Web Usability Probe: A Tool for Supporting Remote Usability Evaluation of Web Sites

Tonio Carta[1], Fabio Paternò[1], and Vagner Figuerêdo de Santana[1,2]

[1] CNR-ISTI, HIIS Laboratory, Via Moruzzi 1, 56124 Pisa, Italy
[2] Institute of Computing - University of Campinas, Albert Einstein Av.,
1251, Campinas, São Paulo, Brazil
{Tonio.Carta,Fabio.Paterno}@isti.cnr.it,
vsantana@ic.unicamp.br

Abstract. Usability evaluation of Web sites is still a difficult and time-consuming task, often performed manually. This paper presents a tool that supports remote usability evaluation of Web sites. The tool considers client-side data on user interactions and JavaScript events. In addition, it allows the definition of custom events, giving evaluators the flexibility to add specific events to be detected and considered in the evaluation. The tool supports evaluation of any Web site by exploiting a proxy-based architecture and enables the evaluator to perform a comparison between actual user behavior and an optimal sequence of actions.

Keywords: Tools for Usability Evaluation, Remote evaluation, Log analysis.

1 Introduction

Although usability has long been addressed and discussed, when people navigate the Web they often encounter a number of usability issues. This is also due to the fact that Web surfers often decide on the spur of the moment what to do and whether to continue to navigate in a Web site. Usability evaluation is thus an important phase in the deployment of Web applications. For this purpose automatic tools are very useful to gather larger amount of usability data and support their analysis.

Remote evaluation [2] implies that users and evaluators are separated in time and/or space. This is important in order to analyse users in their daily environments and decreases the costs of the evaluation without requiring the use of specific laboratories and asking the users to move. In addition, tools for remote Web usability evaluation should be sufficiently general so that they can be used to analyze user behavior even when using various browsers or applications developed using different toolkits. This work presents Web Usability Probe (WUP), a tool that follows a proxy-based architecture, performs remote evaluation, and considers client-side logs as data source. We prefer logging on the client-side in order to be able to capture any user-generated events, which can provide useful hints regarding possible usability problems. Moreover, WUP allows the usability experts to analyze through some graphical representations of the logged data how users interacted with the user interface (UI).

Ivory and Hearst [4] provided a good discussion of tools for usability evaluation according to a taxonomy based on four dimensions: method class (the type of evaluation); method type (how the evaluation is conducted); automation type (the evaluation aspect that is automated); and effort level (the type of method required to execute the method). According to this classification, the WUP solution for usability testing involves: capturing logs generated at client-side, supporting automatic analysis and a number of visualizations to ease the identification of the usability issues, and only requiring that users perform some predefined tasks specified by the evaluators.

Google Analytics [1] has the potential to be configured to capture custom events at client-side and it offers a number of statistics information and reports, but it is rather limited in terms of number of events that it captures for each session. Model-based approaches have been used to support usability evaluation. An example was WebRemUsine [6], which was a tool for remote usability evaluation of Web applications through browser logs and task models. Propp and Frobrig [7] have used task models for supporting usability evaluation of a different type of application: cooperative behavior of people interacting in smart environments. A different use of models is in [5], in which the authors discuss how task models can enhance visualization of the usability test log. In our case we do not require the effort of developing models to apply our tool. We only require that the designer provides an example of optimal use associated with each of the relevant tasks. WUP will then compare the logs with the actual use with the optimal log in order to identify deviations, which may indicate potential usability problems.

2   The Approach Proposed

The approach proposed is based on an intermediate proxy server whose purpose is to annotate the accessed Web pages in order to include JavaScripts that will carry out the logging of the actual user behavior (see Figure 1). WUP does not require use of plug-in installation or specific client configuration. The scripts used by the tool are stored in the proxy server and thus there is no security conflict when the page is accessed from the client since the page appears as coming from the proxy server.
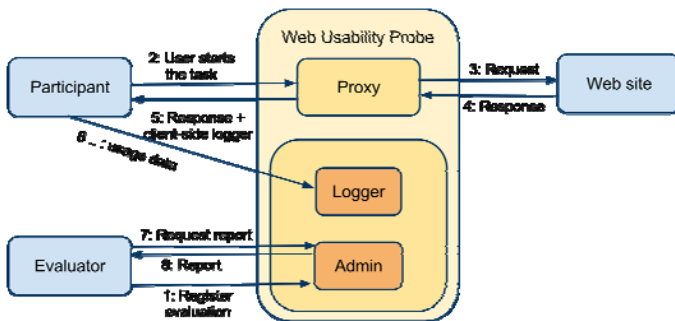

Fig. 1. Overview of the approach proposed in the Web Usability Probe

The WUP's server also acts as a usability server in which the evaluator can enter information useful to guide the usability test, such as the list of tasks to perform, what events to log, etc. Once users have carried out their test tasks, the logs can be accessed by the evaluators who can customize their representations by selecting what actions to be represented in the reports in order to ease the identification of usability issues.

When a new user test is to be started, the evaluator can access the tool to provide users with the information required to carry out the test (task descriptions, etc.). The evaluator then plays the role of user by performing the tasks in an optimal way, without errors (which are actions useless for the current task). The optimal task is defined during the planning phase of user test and the evaluator has to perform it according to tasks' descriptions, considering natural usage, and without useless actions. Users only need to start using the Web site to be evaluated through the proper link available at the proxy. Then, when users start the user test they are informed about the task to accomplish. When they complete it, they have to indicate this by accessing the dialogue box automatically generated with the controls related to the task. Next, a new task to perform will be indicated until the end of the test is reached.

The scripts that are injected by the server in order to log usage data and always redirect navigation through the proxy are implemented in JavaScript and use jQuery[1]. All the logged data are sent asynchronously to the server while the users move from one page to another.

This approach satisfies a number of requirements about evaluation tools [8]: it works in different configurations of hardware and software; it does not depend on specific configurations; it does not impact on the Web site usage; and it does not interfere with the Web page.

The development of a proxy-based tool considering client-side data encounters different challenges regarding the identification of the elements that users are interacting with, how to manage element identification when the page is changed dynamically, how to manage data logging when users are going from one page to another, among others. The following are some of the solutions we adopted in order to deal with these issues.

When logging data at the client-side, identification of the target element is an issue if the target element of a certain event does not have a name or the *id* attribute. In this case, two main approaches can be followed: either ignore events involving unidentified elements or assign an *id* attribute according their position in the Web page structure, i.e., DOM (Document Object Model) Tree. The first approach does not allow the evaluator to know exactly the elements referenced by the trigged events if any usability problem is identified, which complicates its correction. The second approach, on the other hand, can cause some overhead. The approach used in the WUP generates *ids* according to the XPath[2] syntax, and thereby allowing the *id* attribute to be computer and human readable. This provides the possibility of mapping back an *id* to the Web page element being analysed.

Another challenge that a client-side logging tool should address is how to inform evaluators that only a certain part of the page has been changed dynamically. This can

---

[1] http://jquery.com
[2] http://www.w3.org/TR/xpath/

occur, for instance, via AJAX, which can result in new UI elements. In this case the event is triggered when the DOM Tree is changed, and it can be reported in the timeline, allowing the evaluators to verify where and when it occurred, helping evaluators to deal with the issue of identifying UI elements users interact with.

3   Setting Up a Remote Usability Test

The evaluators can create the settings for a remote usability test at any time. In the administrator part of the usability tool there is an item for each test indicating the name, the description, the evaluator who created it, and the number of tasks that should be performed. The tool provides dynamically the indication of the number of sessions that have been logged for that test.

Associated with each test there is an editable list of tasks (an example in Figure 2). For each task there is the indication of the name, the description, the indication of the URL where the task should be started, whether it is skipable, and if its performance depends on some other task. A dependency means that the other task should be performed first than the current one.
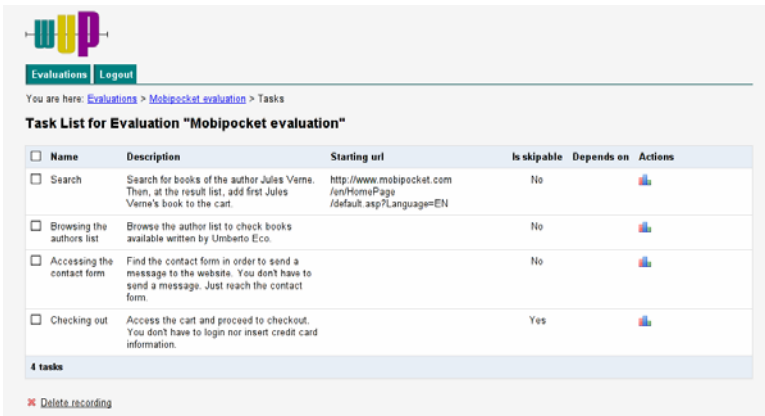

Fig. 2. An example of task list for a user test

These features aim to allow a remote participant to freely perform tasks. The name and description are the texts that users will see through automatically generated dialogue boxes. The starting URL provides flexibility in the evaluation setup, since allows evaluators to define that certain tasks have to start in different parts of the evaluated Web site, or even start in another Web site, allowing evaluations to perform comparison among Web sites.

Finally, the dependency feature among tasks is provided in order to make possible to define evaluations where one tasks is mandatory (e.g., login) in order to perform others (e.g., creation of content in a login protected Web site).

The vocabulary of events supported by WUP is one of our contributions, since it captures any standard event[3], jQuery Events[4], touch, gestures, and accelerometer events present at the Safari API[5]. The set of events observed by WUP is shown by grouping them by their type (defined according to the device that generate them), for instance: accelerometer, keyboard, mouse, touch. We also consider form-related events (e.g., change, select, and submit), system related events, and customizable events. The evaluator can define custom events, which can be various types of composition of basic events in terms of their ordering or standard events on specific parameters (e.g. a pageview event is triggered when a Web page is shown to the user), and it is possible to associate them with specific events names that can then be visualized in the reports. Moreover, the pageview custom event allows WUP to log information commonly present at server logs, since this custom event counts with URL path and its parameters.

4   Analysis of a Remote Usability Test

Once some users have actually performed the user test, the evaluator can access graphical representations of such logs. Hilbert and Redmiles [3] indicated that timelines can be useful for this purpose. We follow this approach: there is one timeline for each task performed by a user (see Figure 3). The first timeline is dedicated to the optimal log. The graphical representation is interactive and allows the designers to line up logs according to some important event under the 'lock scroll' UI control. In this way the evaluator can compare the optimal behavior with those of the actual users in order to see whether there was some particularly long interaction, due, for example, to the difficulty to understand how to proceed, or some errors that indicate some usability issue. In addition, the zoom level of timelines can be interactively set in order to identify the more adequate representation scale of the timeline.
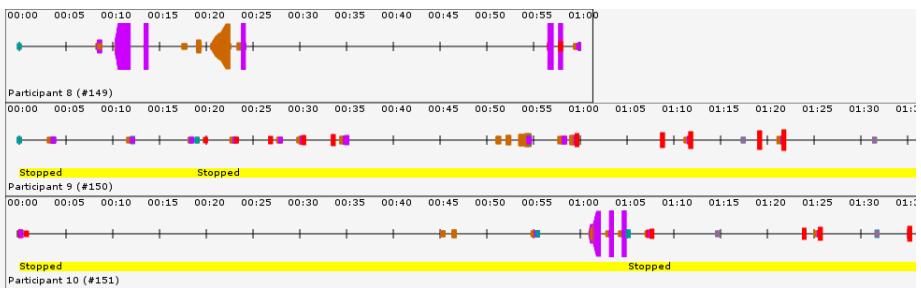

Fig. 3. Example of timelines generated by the tool in order to represent usage data

---

[3] http://www.w3schools.com/jsref/dom_obj_event.asp
[4] http://api.jquery.com/category/events/
[5] http://developer.apple.com/library/safari/#documentation/
appleapplications/reference/safariwebcontent/
HandlingEvents/HandlingEvents.html

Our proposal for representing event streams in timelines uses also the height of the time markers present in the timeline to represent the repetitions of a certain event in an element. This attribute is an indicator that allows evaluators to identify useless actions as well as to check repetitive mouse movements over bad designed link, misguided clicks on a non-clickable element, among others.

When visualizing the timelines of a certain task it is possible to zoom in and out in such representations and visualize them at the very basic event level or at the categories level. In the case of the categories visualization, the timeline uses different colors for different sets of events.

5   An Example Application

In this section we illustrate an application of our tool and show how an evaluator can infer usability issues from the visualizations provided by the tool. We report on a usability test of the Mobipocket[6] Web site, a known Web site for buying e-books. The tasks to be performed at the Mobipocket Web site, which were specified in the configuration of the WUP, were the following:

1. 'Search for books written by Jules Verne. Then, at the result list, add first Jules Verne's book to the cart.'
2. 'Browse the author list to check books available written by Umberto Eco.'
3. 'Access the cart and proceed to checkout. You don't have to login nor insert credit card information.' This task was dependent on the first task.
4. 'Find the contact form in order to send a message to the website. You don't have to send a message. Just reach the contact form.'
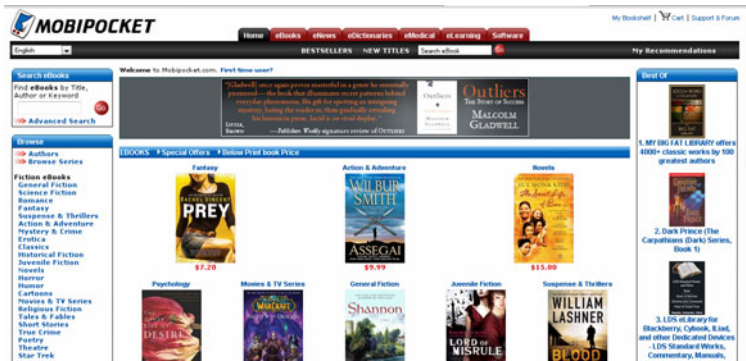

Fig. 4. The application considered in the case study

This usability evaluation involved 10 users. All of them were used to use Internet and were potential buyers of e-books, thus making part of the set of target users. Examples of results obtained from the timelines are the following:

---

[6] http://mobipocket.com

- During the task of adding a book of a certain author to the cart, the Mobipocket Web site showed as the first result a promotion of a packet of 100 authors. Figure 5 (timeline A) presents that one user was in doubt on clicking on the first result, moved the mouse over the result list and finally clicked on the correct book related to the task. It was possible to verify this issue by analyzing the timelines and checking that one participant performed the task in a different way from other participants. Thus, after zooming in the indicated timeline region it was possible to check details about events, for instance, mouse enter and mouse leave events over the first result of the search (indicated by the generated *id*) and then mouse enter event followed by a click on the add to cart button at the second result;
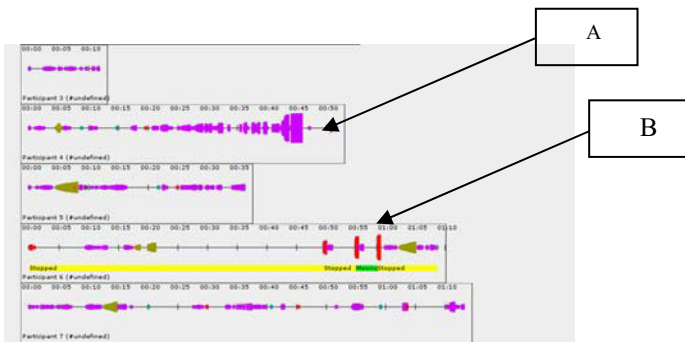

Fig. 5. Visualization of timelines; A) Repeated mouse move events indicating the doubt of the user in selecting the first search result; B) Resize events triggered by the popup blocker

- When adding a book to the cart, the feedback presented to the user was via a popup informing "This e-Book has been added to your shopping cart". Figure 5 shows page resizing events just after adding the book to the cart. After noticing this result in the timeline and mapping back to the UI it was possible to verify that when some browser's popup blocker message was presented, a sequence of resize events was then triggered;
- For the browsing authors list task it was possible to identify that some users tried to find 'Umberto Eco' under the letter 'U' page instead of the letter 'E' page, indicating lack of user guidance referring to the index, since users found unclear if the letter index refers to names or surnames;
- Considering the task of accessing the contact form we detected one usability issue: when users try to access the contact form via the 'contact us' link located at the footer of the Web site, they are sent to a Web page with another link to the contact page. This link in turn is an anchor to forum topics located at the middle of the contact Web page, making hard to users to find the contact form at the top of the page. Figure 6 presents the scrolling and mouse movements to find the link and then the same after accessing the page with forum topics and the contact form.
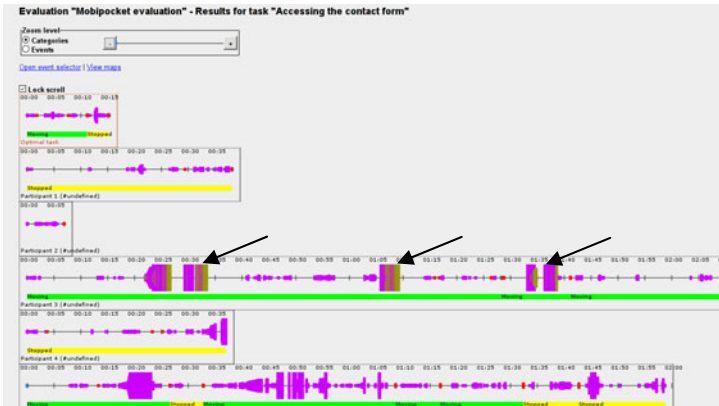
Fig. 6. Timelines referring to the access contact form. Arrows indicate mouse movements and scrolling deviation from other timelines.

6 Conclusions and Future Work

This paper presents WUP, a tool that allows evaluators to decide what tasks users should perform, and gather many types of data related to user interaction. The tool provides some graphical representations, which allow evaluators to analyze the data collected from a usability perspective. WUP also allows the end users to freely access the Web applications with any browser-enabled device without any constraint regarding where and when to perform such accesses during the test sessions. The case study reported indicated that the visual reports provided by the tool in form of timelines summarize event streams and highlight useless actions, allowing evaluators to identify usability problems, easing the task of mapping back events to actual actions occurred during user sessions. The experiment reported provided us with encouraging feedback, even if more validation will be carried out in the near future.

As future work we plan to extend the application of the Web Usability Probe considering the mobile Web, exploiting the flexibility in the definition of the events vocabulary in order to consider events related to mobile technologies.

References

1. Google: Google Analytics (2009), http://www.google.com/analytics/index.html
2. Hartson, H.R., Castillo, J.C., Kelso, J.T., Neale, W.C.: Remote Evaluation: The Network as an Extension of the Usability Laboratory. In: CHI 1996, pp. 228–235 (1996)
3. Hilbert, D.M., Redmiles, D.F.: Extracting usability information from user interface events. ACM Comput. Surv. 32(4), 384–421 (2000)
4. Ivory, M.Y., Hearst, M.A.: The state of the art in automating usability evaluation of user interfaces. ACM Comput. Surv. 33(4), 470–516 (2001)

5. Malý, I., Slavík, P.: Towards Visual Analysis of Usability Test Logs Using Task Models. In: Coninx, K., Luyten, K., Schneider, K.A. (eds.) TAMODIA 2006. LNCS, vol. 4385, pp. 24–38. Springer, Heidelberg (2007)
6. Paganelli, L., Paternò, F.: Tools for Remote Usability Evaluation of Web Applications through Browser Logs and Task Models. Behavior Research Methods, Instruments, and Computers 35(3), 369–378 (2003)
7. Propp, S., Forbrig, P.: ViSE - A Virtual Smart Environment for Usability Evaluation. In: HCSE 2010, pp. 38–45 (2010)
8. Santana, V.F., Baranauskas, M.C.C.: A Prospect of Websites Evaluation Tools Based on Event Logs. In: Human-Computer Interaction Symposium. IFIP, vol. 272, pp. 99–104 (2008), doi:10.1007/978-0-387-09678-0_9