Automatic Support for Web User Studies with SCONE and TEA

Hartmut Obendorf, Harald Weinreich, Torsten Hass
Department for Informatics
University of Hamburg
{obendorf, weinreich, 2hass}@informatik.uni-hamburg.de

ABSTRACT
This paper describes the concepts of TEA, a flexible tool that supports user tests by automating repetitive tasks and collecting data of user inputs and actions.

TEA was specifically designed for user studies in the World Wide Web and is able to interact with a web browser. Building on a web intermediary (WBI) and a framework for web enhancement tools (SCONE), TEA can be applied in a range of test settings – providing either a controlled laboratory environment or a quick tool for collecting informal data.

Author Keywords
WWW, Usability Testing Methods, Test Automation

General Terms
Experimentation, Human Factors

ACM Classification Keywords
H5.2 Information interfaces and presentation (e.g. HCI): User Interfaces – Evaluation/methodology.

USER TESTS FOR WEBSITES

Costs and effectiveness of usability evaluation methods are of great practical relevance and often the main factors for neglecting tests. This also became evident in the discount usability discussion, focusing on the question of how many users are needed to identify an acceptably high ratio of all usability shortcomings in an application [1]. However, this discussion did hardly consider the importance of how to ensure an *efficient* execution of user tests, providing a constructive message for usability practice. For the practitioner usually not the *number* of errors detected is significant, but the acquisition of information on how the system can be improved [13].

Empirical usability methods, and especially user test methods, are a vital source of information for the design of applications and websites, as many problems will only surface when users are involved in testing [10, p. 165]. However, analytical or inspection methods – such as expert walkthroughs or heuristic evaluations – are often preferred, as empirical test methods are commonly considered more costly. On the other hand there are also indications that empirical methods reveal more *severe* problems and help to gather ideas for *how* a better interface alternative could be developed – information that is often unavailable from other tests methods [10]. Furthermore, preparation and analysis times are often underestimated for walkthrough techniques [7].

Websites differ from common WIMP (windows, icons, menus, pointers) software products, and although similar methods can be applied to evaluate their usability, only few tools support usability experts in executing user tests for websites [6]. Experts are needed to perform each phase of a study, like running the actual tests or analyzing and transcribing the audio and video recordings. This renders such tests very time and cost intensive. These findings motivated the development of the tool presented in this paper, called *TEA* for *Test Environment Automation*.

BENEFITS OF USER TEST AUTOMATION

Several techniques exist to evaluate single aspects of the usability of a website. These methods range from link checkers, accessibility and HTML code validators to server log analyses and visualization tools [5]. Other approaches evaluate the usability of a site by means of statistical heuristics like the average page length and the number of links per page [6]. However, to *improve* the usability of a site, all aspects – such as structure, page layout and wording – need to be consistent and coherent from the user's point of view [4]. Only empirical tests with real users can ensure this.

This paper aims at a reduction of effort for the usability engineer while preserving the benefits of user studies. The developed tool, TEA, was designed to provide an *integrated* automation support for user test studies of websites.

User tests can be divided into different phases: First, a test has to be planned, participants have to be acquired and the testing environment, including test material and prototypes, has to be prepared. Then the actual user test can be performed; as much relevant information as possible has to be recorded during this phase. Next, the gathered data must be transcribed into a format suitable for analysis. Finally, this data has to be analyzed and summarized.

The actual execution of a user test has many repetitive elements for the evaluator, which we believe to be potential

CHI 2004, April 24–29, 2004, Vienna, Austria.
ACM 1-58113-703-6/04/0004.

targets for automation support. Repetitive elements can be found in test introduction and in the assignment of tasks, by inquiring the user and recording her answers during the test. Most of these tasks could be supported through a software tool.

Such automation for user tests promises many advantages. It can help to ensure a *controlled environment* by reducing the impact of human and environmental factors. It also simplifies the reproduction of a test, making it possible for several evaluators to perform the same study simultaneously.

Partial test automation can also reduce the cognitive demands on the experiment conductor, allowing her to concentrate on observing the participant, thus increasing the quality of her annotations. At the same time, the required level of expertise is reduced, and the main researcher can employ co-evaluators with less expertise to perform tests. By reducing the cost per test, more user studies can be included in an evaluation, resulting in data that yields more reliable and statistically significant effects.

Another information source that is often used in user tests are data of the participant's actions. The collection of this data and the integration and synchronization with other data is an important area for automation support. Today, often audio and video recording techniques are used to observe the user's behavior. Additionally, tools such as screen capturing software or key loggers can be utilized to monitor the actions of a user during the tests. However, these methods only deliver low-level information and require time-consuming coding of the recordings to make further analysis possible. By providing more high-level, pre-processed information, the analysis phase could be simplified.

Unfortunately, automation support of empirical methods is greatly under-explored, and only few tools exist that achieve some of these tasks [5]. TEA is a tool that was developed to provide integrated support for test execution *and* data capturing at a high level – to make the execution of user tests with web systems easier, faster and cheaper.

TEA AT WORK: SUPPORTING USER TESTS WITH TEA

Automated tests can easily be created for TEA using XML definition files. Using the provided XML schema, XML editor tools can assist the experiment designer in writing test definitions. This enables non-programmers to define new test scenarios. XML tags define the test procedure, including tasks of the test and data to be gathered. Each task is characterized by screen output data (in HTML), input fields and interactive elements, as well as the behavior of the tool and the web browser during that task.

One of the key benefits of TEA is the automation of data collection. All required user input can be recorded as well as the user actions with the web browser. This decreases the amount of work needed for data collection and analysis phases, which otherwise often demand a considerable amount of project time.

To facilitate further processing, TEA provides test output in two common formats: structured XML files allow post-processing in analysis or visualization tools without loss of contextual information; CSV files allow the direct import in tools like Excel or SPSS for statistical analysis.

TEA is a very flexible tool that can easily be extended, as the complete object-oriented Java source code is provided. Other than related projects, TEA provides an *integrated* test environment for several kinds of user tests.

TEA Illustrated

TEA can be applied in different contexts. Example uses are described here in the order of their complexity. The first section will show the automation of a typical user test followed by a survey. Then the use of TEA will be extended to control the browser and log clickstream data. Finally, the capabilities of TEA and Scone for filtering the web and for evaluating web enhancements will be illustrated.

Supporting Survey Execution

A typical website evaluation, e.g. aiming at the structural redesign of a complex news site, could follow a procedure like this: first, the participants individually explore the site to get a feel for it. After this phase, they are asked to search for specific information on the site and type in the answers. The session is closed with a short survey.

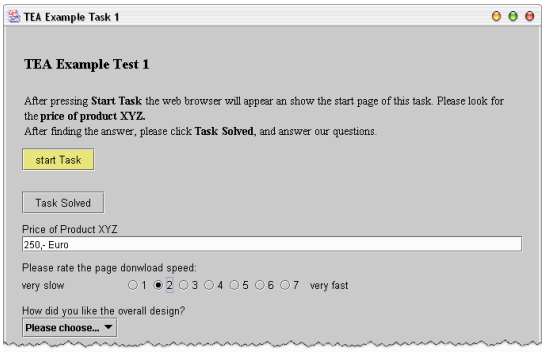

Figure 1: TEA test example built with standard widgets.

TEA controls the test procedures and manages the interaction with the users. It can provide consistent textual instructions, present the test tasks to the users and record their answers, keeping track of task time.

A closing survey can also be handled by TEA. It provides standard questionnaire widgets (pull-down menus, point-scales and text fields). Using a template facility, the evaluator can easily reuse tests designed for other evaluations or standard surveys like QUIS [2] or WAMMI [6] that help to gather additional subjective data. It is also easily possible to interleave short surveys with the participants' tasks. This might increase the quality and amount of information remembered by the participants.

Browser Control

To offer all users the same opportunity to explore a site, an exploration phase can be stopped after a predefined time or

a pre-defined number of page visits. TEA can be used to automate both: It allows controlling of the pages displayed by the browser and also tracing all navigational actions of the user. TEA is also able to trigger the browser to open certain web pages to ensure that relevant parts of the site are visited.

During the task phase, TEA can focus the browser window and bring it to the foreground. After e.g. a predefined task-time it can also blank the page and send the window to the back to control the flow of the study. This works in dual monitor mode and with layered windows on one screen.

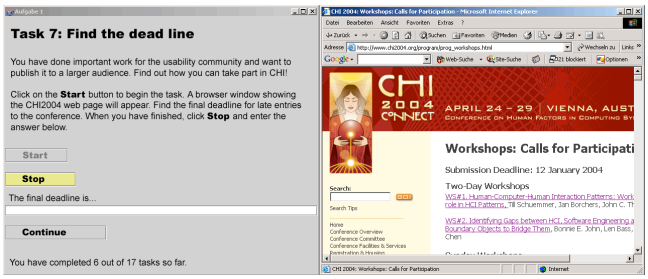

Figure 2: Dual screens or automatic control of top window.

Capturing Clickstream Data

Motivated by political or research questions, it is often necessary to combine the gathered qualitative information with quantitative data that can show statistical differences, e.g. between different versions of a website.

Quantitative usability approaches in the web are usually based on server log analyses. Thus, they are limited by the data contained in the logs and fail to collect data on many potentially crucial user interactions on the client-side, like scrolling activity or mouse movements [3]. Proxy servers, browser caches and dynamic client-IP addresses bias the data and hide the exact paths a single user takes. Also, interaction with external websites remain hidden, and it cannot be discerned how a user left the site.

WebVIP was one of the first tools developed to trace user actions on a website in greater detail [12]. However, WebVIP requires a local copy of the entire website to be evaluated and modifies it by adding JavaScript code to every link. WET is another tool that follows a similar approach. Though, WET needs modifying all original pages on the server to add some JavaScript code. Thus, write access to the evaluated site is required, and only activities on one server can be traced [3].

WebQuilt [4] functions as server-site proxy, redirecting all links of a remote site, thus making it possible to evaluate any website without changes to client or server. However, it has the disadvantage that the captured data is only marginally more precise than data collected from a server log. Also, non-HTML links in Flash or other multimedia objects disable the tool.

TEA works as client-side proxy automatically adding JavaScript code and a hidden Java applet to each page. The JavaScript code is used to trace user events, while the applet instantly transmits this information to TEA, which stores it in a database. The combination of these techniques makes it possible to capture more detailed and extensive data than regular server or proxy logs could provide. It also eliminates problems like unrecorded backwards navigation hidden by the browser cache.

TEA not only provides all browsing events, recording what pages were visited by a user, their loading times, and how long a user stayed. It also gathers information on *how* a user got to a page, as both the initiation of a link as well as any form input is stored. TEA can also handle interaction with multiple browser windows or frames. Capturing of further events can easily be added. At the same time, the data are recorded at a much higher level of abstraction than with key logging tools or video recordings, making analysis easier.

TEA can also take advantage of user actions to control the test procedure, e.g. a task can be automatically ended when a link is clicked, increasing the precision of time-to-target measurement. It might also be interesting to analyze the number of abort and back button uses as an indicator for navigational breakdowns. Finally, TEA will also take note of unorthodox approaches, such as the use of search engines, and store the input of the form fields used.

Filtering the Evaluated Website on the Fly

TEA integrates with an intermediary framework [8], utilizing a number of sophisticated techniques to change what the user is experiencing when interacting with the web. Incoming data can be filtered, modified or enriched with data stored locally or obtained from the internet. The applied framework is called Scone [14]. Scone is based on IBM's WBI Intermediary architecture, enhancing it with tools for faster and easier HTML processing and analysis.

One example application of the intermediary is to hide parts of the web from the user, like removing external links from a site or filtering the output of search engines. Thus, the attention of the participants can be 'guided' to the information they are supposed to evaluate.

Scone also allows the testing of alternative designs without need for write access to the tested site. The intermediary can alter a given website, e.g. simply by exchanging its CSS file. One could also implement a completely new navigation logic while drawing on the contents of the original website, or create a prototype site based on several Web Services. Possible modifications also include varying the site's response speed or testing new collaborative functionality or different navigational aids, e.g. breadcrumbs vs. a navigation bar.

Testing Web Enhancements with Scone

The Scone framework offers components supplementing the intermediary, like a multi-threaded robot, browser usage monitoring and a data persistency layer. These components

support the creation of many kinds of new web navigation tool prototypes [14].

Two examples of such prototypes are HyperScout [15] and Browsing Icons [9] that were evaluated with prior versions of the current TEA implementation.

FIRST EXPERIENCES WITH TEA

The current TEA implementation and its predecessors were successfully used in several laboratory studies. For instance, we conducted two studies on effects of different link marker visualizations [11]. The first study compared the effects of different link appearances on the readability of web pages. The second study changed the browser's behavior, making links only visible after pressing a special key.

These studies faced a threefold challenge: a controlled experimental environment that supported the quantitative nature of the study and minimized environmental factors was needed. Furthermore, exact task times had to be recorded, while the type of measurement (user-ended task vs. link event) differed between task types, making a manual measurement error-prone. TEA simplified the statistical analysis of the recorded user actions without expensive transcription.

Finally, we needed to change the link marker visualization of web pages grabbed from a news ticker by inserting CSS into the pages, also eliminating distracting advertisements and navigation elements. All this was accomplished by implementing a simple filter using the Scone framework.

User reactions were very encouraging. The computer-supported test procedures did not cause any severe problems. Minor problems we observed with some user interface features of TEA, like the coloring of buttons or characteristics of Java Swing, were consequently fixed.

In another study, TEA was used to evaluate a tool that augmented the information contained in hyperlinks. Tooltips containing additional information about the target of the link were activated via mouse-overs. This test involved gathering information about mouse movements and navigation paths taken by the users.

CONCLUSION

TEA is the first *integrated* environment for user tests in the World Wide Web. Although the components – clickstream logging using a proxy and scripts, controlling the browser by Java Applets and conducting surveys with online tools – are known and widely in use, the combination of these elements with a simple XML definition language and high-level protocol files, creates a powerful new tool to support user testing in a variety of settings.

TEA is an open source re-implementation based on earlier prototypes and specifically designed for further enhancements and modifications, thus providing a basis for a variety of different user tests on the World Wide Web.

By providing TEA, we hope to enable and encourage more researchers doing user tests with new web technologies.

More information can be found at: `www.scone.de/TEA`

REFERENCES

1. Barnum, C., Bevan, N., Cockton, G., Nielsen, J., Spool, J. & Wixon, D.: The "Magic Number 5": Is It Enough for Web Testing? Ext. Abstracts ACM CHI 2003, 698-699.
2. Chin, J. P., Diehl, V. A. & Norman, K. L.: Development of an Instrument Measuring User Satisfaction of the Human-Computer Interface, Proc. ACM CHI'88, 213-218.
3. Etgen, M. & Cantor, J.: What does getting WET (Web Event-logging Tool) Mean for Web Usability? Proc. of NSCL Human Factors & the Web Conference, 1999.
4. Hong, J. I., Heer, J., Waterson, S. & Landay, J.: WebQuilt: A Proxy-based Approach to Remote Web Usability Testing. ACM Transactions on Information Systems, 19(3), 2001, 263-285.
5. Ivory, M. Y. & Hearst, M.A.: The State of the Art in Automating Usability Evaluation of User Interfaces. ACM Computing Surveys, 33(4), 2001, 470-516.
6. Ivory, M. Y., Sinha, R. R. & Hearst, M. A.: Empirically Validated Web Page Design Metrics, Proc. ACM CHI'01, ACM Press (2001), 53-60.
7. Karat, C.-M., Campbell, R. & Fiegel, T.: Comparison of Empirical Testing and Walkthrough Methods in User Interface Evaluation. Proc. ACM CHI 1992, 397-404.
8. Maglio, P. & Barrett, R.: Intermediaries Personalize Information Streams. Communications of the ACM, 43(8), 2000, 96-101.
9. Mayer, M. & Bederson, B. B.: Browsing Icons: A Task-Based Approach for a Visual Web History, HCIL-2001-19, CS-TR-4308, UMIACS-TR-2001-85, HCI Lab, University of Maryland, Maryland, USA, 2001.
10. Nielsen, J.: Usability Engineering. Acad. Press 1994.
11. Obendorf, H. & Weinreich, H.: Comparing Link Marker Visualization Techniques – Changes in Reading Behavior. Proc. 13th WWW Conf., 2003, 736-745.
12. Scholtz, J., Laskowski, S. & Downey, L.: Developing Usability Tools and Techniques for Designing and Testing Web Sites, 4th Annual Human Factors and the Web Conference, June 5, ATT, NJ., 1998.
13. Wixon, D.: Evaluating Usability Methods: Why the Current Literature Fails the Practitioner. ACM interactions, 10(4), 2003, 28-34.
14. Weinreich, H., Buchmann, V., Lamersdorf, W.: Scone: Ein Framework zur evaluativen Realisierung von Erweiterungen des Webs, KiVS '03, Springer, 2003, 31-42.
15. Weinreich, H. & Lamersdorf, W.: Concepts for Improved Visualization of Web Link Attributes, Computer Networks, vol. 33, no. 1-6, 2000, 403-416.