

Towards a Pattern Based Usability Inspection Method for Industrial Practitioners

Martin Schmettow

Fraunhofer IESE, Fraunhofer-Platz 1, D-67663 Kaiserslautern
martin.schmettow@iese.fraunhofer.de

Abstract. Usability Inspections are essential for early detection of defects in UI design, but they require sound usability knowledge. Usability Patterns are the state-of-the-art format for describing usability knowledge. Thus, it seems obvious to use them as a means for evaluating the design of user interfaces. In this paper a usability inspection method based on usability patterns is introduced, which may be especially well suited for practitioners in software development, who are in general not usability experts.

1 Introduction on Usability Inspections

Inspections in Software Engineering are techniques to find defects in software products. As they can – opposed to testing – be performed on static artifacts (meaning not runnable modules), they are excellent in finding defects very early, which knowingly saves time and money [1].

Usability inspections in general share these goals and are in the underlying principles and processes quite in line with software inspections. But they seem to not be as established in software development, which regularly leads to late and costly defect elimination or, in the worst case, unusable products.

The main problem in current software development practice is, that construction as well as evaluation of user interfaces (UI) are most often done by developers, who are in general not dedicated usability experts [2]. Thus usability evaluation methods are needed, which are well suited for non usability experts. To be well suited for non usability experts a method must at least provide comprehensible usability guidelines (or usability knowledge in general) and must support the evaluator to take the perspective of the common user. Of additional benefit is, if a method facilitates explicit recommendations for design improvement instead of mere defect identification.

Indeed, the known methods provide usability knowledge via differing forms of ergonomic guidelines, like heuristics [3], psychological theories [4] or interface standards. But these guidelines have some shortcomings which make them cumbersome or simply unusable for software developers. And, in fact, several studies show, that less skilled evaluators are less efficient in identifying usability defects and that there are considerable differences in efficiency even for high skilled evaluators [3, 5].

In the following we propose the construction of a usability inspection method, which uses usability patterns as a source of usability knowledge. We are optimistic, that this

method would be well suited for software developers, because it makes use of detailed and problem-oriented usability knowledge and has a procedure to strongly encourage taking the perspective of the user.

2 Usability Patterns as a Source for Usability Inspections

Usability patterns are on the way to become the most prominent format to collect, describe and structure usability knowledge [6]. They describe well established solutions together with preconditions and rationales for ergonomic design of user interfaces. Usually they are collected and classified in pattern collections or languages for general application domains or more specific purposes [7-9].

Having a further look into the nature of usability patterns, several advantages and also a few problems arise for basing a usability inspection method on patterns. Compared to usability heuristics and interface standards, patterns have several advantages:

- Patterns state preconditions and rationales for the proposed solution, which *helps to identify the correct pattern for a specific situation*, while the decision of when to apply a specific heuristic is left to the evaluator.
- The descriptions of problems and solutions are far more verbose and problem-oriented than with heuristics, which makes them *easier to understand and correctly apply* for evaluators.
- Opposed to heuristics patterns describe concrete solutions, which makes them a *natural means for design recommendations* in an evaluation.

Additionally, the idea of describing best practices as patterns is well established in other fields of software engineering for years [8, 10, 11] and thus is widely known and accepted by software developers.

On the other hand there are several problems in using patterns as a source for usability inspection:

- Since pattern collections use to be very large (but still not as large as design standards), a powerful information retrieval mechanism has to be provided to find applicable patterns for each step of an inspection.
- Established pattern collections use to refer to quite general application domains (WIMP interfaces, Web sites), which might be applicable but not sufficient for applications in specific domains. However, there already exist some collections for special purposes [12, 13].
- The fact that a pattern proposes a good solution for a specific problem does not deny, that the designer might have found an innovative or at least alternative good solution for the problem. However, using a standardized solution for similar design problems leads to a high consistency inside and across applications, which is desirable for e.g. better learnability.
- The completeness of defect identification is highly dependent on the existence of applicable patterns. But this is an inherent problem of any guideline or the mere expertise of an evaluator, which might always be limited, inadequate or outdated. With the pattern-based method this problem will degrade

with the appearance of more complete and also domain-specific pattern collections.

These problems have to be solved by the construction of the inspection method or at least have to be kept in mind when using it. In the following a proposal for the construction of a usability pattern based inspection method (UPI) is outlined.

3 Description of the UPI method

Process of the method

The overall process of the UPI method is very similar to other inspection methods in UE and SE: In the preparation phase evaluators will be selected and advised and the artifacts and evaluation environment is prepared. Especially the compilation of an adequate pattern collection is crucial here. For many applications the general pattern collections available [14, 15] should be sufficient, but these may be augmented with domain specific patterns for special purposes (examples of how to build domain-specific pattern collections can be found in [12, 16]).

Then several single evaluation sessions are performed before a final review meeting is held which consolidates the results and prepares them for presentation. The innovative core of the method is the application of usability patterns as evaluation criteria in the single evaluation sessions – therefore this will be the focus of the further description.

Principles of the method

In a first overview the method can be described in seven principles which form the basis of the single evaluation procedure:

1. *Guidance* for the inspection is provided by detailed descriptions of user tasks, which are to be performed by the evaluator.
2. The evaluator is advised to always be aware of what *user activity* she is performing, what is called *self monitoring* further on. The user activity is selected from a predefined set.
3. *Checkpoints* for the inspection are defined by changing dialogues and shifting user activity.
4. *Matter of inspection* are dialogues, screens and single screen elements.
5. *Criterion for evaluation* is a match or mismatch of a dialogue or screen element to a specific pattern.
6. *Selection* of applicable patterns is done by the current *user activity* and – of course – the patterns described preconditions.
7. *Recording* is done immediately for every single step of the evaluation for positive as well as negative results.

Single Evaluation Procedure

The procedure of the method as well as the recording format is quite well-defined. Since the inspection path is guided by a detailed task description, the first step for the evaluator is to start doing a task or a subtask. While he is doing the task he is held to monitor his own flow of activities. These activities stem from a predefined set of 15 general user activities (some examples are shown in Figure 1) and every pattern in the used pattern collection is classified to one or more of them.

This *self monitoring* of activities is one central idea in the method and acts in a three-fold way:

1. It is the *pacemaker* in the procedure: Whenever the current user activity changes, the current dialogue is compared to usability patterns.
2. It is the main *information retrieval mechanism* for preselecting patterns for a given situation.
3. It enables the evaluator to become better aware of the *user perspective*. This works, because the self monitoring hinders the evaluator to use the application in an intuitive or highly automated manner - especially when he is an application or domain expert. For example, when a new dialogue appears, the evaluator could ask: Would the user immediately comprehend the dialogue and start entering data or would he first orient himself or search for help?

The procedure of selecting and applying patterns to each dialogues, as outlined in Figure 1, is as follows: When the activity changes and/or a new dialogue appears, the evaluator will select all patterns, which are classified to this activity, and check their applicability in two steps: First he compares the current situation to the preconditions (problem, context and forces) of each pattern. Then he tries to imagine roughly, if and how each pattern could be instantiated into the current dialogue. If both checks - preconditions and imagination - pass, the pattern is considered as applicable. The set of applicable patterns is then compared to the current dialogue to decide, if they match fully, partly or not at all. In the latter two cases a mismatch is recorded and a recommendation according to the pattern's solution is given.

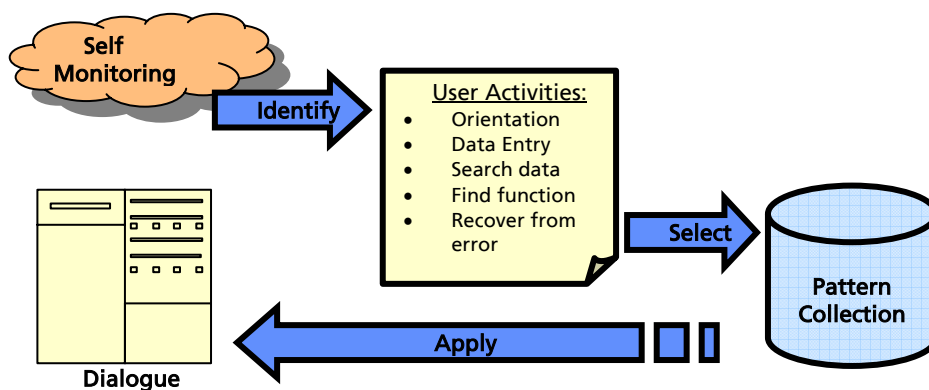


Figure 1 Procedure of selecting and applying patterns to the dialogue in question

4 Evaluation

The method was successfully applied in one industrial and two governmental projects as well as in several minor examples, where it in fact revealed a significant number of usability problems. In one of the projects, it was also shown, that the method is quite applicable to early design artifacts (static UI prototypes). In all three projects the results were presented to the developers and project sponsors and were in general easily understood and accepted by those.

The retrieval of applicable patterns through the selection by user activities works quite well. For example the collection of patterns for windows-like GUIs has about 75 entries. Far most user activities reduce the number of preselected applicable patterns to below ten results (which is below the number of heuristics proposed by Nielsen [3]).

At the moment a comprehensive validation study is being prepared, which will provide statistical measures for reliability, validity and efficiency compared to either heuristic evaluation or usability testing. Since the participants will mostly have few experiences with usability only, this experiment will also stress our hypothesis, that the Usability Pattern Inspection method is outstanding well-suited for non-experts.

5 Conclusion

In this paper a usability inspection method is presented which makes use of usability patterns as a source of usability knowledge. It is believed that the method is well suited for software developers, who are in general not experts in usability evaluation. This is mainly due to two facts: (1) patterns describe usability knowledge in detailed and well structured way, stating preconditions and rationales for concrete and well established solutions (2) the principle of self monitoring forces the evaluator to take the perspective of the common user. The main problem in using patterns is the mere size of pattern collections. This was met with the procedure of selection by user activity, which basically results in a context-adaptive checklist.

When the method is empirically proven to be reliable, valid and efficient, in the future we will focus on the following improvements:

1. The collection of patterns will be enhanced for special application domains (e.g. business process modeling tools [16]).
2. Tool support will be enhanced for session recording and report generation.
3. Comprehensive training material for industrial practitioners will be developed.
4. When the detailed characteristics of the method are explored, it should be integrated into a well defined UE/SE process framework.

6 References

- [1] B. Boehm and V. R. Basili, Software Defect Reduction Top 10 List. In: *Foundations of Empirical Software Engineering*, B. Boehm, H. D. Rom-

- bach, and M. V. Zelkowitz, Eds. Heidelberg, Germany: Springer, 2005, pp. 426-431.
- [2] J. McKirdy, An Empirical Study of the Relationships Between User Interface Development Tools. University of Glasgow, Department of Computing Science, Technical Report TR-1998-06, 1998.
http://www.cs.unb.ca/profs/jlumsden/publications/tech_report_tr-1998-06.pdf
 - [3] J. Nielsen, Heuristic Evaluation. In: *Usability Inspection Methods*, J. Nielsen and R. L. Mack, Eds.: John Wiley & Sons, 1994, pp. 25-61.
 - [4] C. Wharton, J. Rieman, C. Lewis, and P. Polson, The Cognitive Walk-through Method: A Practitioner's Guide. In: *Usability Inspection Methods*, J. Nielsen and R. L. Mack, Eds.: John Wiley & Sons, 1994, pp. 105-139.
 - [5] H. W. Desurvire, Faster, Cheaper!! Are Usability Methods as Effective as Empirical Testing? In: *Usability Inspection Methods*, J. Nielsen and R. L. Mack, Eds.: John Wiley & Sons, 1994, pp. 105-140.
 - [6] J. Borchers, *A Pattern Approach to Interaction Design*. Chichester, England: John Wiley & Sons, Ltd, 2001.
 - [7] M. Fowler, *Analysis Patterns : Reusable Object Models*. Boston: Addison Wesley, 1996.
 - [8] M. Fowler, *Patterns of Enterprise Application Architecture*. Boston: Addison-Wesley, 2002.
 - [9] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design Patterns - Elements of Reusable Object-Oriented Software*. Boston: Addison Wesley, 1994.
 - [10] D. M. Dikel, D. Kane, and J. R. Wilson, *Software Architecture: Organizational Principles and Patterns*: Prentice Hall PTR, 2000.
 - [11] L. Hage and K. Lappe, Patterns for the RE Process. In: *12th IEEE International Requirements Engineering Conference*. Kyoto, Japan: IEEE, 2004, pp. 90 - 99.
 - [12] M. Leacock, E. Malone, and C. Wheeler, Implementing a Pattern Library in the Real World: A Yahoo! Case Study. Presented at Information Architecture Summit, 2005.
 - [13] R. Gibbert, Eine Design Pattern-Sprache für mobile Applikationen mit dem Schwerpunkt Navigationssysteme. diploma thesis 2003.
 - [14] M. v. Welie, Patterns in Interaction Design.
<http://www.welie.com/patterns/gui/index.html> (last accessed: 2005)
 - [15] J. Tidwell, COMMON GROUND: A Pattern Language for Human-Computer Interface Design.
http://www.mit.edu/~jtidwell/interaction_patterns.html (last accessed: 2005)
 - [16] K. Kohler, D. Kerkow, S. Hess, and K. Schmid, Best Practices und Usability Pattern für Geschäftsprozess-Modellierungswerkzeuge. Fraunhofer IESE, Kaiserslautern, Germany, IESE.Report, 060.05/E, July 2005.
http://www.iese.fhg.de/Publications/Iese_reports/