BVG Trust J. Sci. Trans. Environ. Technov. 2017, 10(3) : 140 - 145 Scientific Transactions in Environment and Technovation

Usability evaluation using Web Usability Probe (WUP) technique

https://doi.org/10.56343/STET.116.010.003.008
http://stetjournals.com

K. Muthukkani[1] and A. Adhiselvam[2]

[1]Department of Computer Application, S.T.E.T. Women's College, Mannargudi - 614 016, Thiruvarur, Tamil Nadu, India.

Abstract

Manual process of evaluation of web sites is still a difficult and time consuming task. The technique of Web Usability Probe (WUP) that supports remote usability evaluation of these sites. They consider clients side of the data on user interaction for JavaScript events. The tools perform evaluation of any web site by exploiting a proxy based architecture and facilitates the evaluator to compare of performance between user behaviour as an optimal of action. We developed a tool to automate a significant part of the activities involved. Within the experiments on a small service-oriented website we identified usability problems, which were cross-validated by domain experts, and quantified usability improvement by the higher task success rate and lower time and effort for given tasks after implementation if suggested corrections. In this method of provides an initial validation of the effectiveness of our method.

Keywords: software tool, web usability, revisit pattern, Web server log

Received : August 2016 Revised and Accepted : January 2017

INTRODUCTION

Usability has been long addressed and discussed, when people navigate the Web often encounter a number of usability issues. This is also due to the fact that Web surfers often decide on the moment what to do and whether to continue to navigate in a Web site. The Usability evaluation is an important phase in the deployment of the Web applications. These purpose automatic tools are very useful to gather larger amount of usability data and support their analysis. Remote evaluation implies that users and evaluators are separated in time and/or space. This is important in order to analyze users in their daily environments and decreases the cost of the evaluation the uses 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 behaviour 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 and this model prefers logging on the client-side in order 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 is as to how users interacted with the user interface (UI).

Ivory and Hearst (2001) provided a good discussion of tools for usability evaluation based on dimensions taxonomy including (i) method class (the type of evaluation); (ii) method type (how the evaluation is conducted); (iii) automation type (the evaluation aspect that is automated); and (iv) 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 have the potential to be configured to capture custom events at client-side and it offers a number of statistical 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 Web Rem Usine which was a tool for remote usability evaluation of Web applications through browser logs and task models. Props and Fro brig have used task models for supporting usability evaluation of a different types of applications: such as cooperative behaviour of people interacting in smart environments. A different use of models which the authors discuss is as to how the 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.

*Corresponding Author :
email: *adhiselvam@yahoo.com*

WUP will then compare the logs with the actual use of the optimal log in order to identify deviations, which may indicate potential usability problems.

Usability engineering provides methods for measuring usability and for addressing usability issues. Heuristic evaluation by experts and user-centred testing are typically used to identify usability issues and to ensure satisfactory usability. However, significant challenges exist, which include (i) accuracy of problem identification due to false alarms common in expert evaluation (ii) unrealistic evaluation of usability due to differences between the testing environment and the actual usage environment, and (iii) increased cost due to the prolonged evolution and maintenance cycles typical for many Web applications .On the other hand, log data routinely kept at Web servers represent actual usage (Byrne, 2001). Such data have been used for usage-based testing and quality assurance and also for understanding user behaviour and guiding user interface design.

Two Types of Logs

Server-side logs and client-side logs are commonly used for Web usage and usability analysis. Server-side logs can be automatically generated by Web servers, with each entry corresponding to a user request. By analyzing these logs, Web workload was characterized and used to suggest performance enhancements for Internet Web servers (Carta and Patern, 2011). Because of the vastly uneven Web traffic, massive user population and diverse usage environment, coverage-based testing is insufficient to ensure the quality of web applications. Therefore, server-side logs have been used to construct Web usage models for usage-based Web testing or to automatically generate test cases accordingly to improve test efficiency Server logs have also been used by organizations to learn (Christoo and Riher, 2012) about the usability of their products. For example, search queries can be extracted from server logs to discover user information needs for usability task analysis. There are many advantages for using server logs for usability studies. Logs can provide insight into real users performing actual tasks in natural working conditions versus in an artificial setting of a lab. Logs also represent the activities of many users over a long period of time versus the small sample of users in a short time span in typical lab testing. Data preparation techniques and algorithms can be used to process the raw web server logs, and then mining can be performed to discover users' visitation patterns for further usability analysis. For example, organizations can mine server-side logs to predict users' behaviour and context to satisfy users' need. Users' revisit patterns can be discovered by mining server logs to develop guidelines for browser history mechanism that can be used to reduce users' cognitive and physical effort.

Client-side logs can capture accurate and comprehensive usage data for usability analysis, (Haynes, *et al.*, 2010) because they allow low-level user interaction events such as keystrokes and mouse movements to be recorded. For example, using these client-side data, the evaluator can accurately measure time spent on particular tasks or pages as well as study the use of "back" button and user click streams Such data are often used with task based approaches and models for usability analysis by comparing discrepancies between the designer's anticipation and a user's actual behaviour (Haynes, Cohen, and Ritter, 2010). However, the evaluator must program the UI, modify Web pages, or use an instrumented browser with plug-in tools or a special proxy server to collect such data. Because of privacy concerns, users generally do not want any instrument installed in their computers (Haynes, Cohen, and Ritter, 2009). Therefore, logging actual usage on the client side can best be used in lab-based experiments with explicit consent of the participants.

COGNITIVE USER MODELS

In the recent years, there is a growing need to incorporate insights from cognitive science about the mechanisms, strengths and limits of human perception and cognition to understand the human factors involved in user interface design. For example, the various constraints on cognition (e.g., system complexity) and the mechanisms and patterns of strategy selection can help human factor engineers to develop solutions and apply technologies that are better suited to human abilities. Commonly used cognitive models include GOMS, EPIC, and ACT-R. The GOMS model consists of Goals, Operators, Methods and Selection rules (Cooley, 1999). As the high-level architecture, GOMS describes behaviour and defines interactions as a static sequence of human actions. As the low-level cognitive architecture, EPIC (Executive-Process/Interactive Control) and ACT-R (Adaptive Control of Thought-Rational) can be taken as the specific implementation of the high-level architecture.

They provide detailed information regarding the simulation of human processing and cognition (Cooley *et al.*, 1999). An important feature of these low-level cognitive architectures is that they are all implemented as computer programming systems so that cognitive models may be specified, executed, and their outputs (e.g., error rates and response latencies) are compared with human performance data. ACT-R provides detailed and sophisticated process models of human performance in interactive tasks with complex interfaces (Cooley *et al.*, 1999). It allows researchers to specify the cognitive factors (e.g., domain knowledge, problem-solving strategies) by developing cognitive models of interactive behaviour. It consists of multiple modules that acquire information from the environment, process information and execute actions in the furtherance of particular goals.

Cognition proceeds (Georgeon *et al.*, 2012) via a pattern matching process that attempts to find productions that match the current contents. ACT-R is often used to understand the decisions that Web users make by following various links to satisfy their information goals. However, ACT-R has its own limitations due to the complexity of its model development and the low-level rule-based programming language. It relies on the application of Software engineering techniques to develop intelligent agents and cognitive models (Conallen, 2003). On the one hand, higher level programming languages simplify the encoding of behaviour by creating representations that map more directly to a theory that states as to how behaviour arises in humans. On the other hand, as these designs are adopted, adapted and reused, they may become design patterns.

PROPOSED MODEL

The proposed technique is based on an intermediate proxy server whose purpose is to annotate the accessed Web pages in order to include JavaScript that will carry out the logging of the actual user behaviour (Fig. 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 (Haynes *et al.*, 2009).

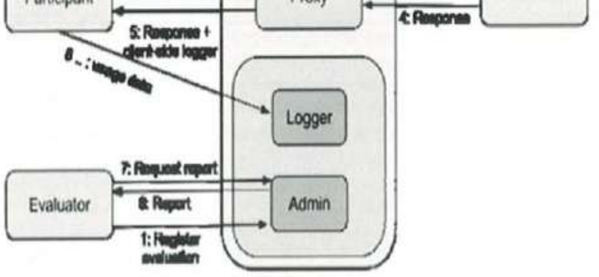


Fig.1. Proposed web usability model

The WUP's server also acts as a usability server in (Georgeon *et al.*, 2012) which the evaluator can enter information useful to guide the usability test, such as the list of tasks to perform these events to log, etc. Once users have carried out their test tasks, the logs can be used by the evaluators who can customize their representations by selecting what actions to be represented of 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 out of 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 task is defined during the planning phase of user test and the evaluator has to perform it accordingly without useless actions. Users only need to start using the Web site to be evaluated through the proper (Haynes *et al.*, 2010).

Link available at the proxy then when users start to the user test to be informed about the task of to accomplish. When they complete it, they have to indicate this by accessing the dialogue box, automatically generated with the related to the task. Next, a new task to perform will be indicated until the end of the test is reached. The server in order to log usage data, always redirect navigation through the implementations of proxy in JavaScript and use jQuery1.

All logged data are sent asynchronously to the server while the users move from one page to another page. This approach satisfies a number of requirements about evaluation tools it works in different configurations of hardware and software system; which does not depend on specific configurations; impact on the Web site usage or interfere with the Web page (Haynes, Cohen, and Ritter, 2009). The development of a proxy-based tool considering client-side data encounters different challenges regarding the identification of the elements that they are interacting with, how to manage element of identification when the page is changed auto dynamically and how to manage data logging when users are going from one page to another page, among others (Heinath *et al.*, 2007). Following are some of the solutions we adopted in order to deal with these issues.

In the event of 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, the two main alternative approaches can be followed i.e., either ignore events involving unidentified elements or assign an *id* attribute according their position in the Web page structure 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 XPath2 syntax, and thereby allowing the *id* attribute to be computer and human readable.

There is another challenge to the client-side logging tool should address to inform the evaluators that only a certain part of the page has been changed dynamically. This includes i). http://jquery.com, and

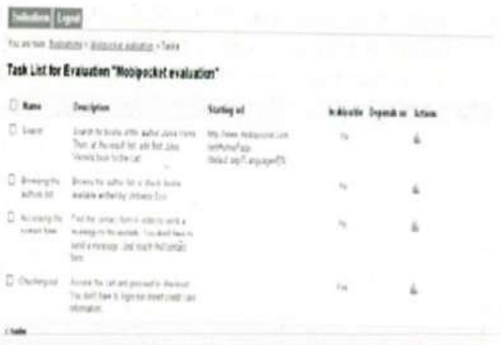
ii). http://www.w3.org/TR/xpath/

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.

WEB USABILITY MODEL

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 skiable and if its performance depends on some other task. A dependency means that the other task should be performed first before the current one.



Task List for Evaluation "Mobipocket evaluation"

Name	Description	Starting url	Is skipable	Depends on	Actions
[illegible]	[illegible]	[illegible]	[illegible]		[illegible]
[illegible]	[illegible]		[illegible]		[illegible]
[illegible]	[illegible]		[illegible]		[illegible]
[illegible]	[illegible]		[illegible]		[illegible]

Fig. 2. An example of task list for the user

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 it allows evaluators to define 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 (Hilbert and Redmiles, 2000) 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 event3, equerry events4, touch, gestures and accelerometer events present at the Safari API5. The set of events observed by WUP is shown by grouping them by their type (defined according to the device that generates them), for instance: accelerometer, keyboard, mouse and 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 page view event is triggered when a Web page is shown to the user), and it is possible to associate them with specific event's names that can then be visualized in the reports. Moreover, the page view custom event allows WUP to log information commonly present at server logs, since this custom event counts with URL path and its parameters (Ivory and Hearst, 2001).

Experiment and Analysis

Once some users have actually performed the user test, the evaluator can access graphical representations of such logs. Hilbert and Redmiles (2000) indicated that timelines can be useful for this purpose. We follow this approach where there is one timeline for each task performed by the user (see Fig. 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 behaviour with those of the actual users in order to see whether there was some particularly long interaction, due.

For example, the difficulty to understand the way to proceed, or some errors that indicates 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.

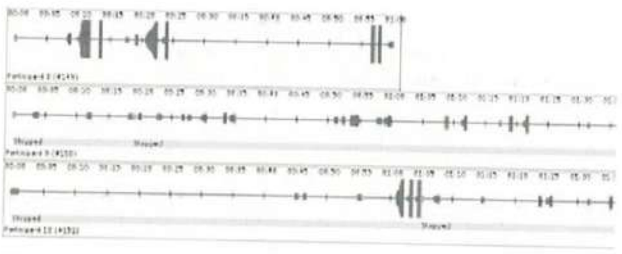


Fig. 3. Timeline generation of usage

This 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 colours for different sets of events (Kallepalli and Tian, 2001). This experiment illustrates an application of our tool and shows as to how an evaluator can infer usability issues from the visualizations provided by the tool. This report is on a usability test of the Flipkart6 Web site, a known Web site for buying e-books. The tasks to be performed at the Flip kart Web site, which were specified in the configuration of the WUP include

i. Search for books written by Jules Verne. Then, at the result list, add first Jules Verne's book to the cart,

ii. Browse the author list to check books available written by Umberto Eco,

iii. 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, and

iv. 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.

This usability evaluation involved 10 users. All of them were used to use Internet and were potential buyers of

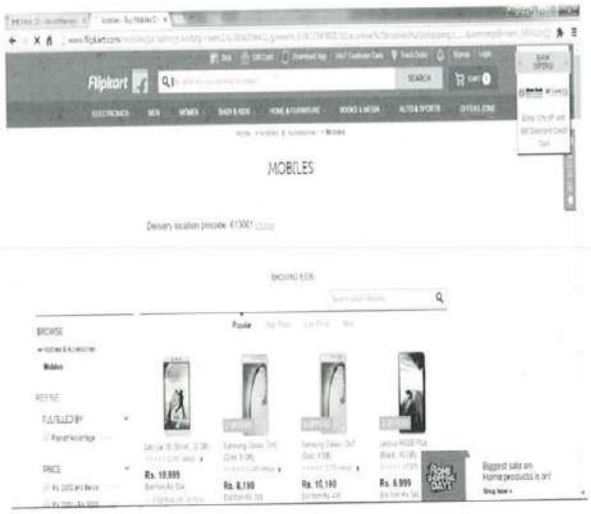


Fig. 4. Example Application used for the experiment

e-books, thus making part of the set of target users. Examples of results obtained from the timelines are the following:

During the task of adding a book of a certain author to the cart, the flip kart 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 (Kieras and Meyer, 1997). 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.

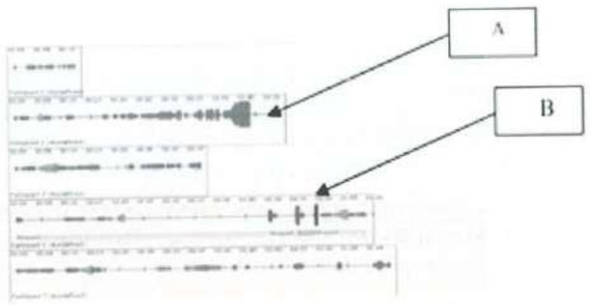


Fig. 5. Visualization of timelines; A) Repeated mouse move events indicating the doubt of the user in selecting the first search result; B) Resize events triggered by the popup blocker

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 is not clear if the letter index refers to names or surnames (Hilbert and Redmiles, 2000)

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 (Koyani *et al.*, 2004).

CONCLUSION

This paper presents WUP, a technique that allows evaluators to decide as to what tasks users should perform, and gather many types of data related to user interaction. The technique provides some graphical representations, which allows the 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 technique in form of timelines summarize event streams and highlight useless actions, allowing evaluators to the identify usability problems, easing the task of mapping back events to actual actions occurred during user sessions. The experiment provides us with encouraging feedback, even if more validation needs to be carried out in the near future.

FUTURE WORK

It is suggested that this plan is 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 in the future.

REFERENCES

Byrne, M.D. 2001. ACT-R/PM and menu selection: Applying a cognitive architecture to HCI. *International Journal of Human-Computer Studies*, 55(1):41-84.

Carta, T., Paternò, F. and de Santana, V. 2011. Web usability probe: a tool for supporting remote usability evaluation of web sites. *Human-computer interaction–INTERACT 2011*, P.349-357.

Christou, F., Ritter, E. and Jacob, R. J. 2012. CODEIN—A new notation for GOMS to handle evaluations of reality-based interaction style interfaces, *Int. J. Human-Comput. Interaction*, 28(3): 189–201.

Cohen, F., Ritter, E. and Haynes, S. R. 2010. Applying software engineering to agent development, *AI Mag.*, 31(2): 25–44.

Conallen, J. 2002. *Building Web applications with UML*. Addison-Wesley Longman Publishing Co., Inc.

Cooley, R., Mobasher, B. and Srivastava, J. 1999. Data preparation for mining World Wide Web browsing patterns, *Knowl. Inf. Syst.*, 1(1): 5–32.

Georgeon, O.L., Mille, A., Bellet, T., Mathern, B. and Ritter, F. E. 2012. Supporting activity modelling from activity traces, *Expert Syst.*, 29(3): 261–275.

Hilbert, D.M. and Redmiles, D.F. 2000. Extracting usability information from user interface events. *ACM Computing Surveys (CSUR)*, *32*(4):384-421.

Haynes, S.R., Cohen, M. A. and Ritter, F. E. 2009. Designs for explaining intelligent agents, *Int. J. Human-Comput. Stud.*, 67(1): 90–110.

Heinath, M., Dzaack, J., Wiesner, A. and Urbas, L. 2007. Simplifying the development and the analysis of cognitive models. *EuroCogSci07*.

Ivory, M.Y. and Hearst, M.A. 2001. The state of the art in automating usability evaluation of user interfaces. *ACM Comput. Surv.*, 33(4): 470–516.

Kallepalli, C. and Tian, J. 2001. Measuring and modeling usage and reliability for statistical web testing. *IEEE transactions on software engineering*, 27(11):1023-1036.

Kieras, D.E. and Meyer, D.E. 1997. An overview of the EPIC architecture for cognition and performance with application to human-computer interaction. *Human-computer interaction*, 12(4): 391-438.

Koyani, S.J., Bailey, R.W., Nall, J.R., Susan Allison, Conrad Mulligan, Kent Bailey and Mark Tolson. 2004. *Research-based web design and usability guidelines*. National Cancer Institute, P. 232.