Tools for Remote Web Usability Evaluation

Fabio Paternò

ISTI-CNR
Via G.Moruzzi, 1 – 56100 Pisa - Italy
f.paterno@cnuce.cnr.it

Abstract
The dissemination of Web applications is enormous and still growing. This raises a number of challenges for usability evaluators. Video-based analysis can be rather expensive and may provide limited results. This paper presents a discussion of what information can be provided by automatic tools for remote Web usability evaluation as well as an analysis of existing approaches, including our proposal.

1 Introduction
There are many motivations for automatic tools able to support the evaluation process (Ivory & Hearst, 2001). Tools that implement the design criteria incorporating usability principles as well as the total or partial automation of usability evaluation can reduce the time and costs involved and release evaluators from repetitive and tedious tasks. Examples of automated usability tools include code parsers and evaluators, image analysis tools, usage measurement tools (such as hit log analysis and instrumented browsers), semi-automated tools to aid the human evaluator, design evaluation and advice in web development tools, and automated online surveys. The goal of applying automatic tools is not to provide designers with an overall, definitive evaluation. Rather, a more meaningful approach is to provide a number of pieces of information that can be helpful to evaluators and developers in order to improve their applications.
Automatic evaluation is going to be an extraordinarily common technique, though it is currently in a germinal state. A good number of organizations are seriously investigating and beginning to consider automatic evaluation tools. Thus, we are at an excellent point to obtain engineered tools for widespread application. For example, with over 30 million of web sites in existence, web sites are the most prevalent and varied form of computer-human interface. At the same time, with these many web sites being designed and maintained, there will never be a sufficient number of professionals to adequately address usability issues without automation as a critical component of their approach.
With the advent of the Web and the refinement of instrumentation and monitoring tools, user interactions are being captured on a much larger scale than ever before. Automated support for the capture, representation, and empirical analysis of user behaviour is leading to new ways to evaluate usability and validate theories of human-computer interaction. It enables remote testing, allows testing with larger numbers of subjects, and motivates the development of tools for in-depth analysis. The data capture can take place in a formal experimental setting or on a deployed system. Particular attention must be paid to remote usability evaluation, where users and evaluators are distant in time and/or space. This type of approach can overcome some limitations of usability laboratories: it is often difficult to bring a considerable number of users to such laboratories, and then they have to interact with the application in an environment different from their daily working environment. In practice, lab-based tests are mostly performed with a small, local sample of the user population, which renders them inadequate for the evaluation of products to be used by people across a wide geographical area and demographic groups.

2 Approaches to Web Remote Evaluation

In empirical testing the actual user behaviour is analysed during a work session. This type of evaluation requires the evaluator to observe and record user actions in order to perform usability evaluation. Manual recording of user interactions requires a lot of effort thus automatic tools have been considered for this purpose. Some tools support video registration but also video analysis requires time and effort (usually it takes five times the duration of the session recorded) and some aspects in the user interaction can still be missed by the evaluator (such as rapid mouse selections). In model-based evaluation, evaluators apply user or task models to predict interaction performance and identify possible critical aspects. For example GOMS (Goals, Operators, Methods and Selection rules) (John & Kieras, 1996) has been used to describe an ideal error-free behaviour. Model-based approaches have proven to be useful but the lack of consideration for actual user behaviour can generate results that can be contradicted by the real user behaviour.

It becomes important to identify a method that allows evaluators to apply models in evaluation still considering information empirically derived. To this end the main goals of our work are:

- To support remote usability evaluation where users and evaluators are separated in time and/or space;
- To analyse possible mismatches between actual user behaviour and the design of the Web site represented by its task model in order to identify user errors and possible usability problems;
- To provide a set of quantitative measures (such as execution task time or page downloading time), regarding also group of users, useful for highlighting some usability problems.

As we are considering remote evaluation without direct observation of the user interactions, it is important that testing furnishes logs with detailed information. An ideal log should provide information such as who has visited the site, the paths that they have followed during the visit, how long users stay on a page, where users leave the site, and the success of the users' activities. Logs can be considered at three different levels: browser, proxy server, and web server. The most straightforward approach is to look at Web server logs. However, a number of problems arise with such an approach. It is difficult to identify the users: even the same host can receive different IP addresses in the case of modem connections, or proxy servers can hide the real original address. It is also difficult to identify the actual pages visited because the browser cache memory may hide the accesses subsequent to the first one. It is possible to determine when a page download starts, but not how long the page is actually in front of the user. It is difficult to identify the end of a user session or whether or not users have achieved their goals successfully. In the case of proxy servers, logging is done through the proxy, which captures user accesses without having to modify the participants' software or accessing the web server (Hong, Heer, Waterson, and Landay, 2001). However, there are some drawbacks also in this case because it is not possible to detect the local interactions with the browser. Thus, the use of browser logging seems to be a more promising solution as it allows detecting all the user interactions at any level of granularity, as well as all the interactions between the browser and the server. The latter information is useful, for example to distinguish the page download time from the display time. An example of tool supporting client-logging is WebVip (Web Visual Instrumenter Program) (Scholtz, Laskowsi & Downey, 1998) that has been developed at NIST. This tool allows logging of user interactions and the resulting log files can be analysed through a graphical tool that visualises the paths followed by the users during the site visit. The logging tool proposed requires a number of modifications in the HTML pages

that must be evaluated because each tag representing a user interface component calls for adding Javascript code to record the interaction. Because of the many modifications required, WebVip needs a copy of the entire site. Unfortunately copying the entire site can generate many problems. Also WET (Etgen & Cantor, 1999) considers client-side logs but they are obtained more efficiently without requiring copy of the entire site. In this case it is sufficient to include the javascript file in the heading of the page. This javascript file includes the specification of the events that can be detected and the handling functions able to capture them. In WET only the some types of events are recorded. This limitation is due also to the lack of automatic tools able to analyse the data. Since the analysis is performed manually it is important to have readable log files with content useful for the evaluator.

3 WebRemUSINE

At the HCI group of ISTI-CNR, we have developed a method and an associated tool to detect usability problems in Web interfaces through a remote evaluation (Paganelli & Paternò, 2002). Our approach combines two techniques that usually are applied separately: empirical testing and model-based evaluation. The reason for this integration is that models can be useful to detect usability problems but their use can be much more effective if they can be related to the actual use of a system. Our tool is able to analyse the possible inconsistency between the actual user interactions and the task model of the Web site that describes how its concrete design assumes that activities should be performed. To support remote evaluation, we have developed a technique that allows recording user actions during a site visit. The analysis of the logged data is based on the comparison of the traces of actions performed with the structure of the task model. This analysis provides evaluators with a number of results that are related to the tasks that users intend to perform, the Web pages and their mutual relationships.

The method is composed of three phases: *Preparation*, which consists of creating the task model of the Web site, collecting the logged data and defining the association between logged actions and basic tasks; *Automatic analysis*, where WebRemUSINE examines the logged data with the support of the task model and provides a number of results concerning the performed tasks, errors, loading time, .. *Evaluation*, the information generated is analysed by the evaluators to identify usability problems and possible improvements in the interface design.

The environment is mainly composed of three modules: the ConcurTaskTrees (Paternò, 1999) editor (publicly available at http://giove.cnuce.cnr.it/ctte.html); the logging tool that has been implemented by a combination of Javascript and Java applet to record user interactions; WebRemUSINE, a java tool able to perform an analysis of the files generated by the logging tool using the task model created with the CTTE tool.

The WebRemUSINE analysis can point out usability problems such as tasks with long performance or tasks not performed according to the task model corresponding to the Web site design. These elements are useful to identify the pages that create problems to the user. Thus the evaluation performed provides information concerning both tasks and Web pages. These results allow the evaluator to analyse the usability of the Web site from both viewpoints, for example comparing the time to perform a task with that for loading the pages involved in such a performance. WebRemUSINE also identifies the sequences of tasks performed and pages visited and is able to identify patterns of use, to evaluate if the user has performed the correct sequence of tasks according to the current goal and to count the useless actions performed. In addition, it is also able to indicate what tasks have been completed, those started but not completed and those never tried. This information is also useful for Web pages: never accessed web pages can indicate that either such pages are not interesting or that are difficult to reach. All these results can be

provided for both a single user session and a group of sessions. The latter case is useful to understand if a certain problem occurs often or is limited to specific users in particular circumstances.
During the test phase all the user actions are automatically recorded, including those associated to the goals achievement. The evaluation performed by WebRemUsine mainly consists in analysing such sequences of actions to determine whether the user has correctly performed the tasks complying the temporal relationships defined in the task model or some errors occurred. In addition, the tool evaluates whether the user is able to reach the goals and if the actions performed are actually useful to reach the predefined goals. In order to determine whether the sequence of tasks performed is complying with the temporal relations defined in the task model we used an internal simulator. For each action in the log, first the corresponding basic task is identified and next there is a check to see whether the performance of that task was logically enabled. If not then a precondition error is identified. If yes, then the list of the enabled tasks after its performance is provided. In addition, also the list of accomplished high-level tasks after its performance is provided and it is used to check whether the target task has been completed.
In the report analysing the user session, for each action there is an indication whether it was correctly performed or a precondition error occurred. The analysis of the user actions allows the detection of problems generated from the execution task order. The precondition errors highlight what task performance did not respect the temporal relations defined in the model describing the system design and consequently mismatch between the user and the system task model occurred.

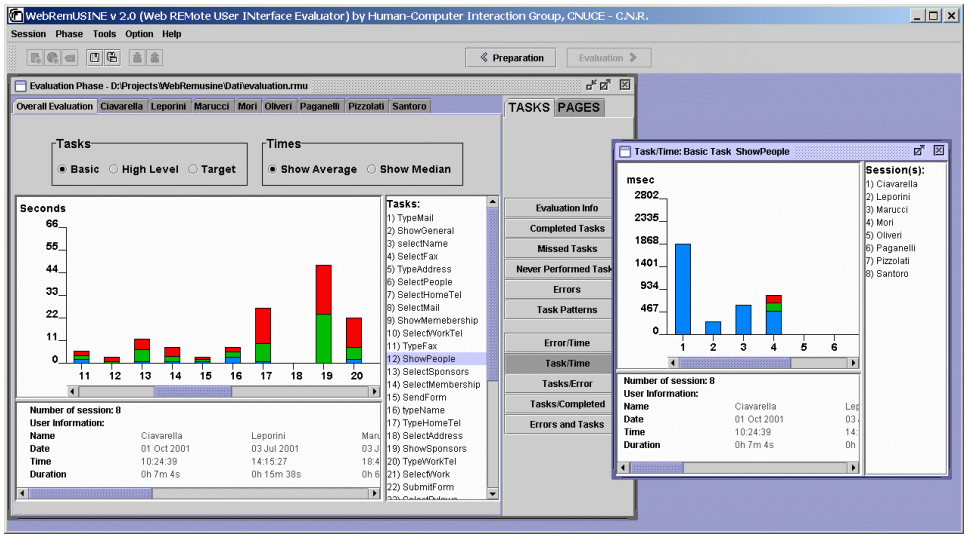

Figure 1: An Example of Results provided by WebRemUSINE.

The presence of events not associated to any task can indicate parts of the interface that create problems to the user. For example, if in the log there are events associated to images that are not associated to any link then evaluators can understand that the image confuse the user. In this case designers can change the page in such a way that it is clear that has no associated function or decide to associate a link to it.

In addition to the detailed analysis of the sequence of tasks performed by the user, evaluators are provided with several results (an example in Figure 1) that provide an overall view of the entire session considered, for example:

- The basic tasks that are performed correctly and how many times they have been performed correctly.
- The basic tasks that the user tried to perform when they were disabled, thus generating a precondition error, and the number of times the error occurred.
- The list of tasks never performed either because never tried or because of precondition errors.
- The patterns (sequences of repeated tasks) occurred during the session ad their frequency.

Such information allows the evaluator to identify what tasks are easily performed and what tasks create problems to the user. Moreover, the identification of tasks never performed can be useful to identify parts of the application that are difficult to comprehend or reach. On the basis of such information the evaluator can decide to redesign the site trying to diminish the number of activities to perform and make the task performance required of the user simpler and easier.

4 Conclusions

This paper provides a discussion of how approaches for remote usability evaluation of web sites. After a brief discussion of the motivations for this type of automatic evaluation, we analyse different solutions to logging user interactions.
Lastly, we discuss the results that can be obtained through a tool (WebRemUSINE) able to automatically analyse the information contained in Web browser logs and task models. Such information regards task performance and Web page accesses of single and multiple users. This allows evaluators to identify usability problems even if the analysis is performed remotely. More information is available at http://giove.cnuce.cnr.it/webremusine.html

5 References

Hong, J. I., Heer, J., Waterson, S. & Landay, J. A. (2001), WebQuilt: A Proxy-based Approach to Remote Web Usability Testing, ACM Transactions on Information Systems, Vol.19, N.3 July 2001, pp.263-285.

Ivory, M. Y. & Hearst M. A. (2001), The state of the art in automating usability evaluation of user interfaces. ACM Computing Surveys, 33(4), pp. 470-516, December 2001.

John, B. & Kieras, D., (1996), The GOMS family of user interface analysis techniques: comparison and contrast, ACM Transactions on Computer-Human Interaction, 3, 1996, pp.320-351.

Paganelli, L. & Paternò, F., (2002), Intelligent Analysis of User Interactions with Web Applications. Proceedings ACM IUI 2002, pp.111-118, ACM Press.

Paternò, F., (1999) Model-based design and evaluation of interactive applications, Springer Verlag, 1999. ISBN 1-85233-155-0.

Scholtz, J., & Laskowski, S., Downey L., (1998), Developing usability tools and techniques for designing and testing web sites. Proceedings HFWeb'98 (Basking Ridge, NJ, June 1998). http://www.research.att.com/conf/hfweb/ proceedings/scholtz/index.html

Etgen, M. & Cantor J., (1999), What does getting WET (WebEvent-logging Tool) mean for web usability?. Proceedings of HFWeb'99 (Gaithers-burg, Maryland, June 1999). http://zing.ncsl.nist.gov/hfweb/proceedings/etgen-cantor/index.html.