

[M22]

Experiences Using Usability Techniques in Development of Active Spaces RBAC GUI

Yong Liu, Geetanjali Sampemane,

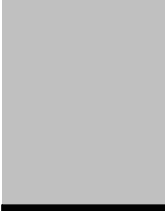
Kay Connelly

Abstract

This paper presents a case study in developing an administration GUI using Cognitive Walkthrough and usability testing. We introduce the evaluation procedure of the GUI. We then compare the predictions made by Cognitive Walkthrough with the usability problems found in empirical usability testing. The strength and weakness of the two methodologies and their respective task scenarios are discussed. Based on the analysis, we arrive at conclusions which could be used to improve the effectiveness of the two evaluation methods and point out the problems still open.

Introduction

Usability is an important issue in today's software design and development. To achieve good software usability, we can perform a usage-centered design during the development (Constantine and Lockwood, 1999). Different from the conventional user-centered design, the process of usage-centered design is more systematic and specified. It relies less on user feedback and testing, but more on modeling. Although the importance of user testing is much weakened in usage-centered design, usability evaluation is still a helpful means to achieve an effective and



efficient design. Usability inspection can help to identify usability deficiencies in early design stages, and so does user testing for working prototypes or systems. In this paper, we report on our experience in evaluating a system administration GUI using Cognitive Walkthrough (Wharton, Rieman, Lewis and Polson, 1994) and think-aloud usability testing (Rubin, 1994). We compare the two methods based on their evaluation results and discuss some problems which still remain open. We also discuss which problems could be addressed by carefully conducting usage-centered design.

Usability Study Situation

An active Space (AS) is a multi-user ubiquitous computing environment. In current Active Spaces system, one form of Role-Based Access Control (RBAC) is being used (Sampemane, Naldurg and Campbell, 2002). To facilitate the management of the AS RBAC system, an administration GUI is developed to help administrators assign and revoke permissions of specific users and accomplish other management tasks.

We use two usability evaluation methods on the administration GUI, which are Cognitive Walkthrough (CW) and usability testing. CW is a method for predicting usability problems in an interactive system. This method mainly focuses on evaluating a design for ease of learning, particularly by exploration. Usability testing is a method in which representative users do some predefined tasks with the product. Typically usability testing involves two types of measurable usability parameters, which are performance measures (how capable the users are using the product) and preference measures (how much the users like the product).

Cognitive walkthrough.

In our CW on the administration GUI, we assume the intended users of the product are administrators of Active Spaces. The users will use the GUI to perform daily role and permission management. Three analysts participate in this usability evaluation. They are all computer specialists. In this CW, analysts are given a prototype of the GUI with full features. They complete the analysis by exploring this prototype and searching for any usability problems. A total of 19 task scenarios are constructed covering all basic functions of the GUI. Among them, 5 are actually inspected in the analysis phase. For each scenario, evaluators check the goal and actions taken to achieve the goal, and try to answer CW questions. By answering these

questions, some usability problems and potential usability deficiencies are identified. There is no time limit for the whole CW.

Usability testing.

In the usability testing, a total of four participants are tested in two weeks. All participants are computer specialists. Three participants have been involved in the development of the Active Spaces system. One of them is the key developer of the AS RBAC system. The fourth participant is a novice to AS RBAC. Test materials include a background questionnaire, two terminology questionnaires (before and after performance test) and a post-test questionnaire. The usability testing consists of the main performance test designed to gather extensive usability data via direct observation, as well as several paper-and-pencil tests designed to gather information about the interface terminology.

Comparison between Cognitive Walkthrough and Usability Testing

Comparison between Results from Two Evaluations.

The CW and the empirical usability testing are both based on the administration GUI prototype. Although system features covered by the task scenarios of the two evaluations are identical, the scenarios used in usability testing are more sophisticated. In CW, we skip some tasks due to time limit and similarity with other tasks. This does not affect the coverage of task scenarios because we can extend our evaluation results to those skipped scenarios. In usability testing, we use three sophisticated task scenarios. These scenarios are designed to simulate the actual cases that will happen in a real working environment. They cover the same set of features as CW scenarios do, but testees are responsible for composing these features to complete the task. Table 1 shows the statistics of usability problems identified using the two methods.

Table 1 – Usability Deficiencies Identified in Two Evaluations

	CW	Usability Testing	Source
Problems found through both evaluations	7	3	Insufficient feedback Confusing labels Functional deficiency
Problems found through CW	7	0	Functional deficiency


Problems found through usability testing	0	11	Insufficient help/hint Functional deficiency
Total	14	14	

Analysis of the Different Results from Two Evaluations.

In our case study, CW finds the same number of usability problems as usability testing does. However, only based on the number of deficiencies found, we cannot tell if one method is better than the other. One main reason that has led to the various results of the two evaluations is their different task scenario design.

In CW, each scenario corresponds to one basic function. For example, administrator can add new applications into the system using the GUI, so there is a corresponding scenario "add a new application to AS system" in CW to test this function. Constructing scenarios in this way has three advantages. Firstly, such scenarios can easily cover all basic functions of the product. Secondly, it is convenient to decompose such scenarios into actions which analysts can follow. Thirdly, such scenarios can help identify more detailed usability problems which are easier for developers to follow and make improvement. However, there are also disadvantages. Simple scenarios cannot reflect the relations between different operations. For example, when a new student comes to AS system, the administrator should add her into system and map her to the student role. Simple scenarios cannot help find out potential usability problems which hinder administrators from accomplishing the operation sequence correctly. Another deficiency of these types of scenarios is that they often lead to trivial and redundant actions such as "click OK button" or "move mouse to file menu". Analysts may become frustrated with these tedious actions and likely skip them as we did in our CW.

In usability testing we use more complicated scenarios. These scenarios are designed to simulate empirical working environments and can identify more subtle usability deficiencies (e.g., deficiencies hidden between successive operations). The main disadvantage of such scenarios is that their difficulty is hard to control. If a scenario is too elaborate, users will easily fail at some points and the test may be interrupted occasionally. On the other hand, if we provide too many hints in a task scenario description, the effectiveness of this scenario will be hurt. Another



problem of this type of scenarios is that it is difficult to cover all of the features of a product with a limited number of scenarios.

In CW, all evaluations are based on the scenarios inspected during the analysis. Evaluation results are totally tied to scenarios. This brings a risk – if these tasks are not designed properly, the effectiveness of the evaluation will be greatly weakened. Even if these scenarios have covered all basic functions of the product, some useful information, such as user preference, may be missed. In usability testing, questionnaires and debriefing sessions can help evaluators gather more usability information, which greatly decreases the risk.

Problems Remaining Open

Tedious CW process.

In our CW, task scenarios cover all basic functions. Since some product functions have the same action sequences, we only evaluate a typical function and skipped others. But even by doing this, much effort must be made to evaluate similar features. We wonder whether there is any means to reduce duplicate walkthrough and tedious story recording in a CW.

Scenario design.

Well designed scenarios are the crucial factor in usability evaluation. Although the advantages of essence use cases used in usage-centered design have been widely recognized, this task modeling method only works well for interface design and requirements modeling (Constantine and Lockwood, 2001). Usability inspections and usability testing usually require more detailed and specific scenarios that will exercise a variety of functions. As mentioned above, the two types of scenarios used in our case study have their own pros and cons respectively. Now some questions are still open. For example, in usability testing, should the description of a task scenario be rough or detailed? How to estimate the information provided to users by these descriptions?

Countermeasure to failures in usability testing.

Usually, the usability test plan does not include detailed countermeasures to user failures which may happen during tests, such as overtime or doing wrong actions. But such failures may occur anytime. Test supervisors need to premeditate countermeasures to every possible failure in a scenario.



Now the question is what the advantages and disadvantages of a specific countermeasure are?

Remote Testing.

Time and other resource issues are often obstacles when conducting usability testing. This can be addressed by developing dedicated remote testing tools. But such tools can be very costly. We believe it is possible to use some off-the-shelf tools to conduct a remote usability testing. For example, MSN messenger is such a competent tool. We can use it to instruct users during tests, watch users' behavior, or even inspect users' screen. We believe that because of their low cost and convenience, such tools will be extensively used in future usability evaluation practice.

Conclusion

Although usage-centered design is more like engineering than a trial-and-error process, usability evaluation is still indispensable in the design process – nobody can assure a design is usability problem free before doing some serious evaluations. CW and usability testing are both scenario-based evaluation methodologies. This case study communicates a clear message to both system designers and developers of this type of evaluation techniques – task scenario is crucial to both cognitive walkthrough and usability testing. Simple scenarios cannot reflect relations between successive operations and often lead to trivial and redundant actions. Sophisticated scenarios can help identify usability problems between successive operations, but the difficulty of decomposition is hard to control. Other testing materials, such as questionnaires, are often good supplements to scenarios.

Most of the usability problems identified in this case study can be avoided through proper usage-centered design. For example, insufficient help/hint can be avoided by carefully analyzing user roles in the system (role modeling); in task modeling phase, lack of feedback can be identified by carefully setting up essential use cases and navigation map. But what is worth mentioning is, even usage-centered design cannot guarantee that these problems will never appear. It only provides a capability to get rid of these usability deficiencies to some extent.



References

- Constantine, L. L., and Lockwood, L. A. D. (1999) *Software for Use: A Practical Guide to the Models and Methods of Usage Centered Design*. Reading, MA: Addison-Wesley.
- Constantine, L. L., and Lockwood, L. A. D. (2001) Structure and style in use cases for user interfaces. In M. van Harmelan, Ed., *Object Modeling and User Interface Design*. Boston: Addison Wesley.
- Rubin, J. (1994) *Handbook of Usability Testing: How to Plan, Design and Conduct Effective Tests*. John Wiley and Sons, New York. 1994.
- Sampemane, G., Naldurg, P., and Campbell, R. (2002) Access control for Active Spaces. In *Annual Computer Security Applications Conference (ACSAC2002)*, Las Vegas, Nevada, Dec 9-13 2002.
- Wharton, C., Rieman, J., Lewis, C., and Polson, P. (1994) The Cognitive Walkthrough Method: A practitioner's guide. In J. Nielsen and R. L. Mack, Ed., *Usability Inspection Methods*. New York: John Wiley. 1994.