guidelines to follow in the construction of a Web Site. One of the most concrete exam-

ples of this phenomenon is the e-Europe Action Plan. The e-Europe 2002 forces all public Web Sites from Europe to be accessible to handicapped people. The e-Europe 2005, which is the following action plan, aims at an improvement of quality and Usability and Accessibility (U&A) of the services to the profit of the whole of the citizens. This reviewed plan confirms the adoption of the recommendations made for the U&A to Web Sites. The U&A to the Web and his usability are part of the preoccupations of organisms such as the W3C Consortium who promotes the recommendations for Web Sites to be accessible with the Web Accessibility Initiative (WAI) recommendations and the Web Content Guidelines.

The need for checking the usability and the accessibility of Web sites is thus widely recognized, approved and recommended by several official organizations. Equally recognized, although less addressed, is the desire to support as much as possible the evaluation process through automated tools [12]. What is even less recognized or addressed is an equal need for repairing the usability and accessibility defects that have been detected during the automated evaluation phase. If such a repair process does not exist, the author of the web page could be left out with only a potentially long list of defects, which may be hard to understand and to fix. Indeed, automated evaluation typically performs a summative evaluation (it specifies the type and location of the defects), but not a formative evaluation (it does not provide any feedback on how to fix the detected defects). If the process is not completely supported by a suite of tools, the evaluation conducted before may have little or no impact on the usability and accessibility.

We have designed such an integrated tool where evaluation and repair are parts of the same system. This paper is focused on the repair part. We start with a short overview of the state of the art and on the shortcomings of repair tools in general. Then we present the DESTINE suite, in order to situate the repair tool. Afterwards we shall describe how the software corrects a web page using Guideline Definition Language (GDL) and finally explain how the repair tool interface has been designed. To conclude, we shall emphasize the strengths of the tool regarding to its rivals.

2. STATE OF THE ART

Research shows the advisability of tools that automatically evaluate and repair Web sites according to ergonomic guidelines, whatever the type of ergonomic guidelines (e.g., usability, accessibility, cognitive walkthrough). Let us cite in particular the work of [6,7] showing how it is possible and important to systematically and consistently evaluate web site against usability guidelines. Several tools have been developed to automate the evaluation

process of a web site according to various techniques. This represents an important research area since it is reported as largely underexplored [12]. They allow evaluating Web Pages from U&A guidelines and informing the evaluator about the different aspects of the site to improve.

Repair tools allow the syntactic correction of ergonomic issues according to U&A guidelines. These tools bring an automatic correction if this is possible; otherwise the user is guided through the correction process and can regulate major conflicts inside the Web page. We can distinguish two types of existing repair tools.

The **first** type can be considered as a plug-in integrated to software, such as a Web page editor. The site author may want to ensure that the site she is creating is "standard compliant". Among these tools we can find:

- Deque Ramp Ascend [9]: this software is compatible with Microsoft Frontpage and Macromedia Dreamweaver authoring environments. They allow validating a web page against the guidelines provided by both Section 508 and W3C WCAG WCAI. The software corrects problems with *n*-dimensional tables, skips navigation, performs automatic detection and correction of missing alternate tags, and corrects animation graphics so that they do not flash in the 2 to 55 Hertz range.
- Lift [15]: this desktop solution is also intended to produce usable and accessible "Section 508/W3C compliant" web sites within Macromedia Dreamweaver only and focuses on the WCAG Priority 1 and US government section 508 checkpoints. It generates HTML or XML reports from the evaluation performed.

The **second** type of tool is a stand-alone software. It can evaluate a page and may offer some support for correcting portions of a web page which have been evaluated negatively. Modifications and corrections to the document are not done by a Web page editor but by the repair tool itself. The tool can modify the code of the pointed page at any time. That kind of software is less spread than the integrated plug-in, the following list shows the most important of them:

- A-Prompt [17]: based on the WCAG Guidelines or Section 508, this tool guides the user by a completely automated correction in the different stages.
- InFocus [2]: similar to other repair tools, it also evaluates separately guidelines contained in Section 508 and W3C WCAG.
- SWAP [14]: the Semantic Web Accessibility Platform (SWAP) is developed according to the international accessibility standards of the W3C for Internet-accessibility guidelines and worldwide legislation.

There are also some other forms of software products that **combine** both aforementioned types. For Example, the tool AccRepair can be used as stand-alone software or could be integrated in authoring environments like

Microsoft FrontPage. This tool corrects errors concerning the guidelines in the Section 508 and WCAG Compliance Errors.

The main shortcomings that one could identify for the above types of software that stem for developing software for automated evaluation are equally applicable to the development of software for automated repair:

- Repair logic is hard-coded in software: all existing tools which perform some kind of automated repair have their repair algorithms completely built-in the software without any flexibility or modifiability.
- The set of guidelines is predefined: only one or two sets of guidelines are supported, typically the W3C guidelines and the Section 508. A customized set of guidelines, such as a corporate style guide, cannot be supported.
- The underlying guidelines are immutable, so are their repair: the tools only support a single interpretation of each guideline, as well as for automated evaluation as for automated repair. It is not possible to modulate the interpretation of a guideline at evaluation/repair time nor is possible to tailor, to refine this process depending on the context of use.
- Only one set at a time: It is not possible to mix guidelines from different sources in a custom way, with possible optimization of their process. No combination is allowed.

Our goal is obviously to design a software for repairing a web page (or a series of web pages) which have been evaluated negatively, which does not suffer from any of these above shortcomings.

3. REPAIR TOOLS CLASSIFICATION

Not all guidelines which have been identified as violated for one or several of their checkpoints could be repaired in an automated way. Therefore, different levels of automation could be considered:

- *Fully automated repair*: the software holds all information required to repair the violated guideline.
- *Partial automated repair*: the software does not hold all the information required to repair the guideline, but may assume some hypothesis in order to choose among options. This may give rise to a warning explaining the user why this choice has been made. This typically occurs for syntactical guidelines.
- *Partial interactive repair*: the software may know how to repair the violated guideline, but requires some interpretation from the user to choose among options. This typically arises from more semantic issues.
- *Manual repair with guidance*: the software may know how to repair the violated guideline, but this operation cannot be processed by the software

for several reasons: human decision, incomplete information, need for additional information, need for a consensus, lack of resources (not everything could be implemented), and so on. Therefore, the software may help the designer with some guidance on how to fix the identified defect.

- *Manual repair without any guidance*: the software has identified a violated guideline, but does not know how to fix it.

4. THE DESTINE ENVIRONMENT

Referring to the repair tool classification presented above, DESTINE consists of a complete environment for evaluating a web page against any previously defined usability and accessibility guideline and a repair tool which executes a partial interactive repair or a partial automated repair depending on the guideline to be processed. As explained before, the repair tool is useful if it is integrated to a suite. DESTINE was a perfect environment for a repair software. We shall first remind the reader about the DESTINE suite before explaining in details our repair tool. DESTINE stands for Design & Evaluation STudio for INtent-based Ergonomic Web sites. The goal of Destine is to help the conception, evaluation and repair of the accessibility quality of web sites. Our tool is based on several bases of ergonomic recommendations and does not need the use of a Web development environment. DESTINE contains five different modules (Fig. 1):

1. *A management tool of usability and accessibility knowledge*: it is responsible for organizing the knowledge bases containing the guidelines: thanks to it, the user can browse and edit any knowledge base.
2. *A Guideline Description Language (GDL) editor*: it allows to formally writing any guideline in the GDL formal language, which is the language used to write the logic evaluation of usability and accessibility guidelines that can be evaluated automatically by a machine.
3. *An Automatic Evaluator of the ergonomic quality of web sites*: it evaluates from the GDL specification the ergonomic quality of pages or web sites and gives a report showing the result of this evaluation. The report allows the user to analyze the quality of his web site but also to correct it manually.
4. *An Ergonomic Repair Tool*: it is aimed at repairing pages where ergonomic problems have been identified by the evaluation tool. The tool gives assistance to the user, which means he controls the repair process. This is the tool the paper will be focused on.
5. *The Multimodal transformation tool*: it is used to transform Web sites originally designed for a computer screen in order to make them compatible with an interactive kiosk.

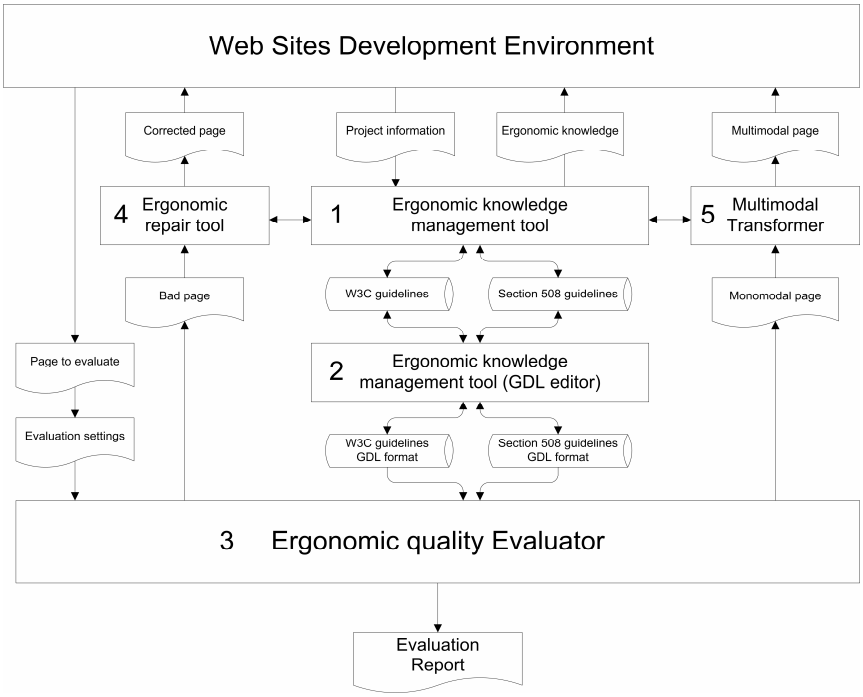

Figure 1. Global architecture of the DESTINE project.

DESTINE is a complete suite that can be used at anytime in the development process of Web sites. It can give a knowledge base for the developer, give an evaluation report of the status of his Web site, then repair ergonomic errors and adapt this site to other hardware. One of the advantages of DESTINE is the possibility of an automatic evaluation of ergonomic guidelines or a manual evaluation by the user using the knowledge base of the management tool. Another advantage is the separation between the logic and the evaluation engine, in both evaluation and repair tools.

5. THE DESTINE REPAIR TOOL

To evaluate automatically and then repair a page, we split ergonomic guidelines in smaller parts that we called “technical checkpoints”. A checkpoint can be converted and integrated in a GDL (Guideline Definition Language) [4,5] file which could then be interpreted by our evaluation or repair tool. We shall call a repair candidate an HTML object that is liable to be involved in an ergonomic guideline; a piece of code extracted from the Web page in order to perform the repair according to a technical checkpoint.

5.1 GDL Files for Repair

The main asset of the DESTINE project is the GDL (Guideline Description Language) which allows a clean separation of the definition of usability and accessibility guidelines from the inference engine. We have already presented the format of this language in previous articles [3,4,5]. We chose to follow the same philosophy for the repair tool.

As a reminder, GDL is used to model the estimable aspects of HTML code (e.g., colors, contrast, alternate text for visual content). GDL specifies a suitable structure to formalize an ergonomic guideline in one or more evaluation contexts. Notice that GDL is XML compatible, like other tools using XML structures to automatically analyze the accessibility of Web pages, for example EvalIris [1] or Bobby [8]. The GDL format that is suitable for our evaluation tool, can be used by the repair tool by adding a new "correction guideline" section, as shown in Fig. 2 and Fig. 3.

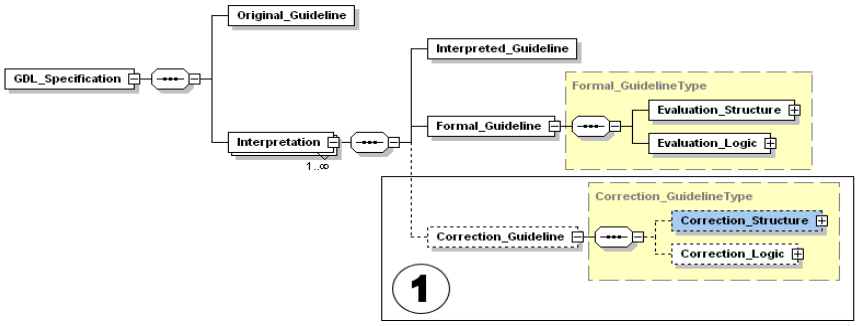

Figure 2. XSD schema of the GDL repair file with its new correction section.

The structure is based on the evaluation and allows editing HTML code instead of reading it only. The correction guideline is split between the correction structure, and the correction logic. The former defines the fields to be generated in the interface in order to let the user apply changes during the correction process. Actually, the user interface must be customized for each ergonomic failure the user wants to correct. The latter specifies the way the inference engine will treat HTML data in a bid to correct the Web page content.

Since repairing each guideline depends on the guideline type, the correction structure hold information characterizing what type of correction could be brought to fix the usability defect. For instance, if an alternate tag is missing for an image, the correction structure will require the designer to input a textual description and will feed the HTML code with this information. The designer does not need to know any HTML code since the repair tool itself adds the corresponding portion of code that fixes the defect.

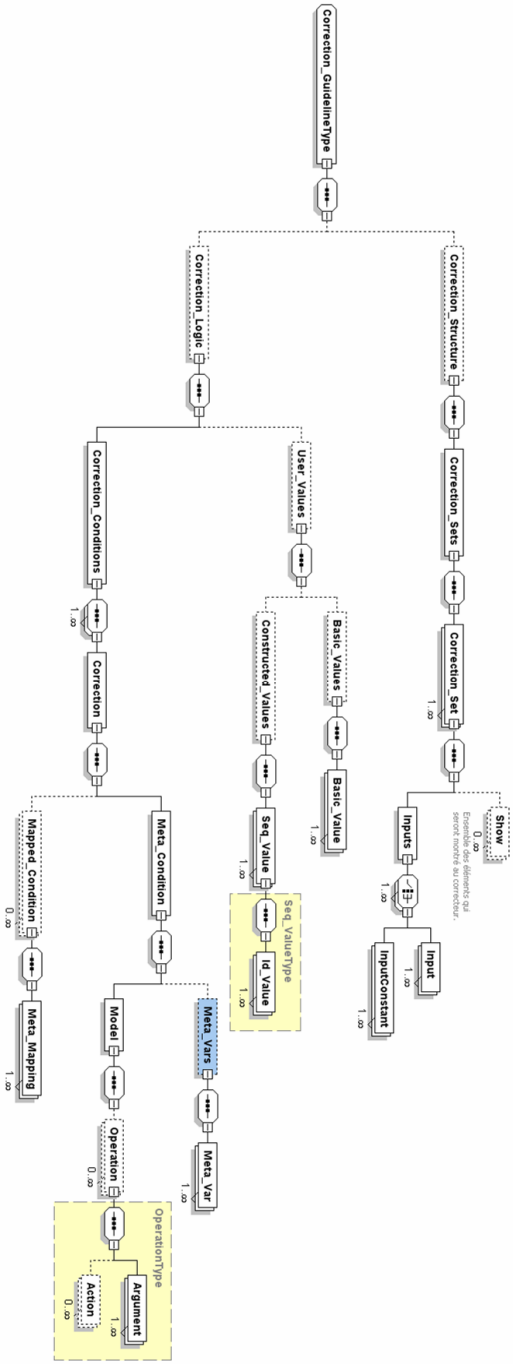

Figure 3. Detail of the correction section.

5.2 The Repair Process and the User Interface

The repair process usually takes place in these following steps:

1. The first step consists of selecting the ergonomic guidelines and the Web pages to be repaired. The user selects in the panel 1 a technical checkpoint (Fig. 4, panel 1) and a page that he had previously loaded in the panel 2 (Fig. 4, panel 2).
2. The user chooses if he wants a preliminary evaluation before starting the repair. The preliminary evaluation allows the user to realize the amount of candidates that should be repaired. Notice that the repair process always begins with an evaluation anyway.
3. In a repair process, as soon as an issue is detected, DESTINE adapts its interface (Fig. 4, panel 4) thanks to the correction structure defined in the GDL (Section 5.1) so the user can fix the HTML code.
4. The application checks the new values entered by the user by running the correction logic (Section 5.1) on them.
5. After the changes have been applied to the page, the user can choose to repair the next candidate of the current checkpoint or to restart the process at step 1.

Let us give an example about the easy checkpoint "*Provide an alternative text for images*" [18]. The GDL file specifies to fetch the tag <IMG> and to look if the attribute "alt" is fulfilled. If this is not the case, the GUI is adapted thanks to the GDL specification to allow the user fulfilling the attribute. This process is flexible: at any time, the user can choose which candidate or page he wants to repair according to a guideline. There is no predefined order to apply the correction, the user is free to follow her way and priorities to achieve the pages repair.

In our case the graphic user interface had to be carefully designed. Indeed the repair process is often fastidious and may appear even worse to the user if the interface is not clear and accessible. Unlike the evaluation tool [4] which gives a complete report of the evaluation process, the latter is presented here in a lighter form, briefing the user on the state. The interface is composed by four main panels as illustrated in Fig. 4:

- Panel 1 shows all the usability and accessibility guidelines and their technical checkpoints which can be repaired for a selected page. This panel allows us to launch the repair process or a light evaluation of the currently selected checkpoint.
- Panel 2 displays the set of pages we want to repair.
- Panel 3 shows either the HTML page in a browser with the repair candidate framed or the HTML source code with the repair candidate emphasized).
- Panel 4 contains a set of input fields in order to enter the corrected values

to repair the page. This panel also allows the navigation into a technical checkpoint among the repair candidates. It finally gives information about the ergonomic issue and how to fix it.

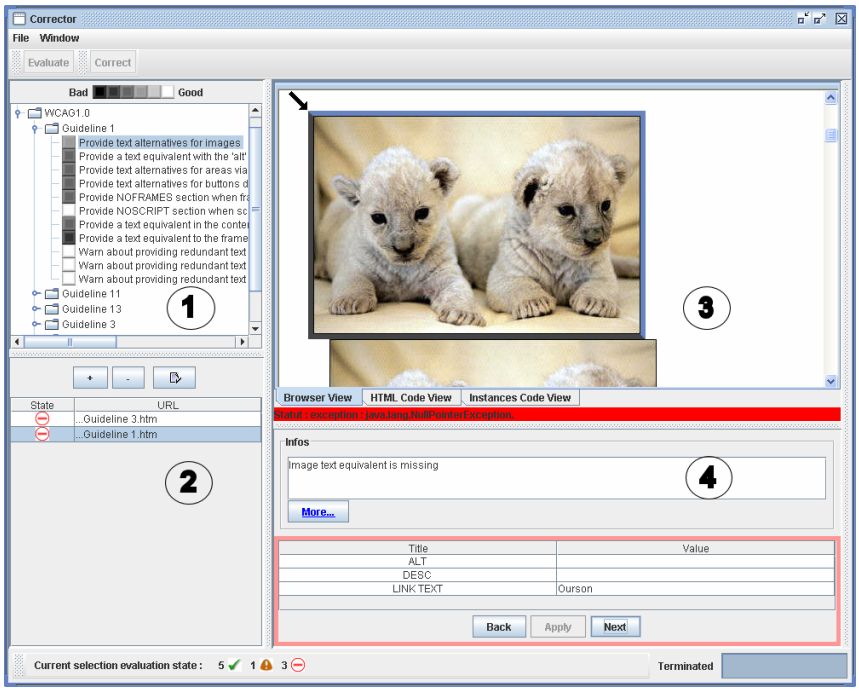

Figure 4. Repair Tool UI.

6. CHARACTERISTICS OF THE TOOL

Let us emphasize what distinguishes this tool from others. As we saw previously in this paper, evaluation and repair processes are quite similar. Indeed, the repair process starts with an evaluation and every correction given by the user needs to be checked until it is valid. The user does not absolutely have to repair, she can just run a partial evaluation independently of his repair process, so he could catch a glimpse of the work to do. The use of GDL files allows a separation between the program and the guidelines expression. That way someone can make a guideline and then evaluate or repair it without having to know anything in programming or having to recompile the application code. Finally, our evaluation tool can deal with problems due to Cascaded Style Sheets, so can our repair tool, unlike most of the existing repair tools. Notice that for the time being, the repair tool can treat CSS files but does not allow correcting them.

7. CONCLUSION

For the time being, a fully automatic repair process that outputs a perfect Web site is not completely formalizable. There are lots of guidelines which cannot be automatically evaluated and thus repaired, for example the guidelines related to the semantic of Web pages. Moreover, the repair process is a creation process; hence the designer must keep the control on the changes carried out on his pages. However, this restriction is also insoluble among the rival tools. We have shown the repair tool of DESTINE. It is able to repair automated U&A rules. In general, only 44% of the problems can be automatically evaluated for interactive application [10] and only 50% for a Web site [8]. The repair is not completely automatic but the tool can guide the user in his correction. In that point of view we can say the repair process can repair every problem that was automatically detected during the evaluation process but with the interaction of the designer. One of the main advantages of DESTINE stands in the fact that the logic expression of the guidelines to verify and the way to repair them are separated. Each repair technique can be logically defined and just parsed to be treated. This is a largely different approach from those where the repair means are hard-coded (like A-Prompt, X-Act, which is the most recent version of Bobby). In our future work, we are extended our repair tool towards the correction of web pages formatted with Cascading Style Sheets.

REFERENCES

[1] Abascal, J., Arrue, M., Fajardo, I., Garay, N., and Tomas, J., *Use of Guidelines to Automatically Verify Web Accessibility*, Universal Access in the Information Society, Vol. 3, No. 1, 2004, pp. 71-79.

[2] Accessibility solutions SSB, accessible at http://www.ssbtechnologies.com/.

[3] Beirekdar, A., Keita, M., Noirhomme, M., Randolet, F., Vanderdonckt, J., and Mariage, C., *Flexible Reporting for Automated Usability and Accessibility Evaluation of Web Sites*, in Proc. of 10th IFIP TC 13 Int. Conf. on Human-Computer Interaction INTERACT'2005 (Rome, 12-16 September 2005), Lecture Notes in Computer Science, Vol. 3585, Springer-Verlag, Berlin, 2005, pp. 281-294.

[4] Beirekdar, A., Vanderdonckt, J., and Noirhomme-Fraiture, M., *A Framework and a Language for Usability Automatic Evaluation of Web Sites by Static Analysis of HTML Source Code*, Chapter 29, in Ch. Kolski, J. Vanderdonckt (eds.), Proc. of 4th Int. Conf. on Computer-Aided Design of User Interfaces CADUI'2002 (Valenciennes, 15-17 May 2002), Kluwer Academics Pub., Dordrecht, 2002, pp. 337-348.

[5] Beirekdar, A., Vanderdonckt, J., and Noirhomme-Fraiture, M., *KWARESMI – Knowledge-based Web Automated Evaluation Tool with Reconfigurable Guidelines Optimization*, in C. Stephanidis (ed.), Proc. of 2nd Int. Conf. on Universal Access in Human-Computer Interaction UAHCI'2003 (Creete, 22-27 June 2003), Vol. 4, Lawrence Erlbaum Associates, Mahwah, 2003, pp. 1504-1508.

[6] Blackmon, M.H., Kitajima, M., and Polson, P.G., *Repairing Usability Problems Identified by the Cognitive Walkthrough for the Web*, in Proc. of the ACM Conf. on Human Aspects in Computing Systems CHI'2003 (Ft. Lauderdale, 5-10 April 2003), ACM Press, New York, 2003, pp. 497-504.

[7] Blackmon, M.H., Kitajima, M., and Polson, P.G., *Tool for Accurately Predicting Website Navigation Problems, Non-Problems, Problem Severity, and Effectiveness of Repairs*, in Proc. of the ACM Conf. on Human Aspects in Computing Systems CHI'2005 (Portland, 2-7 April 2005), ACM Press, New York, 2005, pp. 31-40.

[8] Cooper, M., Limbourg, Q., Mariage, C., and Vanderdonckt, J., *Integrating Universal Design into a Global Approach for Managing Very Large Web Sites*, in A. Kobsa, C. Stephanidis (eds.), Proc. of the 5th ERCIM Workshop on User Interfaces for All UI4ALL'99 (Dagstuhl, 28 November-1 December 1999), GMD Report 74, GMD – Forschungszentrum Informationstechnik GmbH, Sankt Augustin, 1999, pp. 131-150.

[9] Deque Ramp Product page, accessible at http://www.deque.com/products/ramp.html.

[10] Farenc, Ch., Liberati, V., and Barthet, M.-F., *Automatic Ergonomic Evaluation: What are the Limits?*, in J. Vanderdonckt (ed.), Proc. of the 2nd Int. Workshop on Computer-Aided Design of User Interfaces CADUI'96 (Namur, 5-7 June 1996), Presses Universitaires de Namur, Namur, 1996, pp. 159-170.

[11] Gaedke, M., Nussbaumer, M., and Meinecke, J., *An Agile System Facilitating the Production of Service-Oriented Web Applications*, in M. Matera, S. Comai (eds.), "Engineering Advanced Web Applications", Rinton Press, Paramus, 2004.

[12] Ivory, M.Y. and Megraw, R., *Evolution of Web Site Design Patterns*, ACM Transactions on Information Systems, Vol. 23, No. 4, October 2005, pp 463-497.

[13] Luque-Centeno, V., Delgado-Kloos, C., Arias-Fisteus, J., and Alvarez-Alvarez, L., *Web Accessibility Evaluation Tools: a Survey and some Improvements*, in M. Alpuente, S. Escobar, M. Falaschi (eds.), Proc. of First International Workshop on Automated Specification and Verification of Web Sites WWV'05 (Valencia, 14-15 March 2005), Valecnia, 2005, pp. 83-96.

[14] The Semantic Web Accessibility Platform SWAP homepage, accessible at http://www.ubaccess.com/swap.html

[15] UsableNet web solutions homepage, accessible at http://www.usablenet.com/

[16] Vanderdonckt, J., Beirekdar, A., and Noirhomme-Fraiture, M., *Automated Evaluation of Web Usability and Accessibility by Guideline Review*, in N. Koch, P. Fraternali, M. Wirsing (eds.), Proc. of 4th Int. Conf. on Web Engineering ICWE'04 (Munich, 28-30 July 2004), Lecture Notes in Computer Science, Vol. 3140, Springer-Verlag, Berlin, 2004, pp. 17-30.

[17] Web Accessibility Verifier A-Prompt, accessible at http://aprompt.snow.utoronto.ca/.

[18] Web Content Accessibility Guidelines 1.0, W3C Recommendation, 5 May 1999, accessible at http://www.w3.org/TR/WAI-WEBCONTENT/.