Remote Evaluation of Mobile Applications

Fabio Paternò, Andrea Russino, and Carmen Santoro

ISTI-CNR, Via G. Moruzzi, 1,
56124 Pisa, Italy
{Fabio.Paterno, Andrea.Russino, Carmen.Santoro}@isti.cnr.it

Abstract. In this paper we present a method and a supporting environment that allows remote evaluation of mobile applications. Various modules have been developed in order to gather contextual data about the usage of such applications in different environments. In addition, issues related to how to visualise usability data have been addressed in order to support the designers' work in analysing such data.

Keywords: Remote usability evaluation, Usability in Mobile Applications, Representation of Usability Data.

1 Introduction

In remote usability evaluation, evaluators and users are separated in space and possibly time during the evaluation [1]. This type of evaluation is becoming increasingly important for the number of advantages it offers. Indeed, it allows the collection of detailed information on actual user behaviour in real contexts of use, which is especially useful in contexts in which it is not possible (or convenient) having an evaluator directly observing or recording the session. In addition, the fact that the users carry out the evaluation in their familiar environments contributes to gain more 'natural' users' behaviour.

In order to have a complete picture of what users did during the session and derive consequent conclusions about the usability of the application, it is crucial for the evaluators to reconstruct not only the interactions that users carried out during the session, but also the contextual conditions that might have affected the user interaction itself. Indeed, if such conditions are not completely known, the evaluators might draw incorrect conclusions about the usability of the considered application. This problem becomes even more difficult to address when dealing with mobile applications. Indeed, while for desktop applications the lack of co-presence between users and evaluators can be compensated to some extent by equipping the test environment with devices such as web cams, mobile applications require different solutions that are able to flexibly support evaluation in different contexts without being too obtrusive on the user side. When dealing with remote applications for mobile devices, there are some additional problems that make it more difficult to gather data to remotely evaluate such applications. For instance, we have to take into account the more limited capability of mobile devices, which imposes constraints on the kinds of techniques to be used for tracking user interactions. In addition, there is the further problem of detecting the environmental conditions in which the session takes place.

In this paper we discuss a novel extension of a tool able to remotely process multimodal information on users interacting with desktop applications. The new solution is enriched with the possibility of tracking and evaluating also user interfaces of mobile applications, including the detection of environmental conditions that might affect user interaction (e.g.: noise in the surrounding environment, battery level consumption, etc.). The new tool, MultiDevice RemUsine, is able to identify where users interactions deviate from those envisioned by the system design and represented in the related task model. In addition, we also improved the graphical representations that are provided to the evaluators for visualizing the gathered data. Indeed, especially when dealing with a large amount of information, it is very important to use effective representations highlighting relevant information so as to enable evaluators to better identify potential issues and where they occur.

The structure of the paper is the following one: in the next section we discuss related work, and next we introduce our general approach. Then, we present the main features of the additional component (Mobile Logger) we developed for supporting the detection of user interactions with mobile applications and environmental conditions that might affect the performance of user's activity. In the following, we also discuss the issue of more effective visualisation techniques for representing the data that have been collected regarding the user activity. Lastly, we conclude with some remarks and indications for future work.

2 Related Work

Interest in automatic support for usability evaluation is rapidly increasing [2], especially as far as the remote evaluation is concerned, because, on the one hand, it is important that users interact with the application in their daily environment, but, on the other hand, it is impractical to have evaluators directly observe users' interactions. In [1] one of the first examples of remote-control evaluation is described. The remote-control method checks a local computer from another computer at a remote site. Using this method, the evaluator's computer is located in the usability lab where a video camera or scan converter captures the users' actions. The remote users remain in their work environment and audio capture is performed via the computer or telephone. This is an example of a flexible technique for asynchronous remote evaluation which is restricted to be used on desktop systems due to some software limitations.

Other studies [3] have confirmed the validity of remote evaluation in the field of Web usability. The work by Lister [4] has been oriented to using audio and video capture for qualitative analysis performed by evaluators on the result of usability testing. Other works have highlighted the importance of performing a comprehensive evaluation able to take into account data derived from multiple sources, and the consequent need to provide analysts from a variety of disciplines (each using distinct sets of skills to focus on specific aspects of the problem) to work cooperatively, in order to adequately gain insight into large bodies of multisource data [5]. A more recent work [6] compares three methods for remote usability testing and a conventional laboratory-based think-aloud method. The three remote methods are a remote synchronous condition, where testing is conducted in real time but the test monitor is separated spatially from the test subjects, and two remote asynchronous conditions, where the test monitor and the test subjects are separated both spatially and temporally. The authors claim that the two methods identified almost the same number of usability problems, and users spent the same time completing the tasks. The asynchronous methods are more time-consuming for the test subjects and identify fewer usability problems, yet they may still be worthwhile because they relieve the expert evaluators from a considerable amount of work, and enable collection of use data from many participants.

Since most applications have been developed for the desktop, the majority of remote evaluation methods have addressed this type of platform. Only a few proposals have been put forward for remote evaluation of mobile applications. An example in this area is the paper by Waterson et al. [7], where the authors discuss a pilot usability study using wireless Internet-enabled personal digital assistants (PDAs), in which they compare usability data gathered in traditional lab studies with a proxy-based logging and analysis tool. They found that this remote testing technique can more easily gather many of the content-related usability issues, whereas device-related issues are more difficult to capture. In [8] the authors describe a usability evaluation study of a system that permits collaboration of small groups of museum visitors through mobile handheld devices (PDAs). As usability evaluation methodology, they propose a combination of a logging mechanism and an analysis tool (the ColAT environment [9]), which permits mixing of multiple sources of observational data, a necessary requirement in evaluation studies involving mobile technology, when users move about in physical space and are difficult to track. The museum system evaluated is based on a client–server architecture and an important characteristic of the application is that the server produces a centralized XML log file of the actions that occur during the visit, and this log file can be combined with a video recording of the visit allowing evaluation of activity during the visit. The methodology was able to deliver data useful for deriving quantitative information (e.g. total and average times for solving the puzzles, etc.), aspects related to group activities (number of exchanges between the group, strategies used for solving the puzzles, ..), behavioural patterns of participants. In general, while logging tools for mobile devices have already been proposed [10], we wanted to develop a novel solution for this purpose able also gather data useful to better identify the context of use related to aspects such as environmental conditions, connectivity and so on.

Another emerging need in this area concerns tools able to support effective representations for enabling the evaluators to analyze the evaluation data collected. To aid analysis of the gathered usability test data, the WebQuilt [11] visualization provides filtering capabilities and semantic zooming, aiming to allow the designer to understand the test results at the overall view of the navigation graph, and then drill down to sub-paths and single pages. A first attempt to provide visual representations linking task models and log data was provided by Maly and Salvik [12]. In this paper, we present a representation sharing similar goals that is based on our experience in order to provide useful information for evaluators.

3 General Approach

Our approach is mainly based on a comparison of planned user behaviour and actual user behaviour [13]. Information about the planned logical behaviour of the user is contained in a (previously developed) task model, which describes how the tasks should be performed according to the current design and implementation. The task model can be built in various way. It can be the result of an interdisciplinary discussion involving end users, designers, application domain experts, and developers. There are also reverse engineering techniques able to build automatically the system task model of Web pages starting with their implementation.

The data about the actual user behaviour are provided by the other modules (eg: the logging tools), which are supposed to be available within the client environment. An overview of the general approach is described in Figure 1. A logging tool, which depends on the type of application considered, stores various user or system-generated events during the user session. In addition, other sources of information regarding the user behaviour can be considered, such as Web Cams showing the actual user behaviour and face expressions or eye-trackers detecting where the user is looking at.

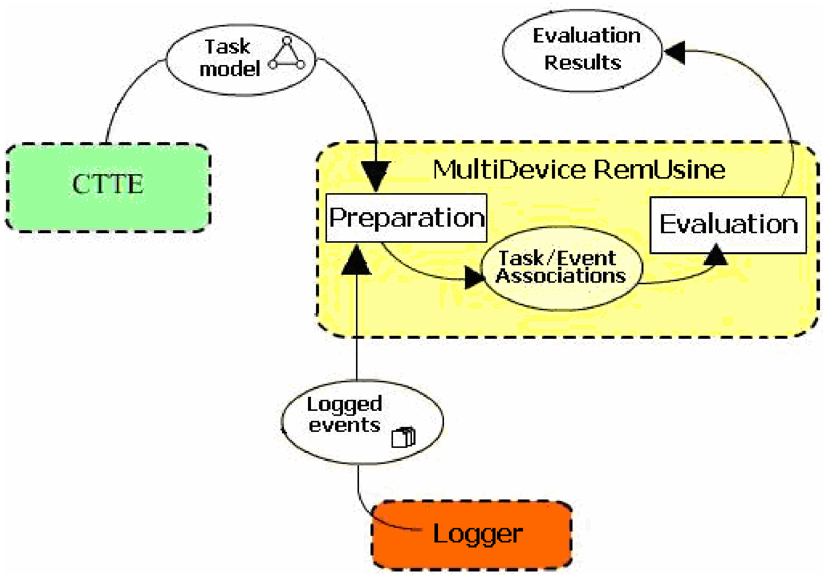


Fig. 1. The architecture of Multi-Device RemUsine

As for the expected user behaviour, CTT [14] task models are used to describe it by their graphical representation of the hierarchical logical structure of the potential activities along with specification of temporal and semantic relations among tasks. It is worth pointing out that, with the CTT notation used, the designer might easily specify different sequences of paths that can correspond to the accomplishment of the same high-level task: this is possible thanks to the various temporal operators available in the CTT notation, which also include, for instance, the specification of *concurrent*, multitask activities, or activities that *interrupt* other ones. On the one hand, for the designer is quite easy to specify in a compact manner even a complex behaviour, on the other hand the behaviour of such operators is automatically mapped by the underneath engine into all the corresponding possible paths of behaviours.

In order to enable an automatic analysis of the actual user behaviour identified by the sequences of actions in the logs against the possible expected behaviours described by the task model there is a preparation phase. In this phase the possible log actions are associated with the corresponding basic tasks (the leaves in the task model). Once this association is created then it can be exploited for analysing all the possible user sessions without further effort. In this way, the tool is able to detect whether the sequence of the basic tasks performed violates some temporal or logical relation in the model. If this occurs then it can mean that either there is something unclear on how to accomplish the tasks or the task model is too rigid and it is not able to consider possible ways to achieve user goals. Thus, by comparing the planned behaviours (described within the task model) with the information coming from log files, MultiDevice RemUsine is able to offer the evaluators useful hints about problematic parts of the considered application. To this regard, it is worth pointing out that the tool is able to discriminate to what extent a behaviour deviates from the expected one (for instance, whether some additional useless tasks have been performed but they did not prevent the user from completing the main target task, in comparison with other cases in which the deviation led to unsuccessful paths).

4 Mobile Logging

With mobile interaction there are some contextual events that should be considered since they can have an impact on the user's activity. Among the relevant events that might be taken into consideration there are the noise of the environment, its lightness, the location of the user, the signal power of the network, as well as other conditions related to the mobile device, e.g. the residual capacity of the battery.

When we deal with usability evaluation in which stationary devices are used, the contextual conditions under which the interaction occurs and involving the location of the user remain unchanged over the experiment session. In this case, the information about user interaction might be sufficient. When we consider interaction with mobile devices, since the interaction occurs in an environment that can considerably change not only between two different executions, but also within the same execution, this is no longer valid. Thus, it is important to acquire comprehensive and detailed information about the different aspects of the context of use in which the interaction currently occurs since they might have an impact on the user's activity.

Each of these variables, as well as combinations of them can affect the user interaction, and in our tool we developed a separate module for detecting to what extent each of these components can affect the user interaction. Currently, we consider aspects connected with the current position of the user (the position itself, together with the noise and lightness of the surrounding environment, according to such a position) together with other variables, which are more connected to objective,

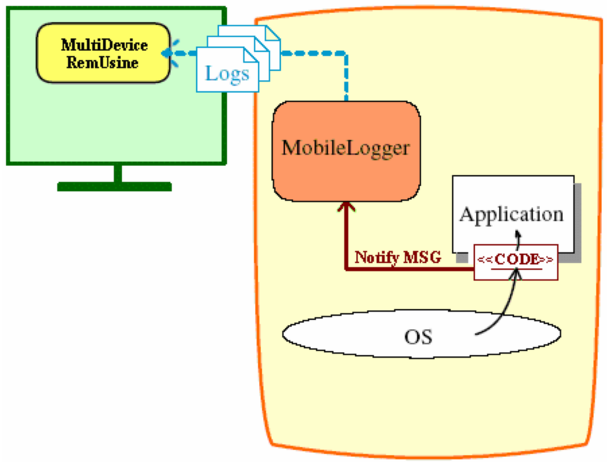


Fig. 2. The architecture of the logging system

technological requirements for enabling/disabling features involving the PDA overall working (battery level and network signal). Differently from the latter ones, the impact of the first aspect has to be put in relation with how the users might subjectively perceive the user interface, therefore, managing them could be harder.

The general architecture of the tool is shown in Figure 2. The core of the logging tool is the Mobile Logger, which is in an intermediate position between the application to be evaluated and MultiDevice RemUsine. Indeed, on the one hand it has to communicate with the operating system to detect events and track the user's activity, and, on the other hand, it has to record and communicate the logged data to the evaluation tool. In the following sub-sessions we will detail both aspects in more depth.

4.1 Tracking the User's Activity: Events and Messages

The task of tracking the activity of the user is carried out by a procedure that is executed by the application evaluated, which uses libraries included in the operating system to detect events, and which exploits an inter-process communication model based on exchanges of messages. The execution of an interactive graphical application is driven by events, notified in Windows systems by means of messages. Indeed, WindowsCE, like other Windows systems, is an operating system based on the "push" mechanism: every application has to be coded to react to the notifications (namely: messages) that are received from the operating system. Each window has a window procedure that defines the behaviour of the component.

Therefore, it is theoretically possible to derive all the information associated with the interaction from such messages. However, it is worth noting that not all messages received by the window procedure of a window/component are useful to reconstruct the user interaction. Indeed, there are messages that do not directly regard the user interaction, for instance the WM_PAINT message forces the refreshing of the display of a certain component, but it is not triggered by the user. As a consequence, only a subset of the messages is considered for our purposes. Such set includes, for instance: WM_LBUTTONDOWN, a message that is received from every component as soon as a click event is detected on it; WM_KEYDOWN, a message that is sent to the component that currently has the focus as soon as the user presses a key on the keyboard.

The functionality to track and save all the interactions of the user with the system is not centrally delegated to a single module but instead distributed over multiple modules that track the activity of the user according to a specific aspect:

- **NoiseMod:** It is the module that has to track possible conditions that might interfere with the user activity on the audio channel. In order to track the conditions on the audio channel, this module executes at regular intervals of time a sampling of the audio. Depending on the samplings recorded, the value to be recorded in the log file is calculated.
- **PowerMod:** It is the module that monitors the battery consumption of the device. The values are saved as they are provided by the system, without performing any calculation on them.
- **LightMod:** It is the component that is in charge of tracking conditions that might interfere on the visual channel, for instance variations on the brightness of the surrounding environment.
- **SignalMod:** Some applications might depend on the availability of a communication network and on the intensity of the signal. In these cases, the task of recording the power of such a signal is delegated to this module.
- **PositionMod:** Some applications might be affected by the current position of the user. In this case, this module will track the location of the user and how it changes over the time.

These modules have been identified taking into account the possibilities opened up by the sensing technologies of current mobile devices. Such modules for gathering environmental data are dynamically loaded only if a logging session is started and the activation of these specific components is requested. They record events using a sampling frequency algorithm, which is able to *adapt* the frequency at which the sampling is taken.

Therefore, the sampling is not carried out at fixed time intervals. It starts with setting an initial interval of time in which events are acquired. Then, it proceeds in the following way: if, in the last interval of time no variation has been registered, the interval of time to be considered for the next acquisition becomes larger, otherwise it decreases (following an exponential law). This choice is based on the consideration that using a fixed interval of time for the sampling frequency might not be a good solution. For instance, if the sampling frequency is much smaller than the frequency at which the environmental condition changes, a more flexible algorithm can avoid the activation of useless event detection during some intervals of time, saving battery consumption, which is not an irrelevant aspect for mobile devices.

4.2 Saving the Logged Data

The tool receives notification messages from the application to be tested, and delivers XML-based log files in which the events are saved according to a specific structure that will be detailed later on in the section.

Therefore, Mobile Logger communicates with Multi-Device RemUsine through the log files: in such files the logging tool records the detected events, and from such log files Multi-Device RemUsine gets the information needed to reconstruct the user's activity.

The log file is an XML-based file, and it is structured into two main parts: a header and the list of events that have been registered by the logger. The header contains information related to the entire evaluation session, for instance, the username of the tester, the temporal interval spent performing the test, the application tested, the list of contextual aspects that have been registered and the related parameters of sampling.

The events are recorded according to the following structure: (temporal event, type of event, value), and they have been categorised into different classes: *contextual* events (all the events that have been registered as a consequence of a contextual condition); *intention* event (which is used to signal that the user has changed the target task, which has to be explicitly indicated); *system* event (the events generated by the system for replying to a user's action); *interaction* event, further specialised into different categories like: click focus, select, check, scroll, edit.

As an example, we can consider an application of the tool focusing on the use of information regarding noise and battery power. In this case, the tested application was a museum guide available on a PDA device. When the tool is activated it appears as shown in Figure 3(a): the user is supposed to fill identification information, then specify the aspects of the environment s/he is interested to consider, and also specify the target task (intention) that she wants to achieve (Fig. 3c), by selecting it from a list of high-level tasks supported by the application (Figure 3-b).

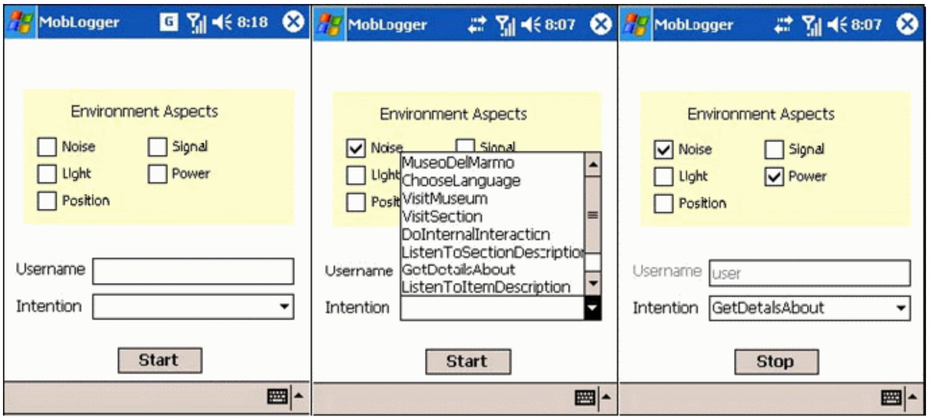
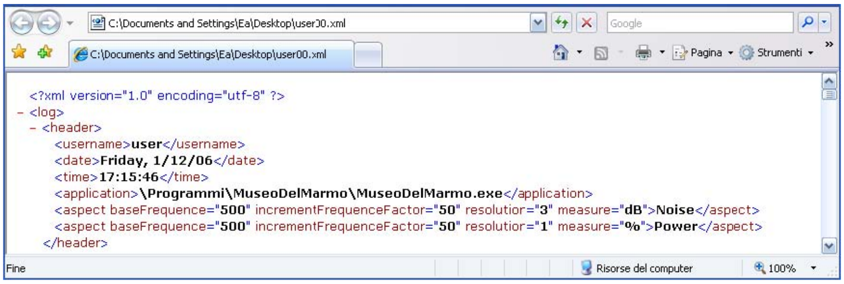


Fig. 3. The logging tool: (a) when it starts; (b) after selecting the environmental conditions of interest; (c) after selecting the intention

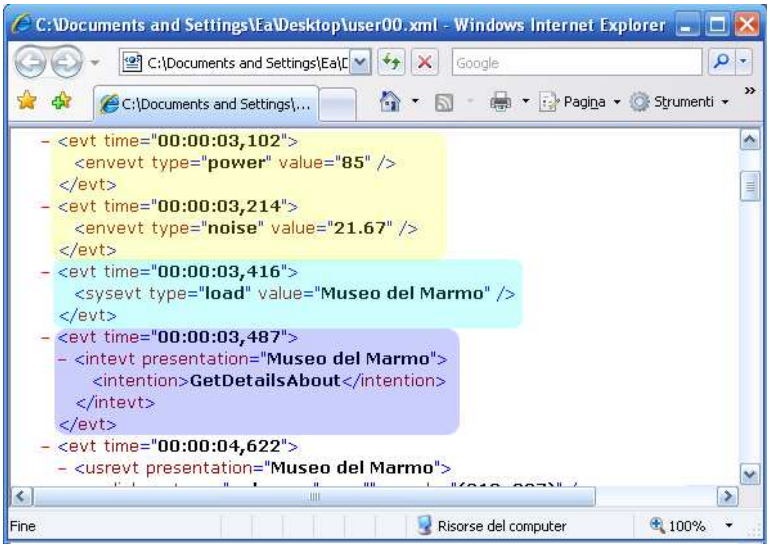
Then, after selecting the "Start" button the log file regarding the user's activity is created. Figure 4 shows an excerpt of the log file indicating the detection of the noise with an initial frequency of 500 ms and an increment factor of 50 ms. In addition, only variations not less of 3dB with respect to the previously detected values will be tracked. As for the battery level, the temporal parameters used are similar apart that the resolution is of only 1 percentage point.



```
  <?xml version="1.0" encoding="utf-8" ?>
- <log>
  - <header>
      <username>user</username>
      <date>Friday, 1/12/06</date>
      <time>17:15:46</time>
      <application>\Programmi\MuseoDelMarmo\MuseoDelMarmo.exe</application>
      <aspect baseFrequence="500" incrementFrequenceFactor="50" resolutior="3" measure="dB">Noise</aspect>
      <aspect baseFrequence="500" incrementFrequenceFactor="50" resolutior="1" measure="%">Power</aspect>
    </header>
```

Fig. 4. An excerpt of log file recorded by the MobileLogger

During the session evaluated, the user is supposed to interact with the application. Figure 5 shows an excerpt of the log file highlighting some events that have been registered and referring to the abovementioned scenario. From top to bottom, we have highlighted two environmental data regarding battery and noise; then, we have the notification of a system event (the loading of a window in the application), lastly, we have the notification of the selection of the target task (intention) made by the user.


C:\Documents and Settings\Ea\Desktop\user00.xml - Windows Internet Explorer
C:\Documents and Settings\Ea\D
Google
C:\Documents and Settings\...
Pagina
Strumenti
Fine
Risorse del computer
100%

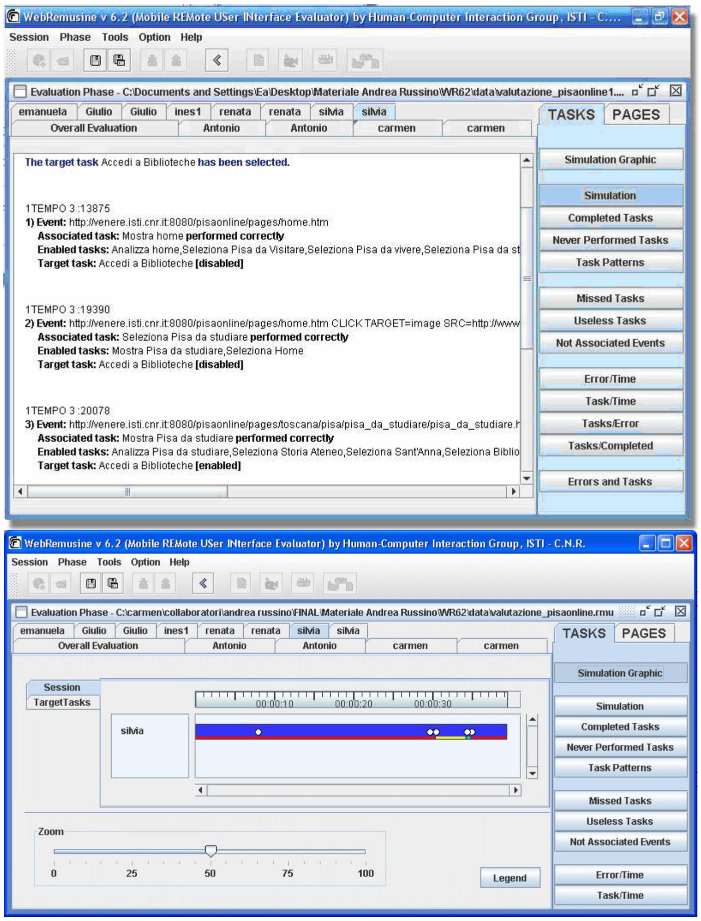
```
- <evt time="00:00:03,102">
    <envevt type="power" value="85" />
  </evt>
- <evt time="00:00:03,214">
    <envevt type="noise" value="21.67" />
  </evt>
- <evt time="00:00:03,416">
    <sysevt type="load" value="Museo del Marmo" />
  </evt>
- <evt time="00:00:03,487">
  - <intevt presentation="Museo del Marmo">
      <intention>GetDetailsAbout</intention>
    </intevt>
  </evt>
- <evt time="00:00:04,622">
  - <usrevt presentation="Museo del Marmo">
```

Fig. 5. Different types of events recorded by the Mobile Logger

5 Representing Usability Data

Usability evaluation is strictly connected to how the gathered data are represented and provided to the evaluator. Therefore, the choice about the particular representation to use is important for enabling an effective assessment by the evaluators. Indeed, the data that are gathered might produce a large amount of information, which, if not adequately represented, might become a burden for the evaluators rather than facilitating their work. A previous version of the tool (see [13]) already provided the possibility to visualise the data gathered during the evaluation session, so that the evaluators could analyse them and derive their results accordingly.

However, the evaluation reports were mainly textual, and therefore they were not able to highlight the main aspects effectively (see Figure 6-top part). The new version


WebRemusine v 6.2 (Mobile REMote USer INterface Evaluator) by Human-Computer Interaction Group, ISTI - C....
Session Phase Tools Option Help
Evaluation Phase - C:\Documents and Settings\Ea\Desktop\Materiale Andrea Russino\WR62\data\valutazione_pisaonline1....
emanuela Giulio Giulio ines1 renata renata silvia silvia
Overall Evaluation Antonio Antonio carmen carmen
TASKS PAGES
Simulation Graphic
Simulation
Completed Tasks
Never Performed Tasks
Task Patterns
Missed Tasks
Useless Tasks
Not Associated Events
Error/Time
Task/Time
Tasks/Error
Tasks/Completed
Errors and Tasks
WebRemusine v 6.2 (Mobile REMote USer INterface Evaluator) by Human-Computer Interaction Group, ISTI - C.N.R.
Session Phase Tools Option Help
Evaluation Phase - C:\carmen\collaboratori\andrea russino\FINAL\Materiale Andrea Russino\WR62\data\valutazione_pisaonline.rmu
emanuela Giulio Giulio ines1 renata renata silvia silvia
Overall Evaluation Antonio Antonio carmen carmen
TASKS PAGES
Session
TargetTasks
00:00:10 00:00:20 00:00:30
silvia
Zoom
0 25 50 75 100
Legend
Simulation Graphic
Simulation
Completed Tasks
Never Performed Tasks
Task Patterns
Missed Tasks
Useless Tasks
Not Associated Events
Error/Time
Task/Time

The target task Accedi a Biblioteche **has been selected.**

1TEMPO 3 :13875
1) Event: http://venere.isti.cnr.it:8080/pisaonline/pages/home.htm
Associated task: Mostra home **performed correctly**
Enabled tasks: Analizza home,Seleziona Pisa da Visitare,Seleziona Pisa da vivere,Seleziona Pisa da st
Target task: Accedi a Biblioteche **[disabled]**

1TEMPO 3 :19390
2) Event: http://venere.isti.cnr.it:8080/pisaonline/pages/home.htm CLICK TARGET=image SRC=http://www
Associated task: Seleziona Pisa da studiare **performed correctly**
Enabled tasks: Mostra Pisa da studiare,Seleziona Home
Target task: Accedi a Biblioteche **[disabled]**

1TEMPO 3 :20078
3) Event: http://venere.isti.cnr.it:8080/pisaonline/pages/toscana/pisa/pisa_da_studiare/pisa_da_studiare.h
Associated task: Mostra Pisa da studiare **performed correctly**
Enabled tasks: Analizza Pisa da studiare,Seleziona Storia Ateneo,Seleziona Sant'Anna,Seleziona Biblio
Target task: Accedi a Biblioteche **[enabled]**

Fig. 6. Representing the evaluation results in the previous version of the tool (top) and in the new version (bottom)

of the tool offers graphical visualisations of the data gathered, which can be managed more easily by the evaluators (an example is shown in Figure 6-bottom part), thereby representing a step forward with respect to the previous visualisation technique. The new graphical representations will be described in further detail in the next sections. One of the most important points to bear in mind when deciding the technique to use for representing evaluation data is that such representation should make it easy to identify the parts of the application where users encounter problems. Therefore, the information represented is effective insofar as it is able to highlight such information, and consequently enable the evaluator to draw conclusions about the usability of the application. One relevant aspect for effectively reconstructing a user session is providing data according to its evolution over the time. In addition, evaluators should be able to easily carry out comparisons between the behaviour of different users; therefore, the use of graphical representations (rather than e.g. lengthy text-based descriptions) can also provide the evaluators with an overview of the collected data and allow them to compare data on different users.

Such considerations led to the type of representation we have investigated to represent usage data, the *timelines*. In particular, we identified three types of timelines:

- *Simple Timeline*: linear representations of the events that have been recorded;
- *State-timeline*, which is an extension of the first one, enriched with information about the current state of the target task, which is represented through different colours associated with *disabled*, *enabled* or active;
- *Deviation-timeline*, which is a representation of the registration over three different parallel levels, in which squared elements indicate the level of deviation from a sort of "ideal" path.

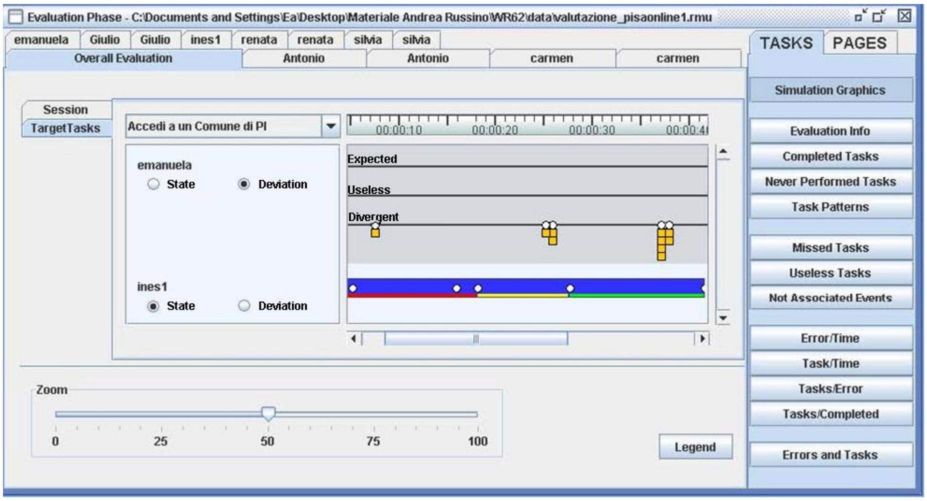


Fig. 7. Deviation Timeline and State Timeline visualized in the tool

In particular, we developed a number of panels in which not only whole sessions but also segments of them are represented both in relation to a single user and group of users. Figure 7 shows an example of both representations (the white circles identify the temporal occurrence of basic tasks whose performance is detected by an analysis of the log file), each one associated with a different user. The lines contained within the State Timeline identify the evolution of the state of the target task that has been selected: *disabled*, *enabled*, *active*, which are represented in different colours. For the Deviation timeline (see Figure 7), each square represents a degree of deviation from the ideal path which was supposed to be followed by the designer.

As Figure 7 shows, the evaluators can select the preferred type of representation and specify if they are interested to visualise data associated with a whole session or associated with single tasks. The two solutions are basically similar, but the second one is especially useful when the evaluator wish to perform some comparisons, because the selection of a single task provides information independent of absolute times. In this way, a target task explicitly selected by a user after a certain period of time from the start of the session will be perfectly lined up with another one from a different user, which started exactly at the beginning of the session. Within the timelines it is possible not only to identify when the task occurred, but also the type of task that occurred, through the use of a particular colour (see Figure 8).

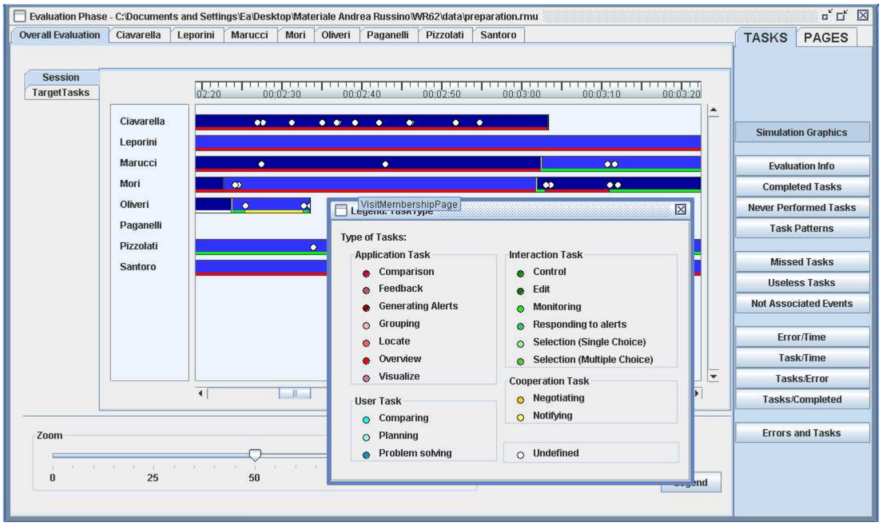
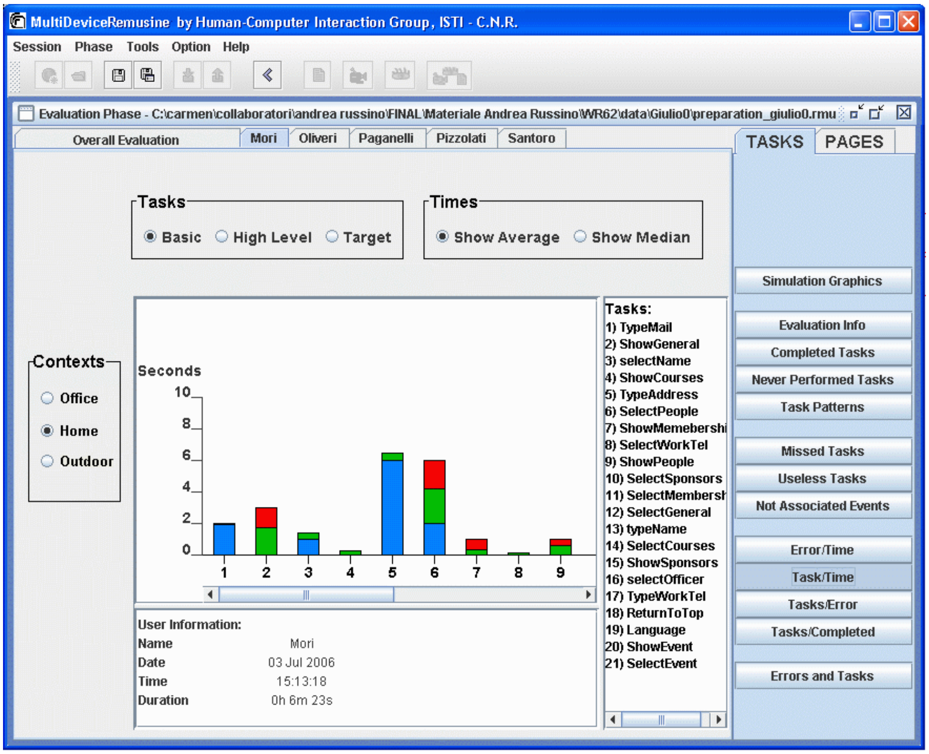


Fig. 8. The different types of intentions represented through different colours

The knowledge about the different contexts in which the user session evolved is relevant for deriving whether any condition might have interfered with the user's activity, then it is important for completely reconstructing the conditions in which the experiment took place. Contexts that are relevant for the evaluation might physically correspond to a certain place and situation, but they might also be associated with the variation of some specific aspects (for example, noise or light) even if the user is still in the same position. Then, two basic manners for defining contexts can be considered: on the one hand, there is the possibility to explicitly list the contexts that are judged relevant and define each of them in terms of the various contextual dimensions we are interested in. For instance, it might be the case that we are interested to only two specific contexts, one characterised by high level of noise, light and network connectivity (such as the office), another one characterised by low levels of noise, medium level of light and low level of network connectivity, which might be at home. On the other hand, we might wish to specify in other cases just the variations that determine the change of context, e.g. the variation of a certain parameter beyond a specific threshold value or percentage. For instance, we might want to investigate the impact on the usage of the application whenever a reduction/increase of 30% in light is registered in the environment.



Tasks: [✓ Basic] High Level Target

Times: [✓ Show Average] Show Median

Contexts: Office [✓ Home] Outdoor

Tasks:
1) TypeMail
2) ShowGeneral
3) selectName
4) ShowCourses
5) TypeAddress
6) SelectPeople
7) ShowMemebershi
8) SelectWorkTel
9) ShowPeople
10) SelectSponsors
11) SelectMembersh
12) SelectGeneral
13) typeName
14) SelectCourses
15) ShowSponsors
16) selectOfficer
17) TypeWorkTel
18) ReturnToTop
19) Language
20) ShowEvent
21) SelectEvent

User Information:

Name	Mori
Date	03 Jul 2006
Time	15:13:18
Duration	0h 6m 23s

Fig. 9. Possibility of selecting task performance information related to a specific context of use

Once the different contexts have been identified, various aspects can be analysed by the evaluator. For instance, if the achievement of a certain goal is obtained by carrying out the related activities partially in an environment and partially in other environments, it might be interesting to see how the performance of a certain task evolves. In other cases, it might be interesting for the evaluator to carry out an analysis that takes into account a specific context and understand the evolution of the sessions in that specific context. For instance, it might be useful to understand the amount of time the user has spent in a specific environment and the number of tasks that have been completed in such an environment. In Figure 9 the time spent for carrying out a certain task in a specific context is visualised: the evaluator can select the specific context ("Home" in Figure 9) and then the tool shows how much time was needed by the user in order to carry out the tasks in that specific context.

6 Conclusions and Future Work

In this paper we describe an extension of a tool for remote usability evaluation. The extension aimed at supporting the evaluation of mobile applications through a logging tool also able to detect some environmental conditions that might affect the user interaction, which can be useful information for the evaluator. The resulting environment is able to support usability evaluation of applications that are executed in different types of interactive devices using a range of logging techniques. In addition, we also present how we have improved the visualization techniques for enabling evaluators to effectively use the gathered data and focus on the aspects that allow them to identify parts of the application in which usability problems might hamper a user experience of high quality.

Future work will aim at enriching the analysis by integrating further aspects of environmental conditions, and also to improve the effectiveness of the representations provided to the evaluator. In addition, some work will be dedicated to the validation of the approach and tools, for instance on the usability diagnosis power, the assessment of evaluator support, etc.

References

1. Hartson, R.H., Castillo, J.C., Kelso, J.T., Neale, W.C.: Remote Evaluation: The Network as an Extension of the Usability Laboratory. CHI 1996, 228–235 (1996)
2. Ivory, M.Y., Hearst, M.A.: The state of the art in automating usability evaluation of user interfaces. ACM Computing Surveys 33(4), 470–516 (2001)
3. Tullis, T., Fleischman, S., McNulty, M., Cianchette, C., Bergel, M.: An Empirical Comparison of Lab and Remote Usability Testing of Web Sites. Usability Professionals Conference, Pennsylvania (2002)
4. Lister, M.: Streaming Format Software for Usability Testing. In: Proceedings ACM CHI 2003, Extended Abstracts, pp. 632–633 (2003)
5. Tennent, P., Chalmers, M.: Recording and Understanding Mobile People and Mobile Technology, E-social science (2005)
6. Andreasen, M., Nielsen, H., Schrøder, S., Stage, J.: What happened to remote usability testing?: An empirical study of three methods. In: CHI 2007, pp. 1405–1414 (2007)
7. Waterson, S., Landay, J.A., Matthews, T.: In the lab and Out in the wild: remote web usability usability Testing for Mobile Devices. In: CHI 2002, Minneapolis, Minnesota, USA, pp. 796–797 (April 2002)
8. Stoica, A., Fiotakis, G., Simarro Cabrera, J., Frutos, H.M., Avouris, N., Dimitriadis, Y.: Usability evaluation of handheld devices: A case study for a museum application. In: Bozanis, P., Houstis, E.N. (eds.) PCI 2005. LNCS, vol. 3746, Springer, Heidelberg (2005)

9. Avouris, N., Komis, V., Margaritis, M., Fiotakis, G.: An environment for studying collaborative learning activities. Journal (2004)
10. Serrano, M., Nigay, L., Demumieux, R., Descos, J., Losquin, P.: Multimodal interaction on mobile phones: Development and evaluation using ACICARE. In: Proceedings of the 8th Conference on Human-Computer Interaction with Mobile Devices and Services, Mobile HCI 2006, Helsinki, Finland, pp. 129–136 (September 12-15, 2006)
11. Waterson, S., Hong, J., Sohn, T., Heer, J., Matthews, T., Landay, J.: What Did They Do? Understanding Clickstreams with the WebQuilt Visualization System. In: Proceedings of the ACM International Working Conference on Advanced Visual Interfaces, Trento, Italy (September 12-15, 2002)
12. Maly, I., Slavik, P.: Towards Visual Analysis of Usability Test Logs Using Task Models (Paper in Conference Proceedings). In: Coninx, K., Luyten, K., Schneider, K.A. (eds.) TAMODIA 2006. LNCS, vol. 4385, pp. 24–38. Springer, Heidelberg (2007)
13. Paganelli, L., Paternò, F.: Tools for Remote Usability Evaluation of Web Applications through Browser Logs and Task Models, Behavior Research Methods, Instruments, and Computers. The Psychonomic Society Publications 35(3), 369–378 (2003)
14. Paternò, F.: Model-based design and evaluation of interactive applications. Springer, Heidelberg (1999)