Support for Remote Usability Evaluation of Web Mobile Applications

Tonio Carta
CNR-ISTI, HIIS Laboratory
Via Moruzzi 1, 56124 Pisa, Italy
Tonio.Carta@isti.cnr.it

Fabio Paternò
CNR-ISTI, HIIS Laboratory
Via Moruzzi 1, 56124 Pisa, Italy
Fabio.Paterno@isti.cnr.it

Vagner F. de Santana
Institute of Computing, UNICAMP
Albert Einstein Av., 1251,
Campinas, São Paulo, Brazil
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 accessed through mobile devices. 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 detect and consider 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 behaviour and an optimal sequence of actions.

Categories and Subject Descriptors
H.5.2 **[Information Interfaces and Presentation]**: User Interfaces – Evaluation/methodology. D.2.2 **[Software Engineering]**: Design Tools and Techniques – User interfaces.

General Terms
Experimentation, Human Factors.

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 analyse user behaviour even when using various browsers or applications developed using different toolkits. 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, the tool allows the usability experts to analyse 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 effort required to execute the method). According to this classification our solution is for usability testing, it captures logs generated client-side, supports automatic analysis and a number of visualizations to ease the identification of the usability issues, and only requires 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 Frorbrig [7] have used task models for supporting usability evaluation of a different type of application: cooperative behaviour 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. The tool will then compare the logs with the actual use with the optimal log in order to identify deviations, which may indicate potential usability problems.

Costa et al. [8] used a logger to collect data from a user session test on a Web interface prototype running on a PDA simulator in order to evaluate different types of Web navigation tools and identify the best one for small display devices. Users were asked to find the answer to specific questions using different types of navigation tools to move from one page to another. A database

SIGDOC'11, October 3–5, 2011, Pisa, Italy.

was used to store users' actions, but they logged only the answer given by the user to each specific question. Moreover they stored separately every term searched by the user by means of the internal search tool.

Our approach is also based on a "question-answer" paradigm, but we log every user's interaction with the Web sites. The answer is often implicit and can be inferred by the page in where the user finished the interaction. Moreover, Web Usability Probe allows the evaluator to create tests based on real Web sites instead of prototypes, giving the possibility to observe the user's interaction in a real environment. While some general preliminary results were introduced in [9], in this paper we provide a description of a novel tool able to address usability evaluation of mobile Web applications.

In the paper, we first introduce our approach, then we describe how setting up a usability test with our tool, and its support for the analysis of user tests. To better present our approach we present an evaluation of a mobile Web application. Lastly, we draw some conclusions and provide indications for future work.

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 behaviour (see Figure 1). The tool 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. The 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. All such information is stored in a database. 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.

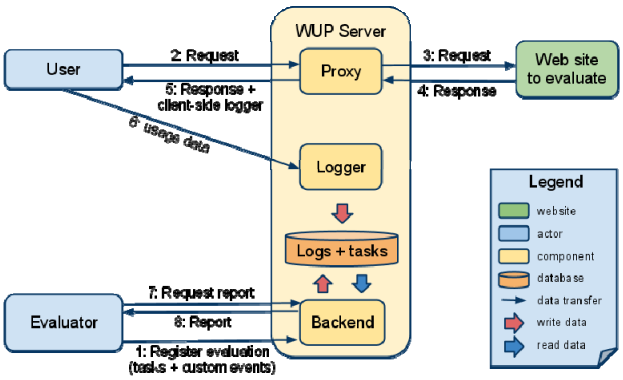

Figure 1. Overview of the approach proposed in the Web Usability Probe.

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. 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). 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 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 user is interacting with the page and when the user moves from one page to another; the server stores the logged data in a database, thus it is possible to extract all data in multiple formats to meet the evaluator's needs.

This approach satisfies a number of requirements about evaluation tools: 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, amongst 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 according their position in the Web page structure (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 Web Usability Probe 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 issue 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 occur, for instance, via AJAX, which can result in new UI elements. Our tool addresses these cases by detecting the events triggered when the DOM Tree is changed, which can then be reported in the timeline, allowing the evaluators to verify where and when they occurred.

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, a description, the evaluator 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.

[1] http://jquery.com

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

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, a description, the indication of the URL where the task should be started, whether it can be skipped, and if its performance depends on some other task. A dependency means that the other task should be performed first than the current one.

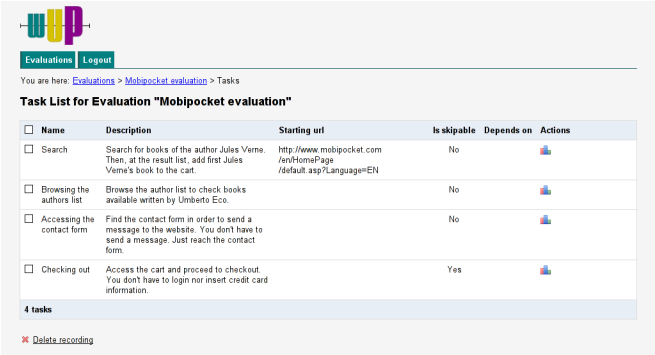

Figure 2. An example of task list for a user test.

These features allow remote participants to clearly understand what they should do: 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).

Our tool supports logging any standard event[3], jQuery Events[4], touch, gestures, and accelerometer events present at the Safari API[5]. The set of events observed by the tool is shown by grouping them by their type, mainly the device that can generate them: 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 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.

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 using the 'scroll lock' UI control. In this way the evaluator can compare the optimal behaviour with that of the actual users in order to see whether there was some particularly long interaction, due, for example, to difficulty in understanding how to proceed, or some errors that indicate some usability issue. The zoom level of timelines can be interactively set in order to identify the optimal representation scale.

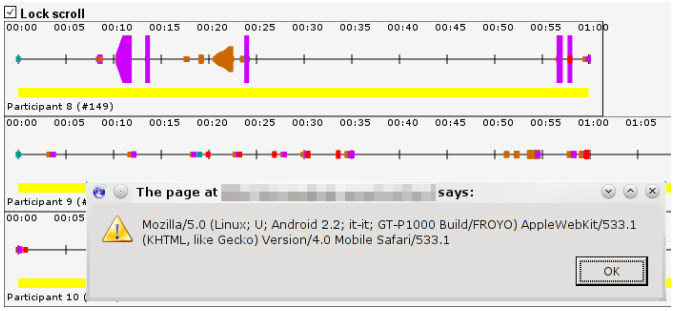

Figure 3. Example of timelines generated by the tool in order to represent usage data.

In addition, the evaluator can easily select the types of events to display on the timelines according to the needs of each analysis. Figure 4 shows the event selection tool of the accelerometer and touch event types, both typical of mobile devices.

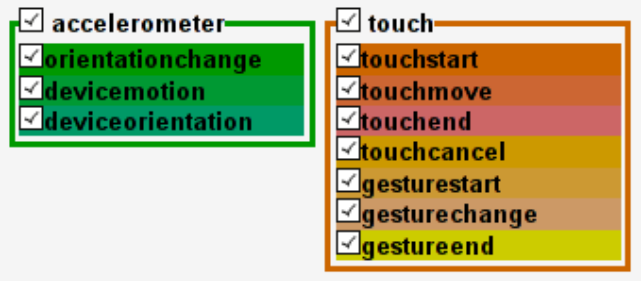

Figure 4. A detail of the event selector feature.

For each timeline the evaluator can also access the User Agent string sent by the user's browser, to easily identify the browser and the device. This is essential information since different devices may lead to a very different user experience and thus a different way to interact with the same page. This information is available clicking on the label "Participant X" (where X is the participant's number) on the bottom left corner of each timeline, and is shown in a popup window (Figure 3).

Our proposal for representing event streams in timelines also uses the height of the time markers present in the timelines to represent the repetitions of a certain event in an element. This allows evaluators to identify useless actions as well as to check repetitive mouse movements over poorly designed links, misguided clicks on non-clickable elements, 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

[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

categories visualization, the timeline uses different colours for different sets of events, for instance, mouse-related events are indicated with the same colour, keyboard-related events in a different colour and so on.

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 Dallas / Fort Worth, Texas airport Web site (http://www.dfwairport.com/mobile/index.php) for mobile devices.

The tasks to be performed at the airport Web site, which were specified in the configuration of the Web Usability Probe tool, were the following:

- Check the United Airlines departure flights to Chicago
- Find the United Airlines terminal / gate
- Find the list of the shops located in the United Airlines terminal

The mobile version of the Web site has a considerably simplified content with respect to the desktop version. Figure 5 left shows the home page.

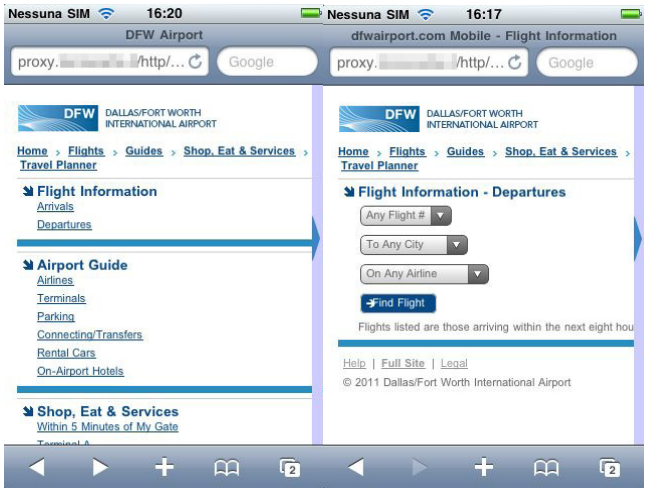

Figure 5. The application considered in the case study.

5.1 Check departure task

The Web site allows users to access all the departure flights but there is no page that contains such information accessible through a specific link. Users must use a form with three pull-down menus (see Figure 5 right), which allow them to filter the relevant flights. The menus also provide the possibility to filter no content. Thus, to find the United Airlines flights to Chicago the users should find the page with the form, and then select Chicago and United Airlines in the corresponding pull-down menus. In order to better analyse the logs we decided to define some custom events:

- *clickOnDeparture* is generated when the user selects the link pointing to the page with the form
- *citySelected* is generated when the user selects an arrival town
- *airlineSelected* is generated when the user selects an airline
- *formOk*, is generated when the user sends the form, only if the preconditions are satisfied: *citySelected* and *airlineSelected* must occur at least once before *formOk* is generated, which means that the user has to first select arrival town and airline;
- *formSubmitted*, is still associated with the form submitted event but it has no precondition.

Figure 6 shows an optimal sequence, obtained through an Apple iPod with iOS 4.2.1 and Safari browser. It is possible to note a high presence of accelerometer events generated by a mobile user. The GPS was turned off in this session.

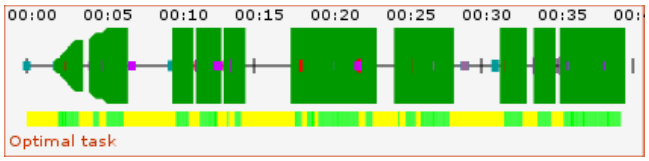

Figure 6. Visualization of Timeline with many accelerometer events.

The visualization tool allows evaluators to hide irrelevant events and thus, if we hide the accelerometer events, we obtain the representation in Figure 7, which is more readable.

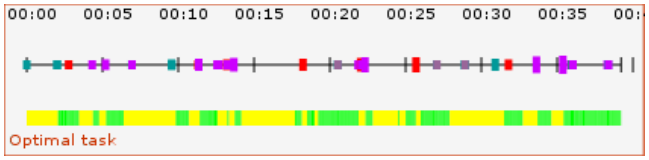

Figure 7. Visualization of Simplified Timeline.

The Safari browser detects the interactions carried out using the touch screen as normal mouse events, represented by purple markers. In addition, whenever users rotate the device to change the orientation from horizontal to vertical or conversely, the browser generates a resize event, with width and height values of the window exchanged, which falls within the System category, whose events are marked in red. The evaluator, in the recording of the optimal session for this first task, visited three pages and for each of them changed the device orientation, a clear sign of the utility of such a feature when you are using a screen of small dimensions.

One of the goals of Web Usability Probe is to ease the comparison between various user sessions in order to better identify behaviours that diverge from the optimal performance. If in the optimal session there is an event whose absence in the user session highlights an error in the task performance, then it is sufficient to appropriately configure the event selector to show that event (and eventually other relevant events) in the timeline.

Consider Figure 8, where the registration of the first participant in the evaluation is compared with the optimal session. The filter has been configured to show the events *pageviews*, *change*, *finishtask* and *formOk*. The latter, as described above, indicates that the user

has interacted with both drop-down boxes needed to successfully complete the task. However, in the second timeline this event (marked in light green in the first timeline) is not present. Nevertheless, at the end of the timeline there is the event *finishtask*, thus the user has erroneously considered the task as successfully completed.

Observing the events preceding the end of the task, it is clear the absence of the event change on the drop-down menu associated with the destination city. If the user had realized the error in the search results, he could have used the browser functions to return to the previous page and fill in the form properly. There are two possible explanations for the decision to consider the task as successfully completed: the task was not expressed clearly enough; the page containing the results of the search does not show the search parameters used properly.

It is not possible to verify the first case, but the fact that most other participants correctly selected an item from both drop-down menus tends to discount this hypothesis.

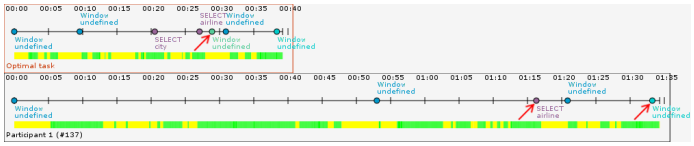

Figure 8. The first user made an error filling in the form.

Instead, we can analyse the second case. Figure 9 shows a comparison of search results in both cases: on the left the result of proper research, on the right the results of a search in which the filter on the destination city was omitted.

The two pages are very similar. The difference is that the page on the left shows only the top three results of a longer list, which can be seen by scrolling the page down, but unfortunately the display size of the device used by the user (an Apple iPhone, as shown by the User Agent string) allows users to see immediately only the first three, which coincidentally are exactly the flights to Chicago. Moreover the absence of visual indications of the page height (which in a desktop browser is done by the vertical scroller, not visible on the iPhone until the user “taps” on the screen) prevented the user from understanding the amount of information contained in the page.

This page has clearly a usability problem, since it does not provide adequate feedback to the user about the operation performed, and it would be appropriate to show in the top of the page a summary of the parameters used for the search clearly stated.

5.2 United Airline Location Task

The second task has no specific starting page, so when users start its performance they should be on the page where they concluded the previous task. If this is done properly, this page is the one containing the research results of the United Airlines flights leaving for Chicago. The purpose of this task is to check the usability of the Web site concerning the search for generic information.

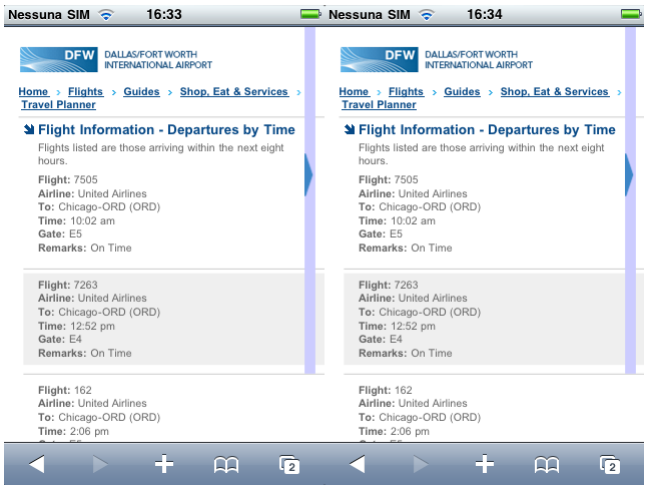

Figure 9. Search result comparison.

The evaluator - of course - successfully completed the first task, and thus begins the course of the second task from the page containing the results of the flights search. The site provides information about each airline by devoting to each of them a small information page. The shortest path to reach this page from the point where the evaluator starts is to follow the link "Guides" in the header, and then use the link "Airlines" on the next page. This link takes the user to a list of all the airlines, the last of which is United Airlines. Following the link leads to a page containing information about the company, including the information sought: the United Airlines is based in Terminal E. In Figure 10 you can see the sequence of pages visited by the evaluator to find the information sought in the most efficient way.

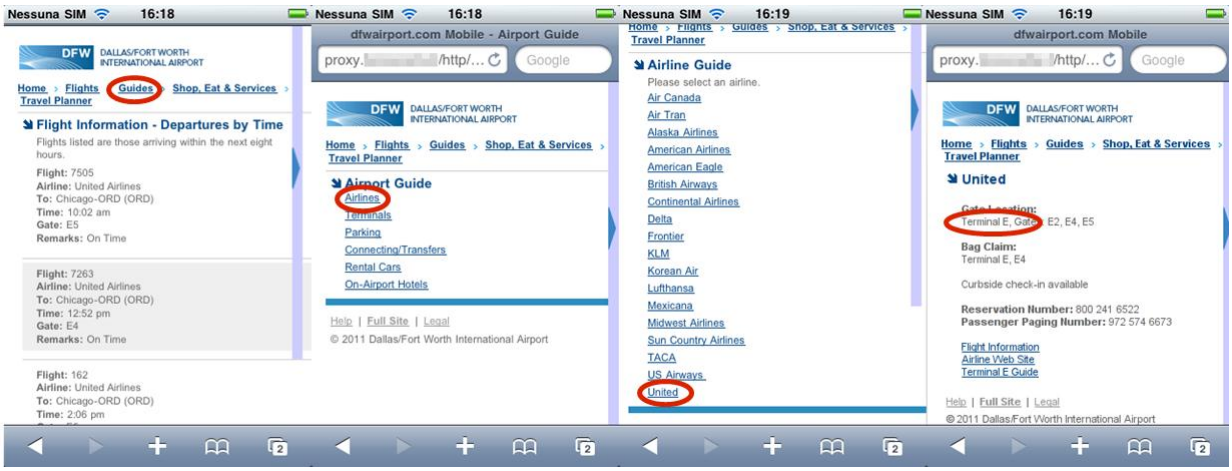

Figure 10. Sequence of pages visited by the evaluator.

The actions taken by the evaluator allow him to complete the task in about thirty seconds, as seen from the timeline shown in Figure 11. The three blue markers represent *pageviews* events, and are exactly the three page changes made. The green marker represents a semantic Custom event (*clickOnAirlines*) that was prepared for this task, and that is generated at the event click on the link that leads to the list of air companies. We chose this combination because such link is present in several places of the Web site and represents the minimum step to access the information sought and conclude that task successfully.

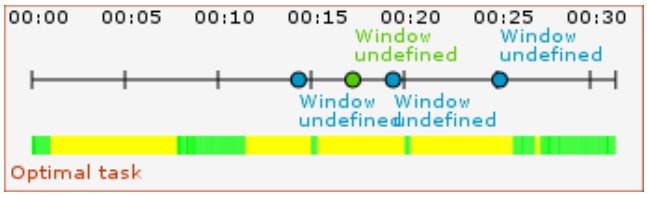

Figure 11.Timeline relating to the evaluator's performance for the second task.

It should be noted that the timeline in Figure 11 shows only four events simply because, as a mere matter of readability, it was decided to display the most relevant information for understanding the shortest path.

In the following we will analyze one timeline different from that generated by the evaluator. The first user successfully completed the first task, then, like the evaluator, started from the page containing the results of the flight departure search. Just like the evaluator, this user decided to follow the link "Guides" included in the page header, and the path is identical to the page containing the information sought.

When he reached this page, he unexpectedly decided to skip to the homepage from which, after about a minute, he again used the link to the page with the list of airlines and repeated the same steps, until completing the task. The entire sequence is illustrated in the timeline of Figure 12, which shows that the two *clickOnAirlines* events (which are the green markers) indicate the two times when the user decided to access the list of airlines. The question is: what caused the user to go to the home page and repeat the same path first to convince himself that he had found the information sought? It seems that the first time he considered the information in the page unsatisfactory or inadequate to the task proposed.

Figure 12. The user followed two different paths.

5.3 Shopping Task

In this case the optimal performance recorded by the evaluator consists of just two steps. The evaluator had concluded the previous task on the page containing the information on United Airlines, including information about the terminal (Terminal E). The evaluator then clicked the link in the header of all pages labelled "Shop, Eat & Services", which leads to a page where users can choose from one of these five terminals or access a search engine that helps users to find the services within five minutes of their gate.

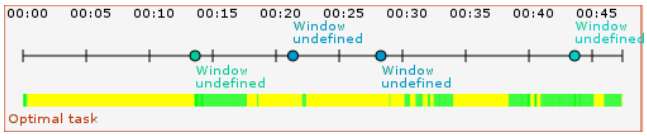

Figure 13. The evaluator's performance for the third task.

The evaluator correctly selected the link "Terminal E", as associated with United Airlines, and this led to the corresponding page with the link "Shops", which allows users to access the list of stores in Terminal E.

The timeline of the evaluator's performance is shown in Figure 13. The event selector has been configured so as to show only *pageviews*, *startTask* and *finishtask* events, as the other events were not considered relevant for this analysis.

We now analyse the timeline of a user who successfully completed the task. He started performing this task from the same page as the evaluator.

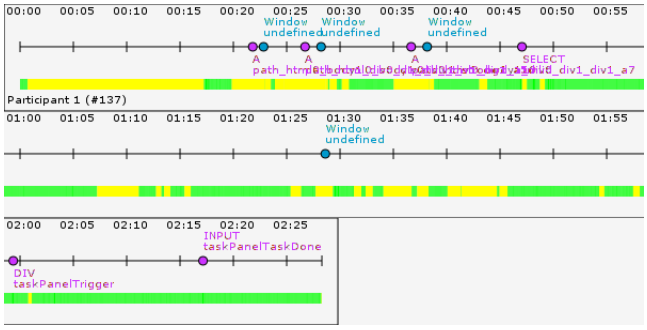

Figure 14. The pageview sequence of the first user performance.

His first two choices of the links to follow were the same, then the user reached a page where, instead of choosing the link "Shops", preferred to follow the link labelled "Within 5 Minutes of My Gate". The page loaded by following this link contains a form that allows users to search for services within five minutes from the Terminal / Gate selected. There is a drop-down menu to select the Terminal, and optionally users can enter the number of the desired Gate in the text field (see Figure 15).

This page has an obvious usability problem: the user in fact reached this position after clicking the link "Terminal E", and accessing first another page. The fact that this page shows a form that allows users to select a Terminal disorients the user, who had already "selected" the desired Terminal via the link previously.

From the timeline shown in Figure 14 one can see that once the user arrives in the page containing the form (*pageviews* event, third blue marker) he uses the drop-down menu to select a terminal (event with SELECT label). The form is not sent, however - in fact there is no submit event - and about a minute and a half after there is a *pageview* event (another blue marker) without any click before. The details not visible in the picture show that the page loaded is the same that the user was already seeing, for which it is likely he did a reload using the browser button. After the page reload the user decided to open the tool's control Panel and declare the task complete.

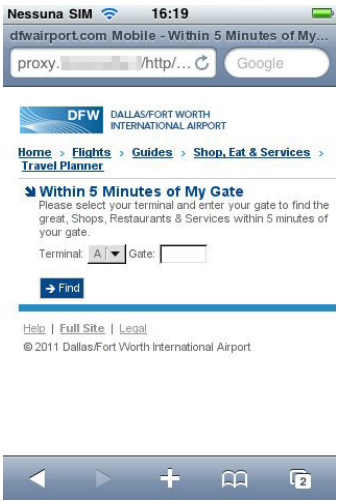

Figure 15. The service search page.

6. CONCLUSION AND FUTURE WORK

This paper presents 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 analyse the data collected from a usability perspective. The tool also allows the end users to freely access the Web applications with any browser-enabled mobile 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.

The extensive events vocabulary, including events typically generated by mobile devices, allows the evaluator to identify usability problems that can occur when using Web applications with devices other than the traditional desktop computer.

As future work we plan to extend the tool in order to better support the possibility of automatically identifying and highlighting potential usability issues. We also plan to investigate the use of the optimal sessions as a training set in data mining processes in order to find similar behavioural sequences and thus isolate those that are very different.

ACKNOWLEDGMENTS

We would like to thank *Fundação de Amparo à Pesquisa do Estado de São Paulo* (grant #2009/10186-9) for the support to the last author.

REFERENCES

[1] Google: Google Analytics. http://www.google.com/analytics/index.html. (2009)

[2] H. R. Hartson, J. C. Castillo, J. T. Kelso, W. C. Neale: Remote Evaluation: The Network as an Extension of the Usability Laboratory. CHI 1996: 228-235.

[3] D. M. Hilbert, D. F. Redmiles: 2000. Extracting usability information from user interface events. ACM Comput. Surv. 32, 4 (Dec. 2000), pp. 384-421.

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

[5] I. Malý, P. Slavík: Towards Visual Analysis of Usability Test Logs Using Task Models. TAMODIA 2006: 24-38

[6] L.Paganelli, F.Paternò: Tools for Remote Usability Evaluation of Web Applications through Browser Logs and Task Models, Behavior Research Methods, Instruments, and Computers, 2003, 35 (3), pp.369-378, August 2003

[7] S. Propp, P. Forbrig: ViSE - A Virtual Smart Environment for Usability Evaluation. HCSE 2010: 38-45

[8] C. J. Costa, J. P. Novais Silva, M. Aparìcio: Evaluating Web Usability Using Small Display Devices. SIGDOC 2007: 263-268

[9] T. Carta, F. Paternò, V. F. de Santana: Web Usability Probe: A Tool for Supporting Remote Usability Evaluation of Web Sites, INTERACT 2011 Proceedings, LNCS, Volume 6948, pp. 349–357, Springer, Lisboa, September 2011