Web Applications Usability Testing With Task Model Skeletons

Ivo Maly, Zdenek Mikovec,

Czech Technical University in Prague, Faculty of Electrical Engineering,
Karlovo namesti 13, 121 35 Prague, Czech Republic
{malyi1, xmikovec}@fel.cvut.cz

Abstract. Usability testing is technique for analysis of the usability problems of applications, but it requires significant effort to prepare the test and especially to analyze data collected during the test. New techniques such as usage of task models were introduced to improve and speed up the test analysis. Unfortunately, only few applications provide the task model. Therefore, we propose a method and tools for partial reconstruction of the task list and the task model called skeleton. This reconstruction is done from the usability expert's application walkthroughs. The task model skeleton is generated automatically, but it should provide similar information during the usability data analysis as manually created full-scaled task model. In order to evaluate usage of the task model skeleton we conducted a usability study with the web e-mail client Roundcube. Results show that the task model skeleton can be used as a good substitution for the manually created task model in usability testing when full-scaled task model is not available.

Keywords: Usability testing, Task list, Task model, Web applications

1 Introduction and Motivation

Incorporating usability testing into software application development process, as presented e.g. in [2], can significantly increase the efficiency of the development process and the acceptance of the final application by the users. The problem is that usability testing is not an easy and straightforward process. It consists of several steps, see steps rectangles in Fig 1, which are time consuming. In each step of the usability test wide range of supportive data and documents are created, e.g. task list, screener, various questionnaires and forms for annotations, test logs. Also, additional data can be recorded like audio/video recordings or status log of the tested application. This information is not always interconnected, e.g. task list and audio/video recordings collected during the execution of the usability test. These not properly interconnected data is hard to analyze in the Step 4 of the usability testing process. If the data is interconnected sufficiently (e.g., usage of task model to create relations between collected data seem to be a very promising approach), we can use more sophisticated analytical methods, e.g. statistical [6] or visual analysis [1], to find usability problems of the tested application.

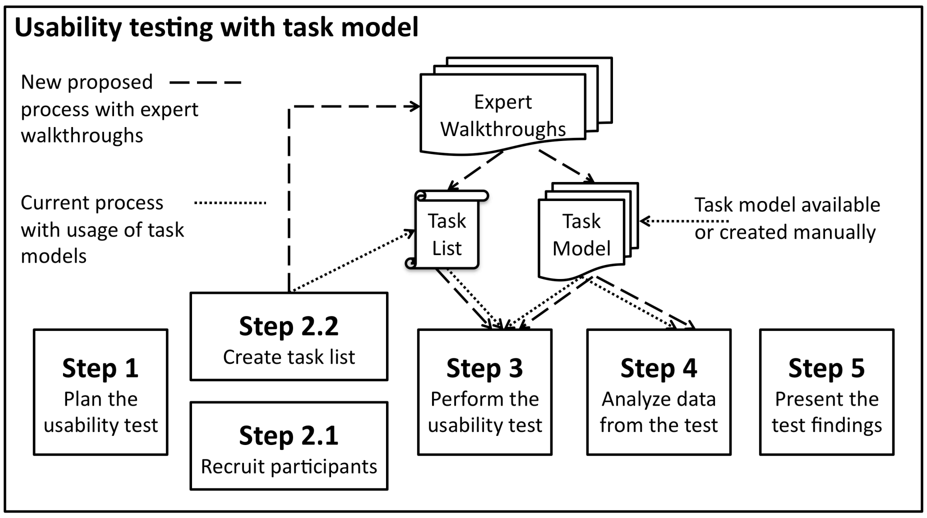
Usability testing with task model
New proposed process with expert walkthroughs
Current process with usage of task models
Expert Walkthroughs
Task List
Task Model
Task model available or created manually
Step 1
Plan the usability test
Step 2.2
Create task list
Step 2.1
Recruit participants
Step 3
Perform the usability test
Step 4
Analyze data from the test
Step 5
Present the test findings

Fig. 1. Schematic diagram of the usability testing process with the task model. In the lower part there are steps of the usability testing process and in the upper part there are the task list and the task model incorporated into the process of usability testing.

1.1 Usability Testing with Task Models

In order to interconnect the collected data the task model can be used. In Fig. 1 we can see that the task model is created prior the start of the usability test and it is used in Step 3 during the execution of the usability test and mainly in Step 4 during the analysis of the data collected from the usability test. In Step 3, we also use task list, which is the foundation of the usability test. Usually, the task list and the task model are created separately, even though there is relation between them. During the execution of the usability test the task model can be used for interconnection of observer's annotations with the task, as presented in [1]. It can be also used for post-test interconnection of log records generated during the remote usability test with the task from the task model, as presented in [6].

Main benefits of the task model usage can be found in the analysis (Step 4), where it is used as an interconnection between the task list and the collected data. For each user interaction collected we can judge whether it belongs to the currently solved task or not without any time consuming search in the video recording or in other data sources. Several usability test analysis tools, which take advantage of task models, were introduced in [1], [4] and [5]. They show collected data in the form of timelines visualizations in order to present the length of each task, length of the interaction and they give information whether the interaction was expected in the current task.

1.2 Issues of Usability Testing with Task Models

Problem of usage of task models in the usability tests is the necessity to have the task model at disposal during the test. There are 2 typical scenarios that may happen:

- **Task model is available.** This is typical for applications that were generated automatically or semi-automatically based on the task model. Therefore the task model was created during the design of the application.
- **Task model is not available**. If we want to use the task models, we need to create them. Creation of the task model may not be an easy process, due to the often complexity of the task or the tested application. Creation of the task model is also additional activity that must be performed before the test begins.

For the majority of applications the task model is not available at the time of the usability testing. Therefore, we were looking for the possibility of automatic or semi-automatic generation of the task model or similar data structure, which would have similar properties like the task model during the usability test execution and during the analysis of the data from the test (Step 3 and Step 4 in Fig. 1). Our solution, depicted by dashed lines in Fig 1, is based on the expert walkthroughs that are created during the task list creation step (Step 2.2 in Fig. 1). From these walkthroughs the *task list skeleton* and the *task model skeleton* are created instead of the task list and the task model. The task list skeleton is a template for the task list with the list of expected steps. Details of the task list skeleton are in chapter 2.3. The task model skeleton is a sequence of expected user interactions. Compared with the task model, the task model skeleton is much simpler, without task relations or without hierarchical structures. Details of the task model skeleton are in chapter 2.4.

2 Creation of Task List Skeleton and Task Model Skeleton for Web Applications

Presented approach of creation of the task list and task model skeletons is on one hand generic but on the other hand it may differ on each application platform (e.g. web, desktop). In this paper, we will focus on the tools for the web application testing and on creation of the task list and the task model skeletons for web applications.

2.1 Related work

Similar approach of automatic task model creation was presented by Paganelli et al. [8]. Authors parsed the HTML structure of each web page and created a Concur Task Tree (CTT) task model for the whole web application. Problem of the presented approach is that it generates quite big task models even for quite simple web applications, e.g. application with about 10 web pages is represented by CTT task model with 181 states. Such huge tree is good for computer processing but will be complicated for human expert. Another drawback is that it follows the HTML interactive objects only. Current rich web applications are using JavaScript to provide interaction with the user. Therefore the presented algorithm will not detect parts of the interactions and task model states based on the JavaScript. While our approach is based on recording of interactions executed by an expert walkthrough we are able to record all of them. Also the task model is containing much smaller number of states that make it easily understandable.

2.2 Interaction Log Recorder

Our approach of automatic task list and task model reconstruction is dependent on logging of user interaction (log record). Logging is used to detect and save both expert walkthroughs in Step 2.2 and user interactions during usability test (Step 3). Each log record must provide sufficient information about the type of interaction performed and about the new state into which the application had moved. In this work we were focused on implementation of a log recorder for web applications. Our recorder is based on Selenium IDE (http://seleniumhq.org/projects/ide/), which is a browser plug-in using JavaScript to listen to mouse clicks and key presses and records the data about interaction. Recorder is application independent so we do not have to install custom JavaScript code into the tested application in order to record interaction log. The structure of the log record is standard Selenium IDE XML log format [9]. While the standard Selenium IDE plug-in does not store time stamps, we have implemented custom function to include them into the log record.

2.3 Task List Skeleton

The task list skeleton is a template of a task list with the list of expected steps. It is a rich text document where expected steps generated by the expert walkthrough are automatically inserted into task list template using XML transformations. If there are more walkthroughs for the particular task, these walkthroughs are transformed into separated lists of the expected steps. In the Table 1, there is an example of the task definition from the usability test of the Roundcube e-mail client. Details about the usability test are in the chapter 3. The task description (in italics) must be entered by the usability expert who prepares the usability test. When the missing parts of the document are filled in the document becomes finalized task list and can be used in usual way by the usability test moderator and logger.

2.4 Task Model Skeleton

The task model skeleton is a sequence of expected user interactions. Visual representation of the task model skeleton automatically generated for the Task 5 (expert walkthrough 5.1) is in Fig. 2. There are 3 interactions that must be executed in order to complete the task. In Fig 3, the manually created task model for the Task 5 is presented, which is based on two expert walkthroughs (walkthrough 5.1 and walkthrough 5.2). Compared with the full-scaled task model, the task model skeleton is much simpler, without task relations or without hierarchical structures. Each subtask of the task model skeleton reflects one step in the task list.

Table 1. Example of the task list skeleton for the task 5 from the Roundcube e-mail client usability test.

	Task list skeleton
Task description	*Open first new message in inbox and read it. Add sender of the message to the address book.*
Expected steps	1. Click at the link "Info bulletin". 2. Double click at the link " Info bulletin". 3. Click at the image "add". -- or -- 1. Click at the link "Info bulletin". 2. Double click at the link " Info bulletin". 3. Click at the "rcmbtn101". (Address book) 4. Click at the "rcmbtn105". (New contact) 5. Type "Info" into field " rcmfd_name ". 6. Type "info@lkom.cz" into field " rcmfd_ email". 7. Click at the "rcmbtn100".

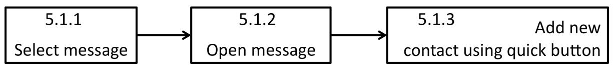

Fig. 2. Visual representation of task model skeleton 5.1 for task 5.

Each subtask of the task model skeleton contains the same information about the user interaction as was recorded by the interaction log recorder. Thanks to that we can perform comparison of the subtask in task model skeleton with particular interaction in user log record in order to match the interaction with the subtask. Task model skeletons are stored in XML file format in order to be easily loaded by the analytical applications used in analysis step (Step 4 in Fig 1).

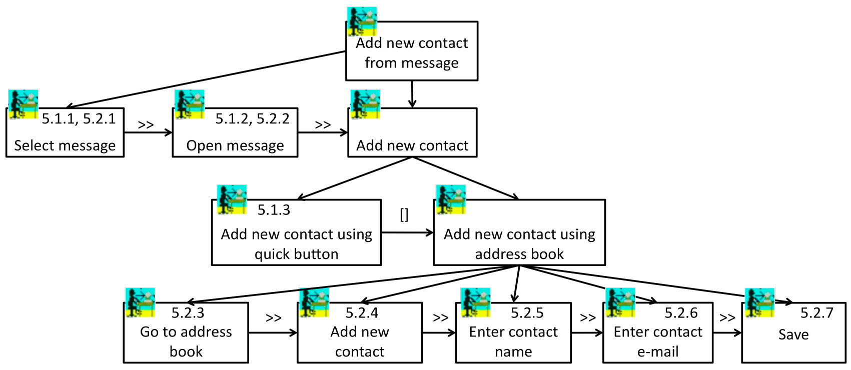

Fig. 3. Task model for task 5 representing possible task model created from the task model skeletons 5.1 (in Fig 2) and skeleton 5.2

4 Use Case: Web E-mail Client Roundcube

In order to evaluate our proposal of the web application usability testing with the task model skeletons, we executed the usability test of the web e-mail client Roundcube (http://roundcube.net). At the beginning, we have analyzed the application and created the expert walkthroughs using the interaction log recorder (see chapter 2.2). Task list and the task model skeleton were created using XML transformations and the task descriptions in the task list were added. During the test we have recorded the user interaction using the interaction log recorder and then we have analyzed the data using Interactive Visualization Environment (IVE) tool [7]. IVE is an interactive tool for visual analysis of data from usability studies. IVE uses an internal object database and convertor plug-ins to convert collected usability test data into the internal database. Usability expert uses set of interactive visualization views to analyze the data. Each interactive visualization view is developed as a plug-in that has access to the IVE internal database and it can communicate with other plug-ins through a simple message dispatching system

3.1 Test Setup, Task Model and Task List Skeleton Creation

According to a minimal required number of participants for usability study [3] we selected 6 participants, both technically experienced and non-technical ones. None of them had a previous experience with the Roundcube client, but some of them had experience with other web e-mail clients, such as Gmail. The test was conducted in the usability lab and including pre- and post-test questionnaires it took from 30 to 40 minutes. Beside the data collected by the interaction log recorder we have also recorded an audio and video from each session.

During the application analysis we have selected 11 tasks and prepared the task model skeletons and the task list skeleton for them. Tasks were focused on typical e-mail activities like reading, sending and forwarding e-mails. We have also focused on e-mail folder management and work with address book. Each task in task list and task model skeleton was created using one expert walkthrough except the task 5, where we have recorded two walkthroughs. Creation of the task model skeleton and the task list skeleton generated only a small time overhead over the standalone task list generation, because we needed to save recorded interactions for every task, which has slightly broken the flow of recording process. Also recording of the alternative walkthrough for task 5 required rollback of the system to the previous state that took few minutes. Overall process of the task model skeleton generation and the task list skeleton generation was perceived positively.

3.2 Analysis of Usability Test Results with help of Task Model Skeleton

In the Fig. 4, there are two timelines representing data collected during the usability study of the Roundcube e-mail client of the participant p7. The lower timeline shows length of each task. The upper timeline shows recorded log interaction combined with task model skeleton. Each rectangle represents one user interaction. The color of the rectangle highlight, whether the interaction was correct (green), incorrect (white) or it may be correct, because the interaction is expected in some moment in the task (yellow). The interpretation of the sequence of various colors is:

- **Sequence of green rectangles.** Such a sequence represents optimal execution of the task (e.g. Task 6). There may be white rectangles before the first green rectangle (e.g. Task 2), which represent unexpected interaction before the optimal execution started.
- **Sequence of green rectangles, which is interrupted by white rectangles and continues with green rectangles.** Such a sequence represents event, when participant performs optimal execution of the task, gets lost, but he/she is able to recover back.
- **Sequence of green rectangles, which turns into sequence of yellow rectangles.** Such s sequence represents same start as in previous example, but the participant recovers with interaction from the task but not the expected one (e.g. user starts from beginning).
- **Sequence of yellow rectangles.** Such a sequence (e.g. Task 3) represents situation when the participant starts the execution of the task with interaction that is part of the particular task model skeleton but is not the expected one. Example of such case is usage of browser navigation buttons or optional execution of first (or expected) interaction.

The interpretation of the timeline visualization allowed us to focus on the most problematic tasks, e.g. in task 3 we have found that the participant performed search action from different web page and in the task 5 the participant added wrong contact. In the task 2, the participant was not sure if he/she finished the task, so he/she performed last interaction again.

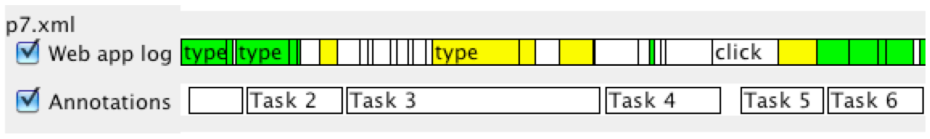

Fig. 4. Visualization of the timeline for participants 7.

4 Conclusion

In this paper we have discussed the advantages of the usability testing with the task models. We pointed out that the biggest barrier of this method is the missing task model for the most of the tested applications. To overcome this barrier we have suggested the modification of the task list creation step be extended to create also the task model. The result is the automatically created task model skeleton and the task list skeleton. These are then used for task model based usability testing and analysis.

We have conducted the usability study of the web e-mail client Roundcube with 6 participants in order to evaluate our concept of the task model skeleton. The analysis of the data showed that we were able to match user's interactions with task model skeleton and to show which interaction belonged to which task. Then we showed the interpretation of data visualization and examples of usability problems that were found.

However, we found out that in the web environment the browser controls can influence the list of recorded interactions, e.g. the usage of the back navigation button. Therefore, the future work in this area should be addition of web browser interactions recording into the interaction log recorder. With this knowledge, we can introduce refinements to the user log record and better visualize user interactions as correct or incorrect. Sometimes it was hard to recognize, what was the object the participant interacted with. The example is click at "rcmbtn101" which means that the user clicked at the address book. As a future work, we should reflect this and design better identification of targets both during the interaction recording and data analysis.

Acknowledgments.

This work has been partly supported by the Ministry of Education, Youth and Sports of the Czech Republic under the research program LC-06008 (Center for Computer Graphics) and by the Sucess project – Sun Center of Excellence for Accessibility & Usability (http://amun.felk.cvut.cz/coe/).

References

1. Maly, I., Slavik, P.: Towards Visual Analysis of Usability Test Logs. In: Coninx, K., Luyten, K., Schneider, K.A. (eds.) TAMODIA 2006. LNCS, vol. 4385, pp. 25--32, Springer, Heidelberg (2007)
2. Rubin, J.: Handbook of Usability Testing, John Wiley and Sons, New York (1994)
3. Jakob Nielsen's Alertbox, March 19, 2000: Why You Only Need to Test With 5 Users. http://www.useit.com/alertbox/20000319.html
4. Paterno, F., Russino, A., Santoro, C.: Remote evaluation of Mobile Applications. In: Winckler, M., Johnson, H., Palanque, P. (eds.) TAMODIA 2007. LNCS, vol. 4849, pp. 155--168. Springer, Heidelberg (2007)
5. Wurdel, M., Propp, S., Forbrig, P.: HCI-task models and smart environments. In: Forbrig, P., Paterno, F., Pejtersen A.M. (eds) 1st TC 13 Human-Computer Interaction Symposium (HCIS 2008), volume 272 of IFIP International Federation for Information Processing, pp 21--32, Springer, Boston (2008)
6. Paganelli, L., Paterno, F.: Intelligent Analysis of User Interactions with Web Applications. In: Proceedings of ACM IUI'02, pp.111--118, ACM Press, New York (2002)
7. Maly, I., Mikovec, Z., Vystrcil, J.: Interactive Analytical Tool for Usability Analysis of Mobile Indoor Navigation Application. In: 3rd International Conference on Human System Interaction (HSI 2010), pp. 259--266, IEEE, Warsaw (2010)
8. Paganelli, L., Paterno, F.: Intelligent Analysis of User Interactions with Web Applications. In: International Journal of Software Engineering and Knowledge Engineering, vol. 13 (2), pp. 169--189, World Scientific Publishing (2003)
9. Selenium IDE documentation. Accessed on April 8, 2010. http://seleniumhq.org/docs/