Building and Evaluating a Pattern Collection for the Domain of Workflow Modeling Tools

Kirstin Kohler and Daniel Kerkow

Fraunhofer IESE
Fraunhofer-Platz 1, 67663 Kaiserslautern
{kirstin.kohler, daniel.kerkow} @iese.fraunhofer.de

Abstract. In this paper, we present the results of a case study conducted together with a small company that develops a workflow modeling tool. During the case study, we created a pattern collection for the domain of workflow modeling tools and evaluated a subset of these patterns. Beside the pattern description itself, the contribution of our work is a systematic process for identifying patterns. The results of the case study showed, that the identified pattern are a valuable instrument for software developers to improve the usability of their software in the given domain. Additionally this finding shows that the process of pattern identification is valuable as well.

Author Keywords: User interface pattern, case study, design methodologies

ACM Classification Keywords: H5.2. Information interfaces and presentations: Miscellaneous theory and methods. D.2.2 Software Engineering: Design Tools and Techniques|

1 Introduction

In the RL-KMU Project, we were in search of methods that support small and medium-sized companies in improving their usability competence. These companies usually do not have their own usability department; also, they cannot effort expensive usability training or consultancy [1]. Software engineers in these companies usually have to do user interface design and coding, but never learned how to systematically do this as part of their education. All these constraints made us investigate the use of user interface patterns in more detail.
User interface patterns are a promising approach to transfer knowledge about user interface design to user interface designers [2] [3] [4]. Due to the fact that patterns are a commonly accepted approach in the area of software engineering [5], they seem especially well suited to train/support software engineers [6]. In addition, several libraries are publicly available without extra charge and provide access to a large set of patterns [7] [8]. While validating the suitability of these libraries for the small enterprise in our project, it turned out that the libraries were not specific enough for the software applications developed in the company. The most popular libraries offer patterns for unspecific software systems. For specific domains, these patterns are

often not sufficiently tailored. Some recently developed libraries have addressed more specific domains, like web systems [9], e-business applications [10], or museum websites [11]. But none of the available pattern libraries addresses the design problems of our company, which develops a graphical workflow modeling tool.

In this paper, we present the results of a case study during which we developed a pattern collection for the domain of workflow modeling tools.
The contribution of our work is twofold:

- It includes the pattern description of 40 patterns identified in the domain of workflow modeling tools. We evaluated these patterns to ensure their validity.
- We developed a process to systematically derive patterns by abstracting them from best solutions found in software applications of the same domain.

In section two, we will elaborate the process we applied to derive the patterns. We will show one of the extracted patterns as an example.
Section three presents the results of the pattern evaluation, which gives evidence that the presented process to create patterns is valuable as well.

2 Derive Patterns from Best Solutions

In order to identify the workflow-specific patterns, we followed the steps shown in Figure 1. First, we created a usage model for novice users and phrased functional and nonfunctional requirements for workflow modeling tools based upon this usage model. In the next steps we evaluated strengths and weaknesses in the usability of four tools (cp. Table 2) and identified the best solutions among the four tools for each functional and nonfunctional requirement. In step 4, we described these solutions in a pattern notation in order to get a collection of 40 workflow-specific patterns.
In the following subsections, we will elaborate each of the 4 steps in more detail.

Step 1: Collecting User Tasks and Nonfunctional Requirements

The first step to gain the pattern was the creation of a usage model derived from the requirements of the workflow manamgent tools. This usage model assumes the following imaginary inexperienced user, e.g., an employee of a small or medium-sized enterprise whose goal is to improve the effectiveness of a specific process within the company.

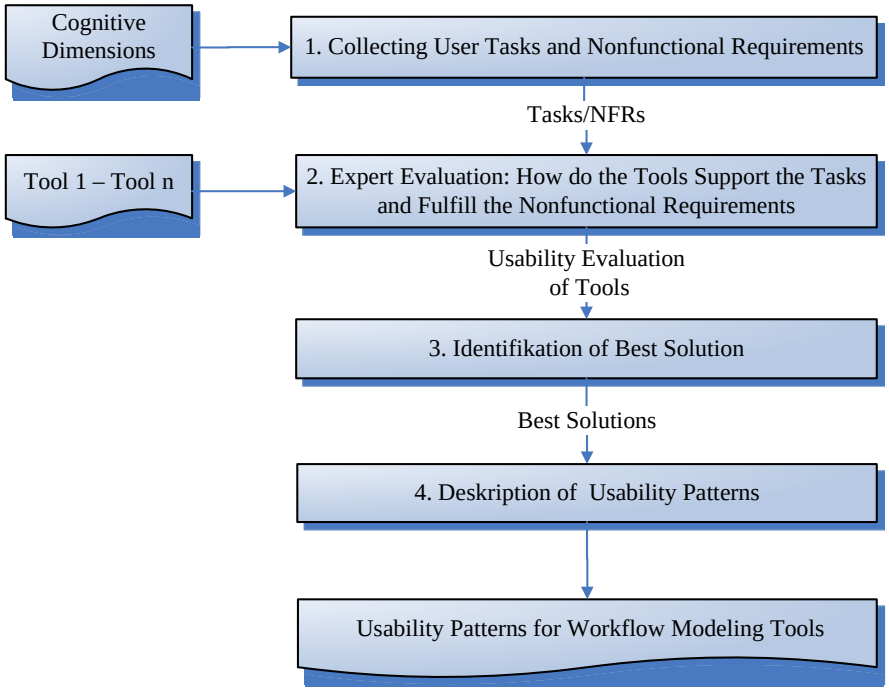

Figure 1: Process for identifying patterns

In order to improve the process, s/he has to implement the process into a workflow tool. The employee has not modeled business processes before and is familiar with standard PC applications, but not with workflow modeling tools.

The main reason for the choice of this actor is the productivity related goal of enabling non-experts to customize workflow tools and thus the higher degree of user support needed. The usage model represented by a use-case diagram is shown in Figure 2.

The employee, after having installed the workflow modeling tool, first of all wants to create or refresh his/her implicit mental model of the business process. Since s/he needs visual representations, the first activity is to build a sketch of the process. Sketching means prototyping on a high level of abstraction and requires many changes and refinements. The ability to get tool support in the sketching activity is a very strong requirement. Afterwards, the actual modeling begins. Elements representing the workflow (such as activity, role, or artifact) have to be inserted, refined, changed, and extended by certain attributes. It would be nice to have the possibility to check the model for correctness and adapt elements by deleting, changing, or inserting new elements. Since a process has many process stakeholders, the employee might want to create the workflow model collaboratively or at least export or communicate the model.

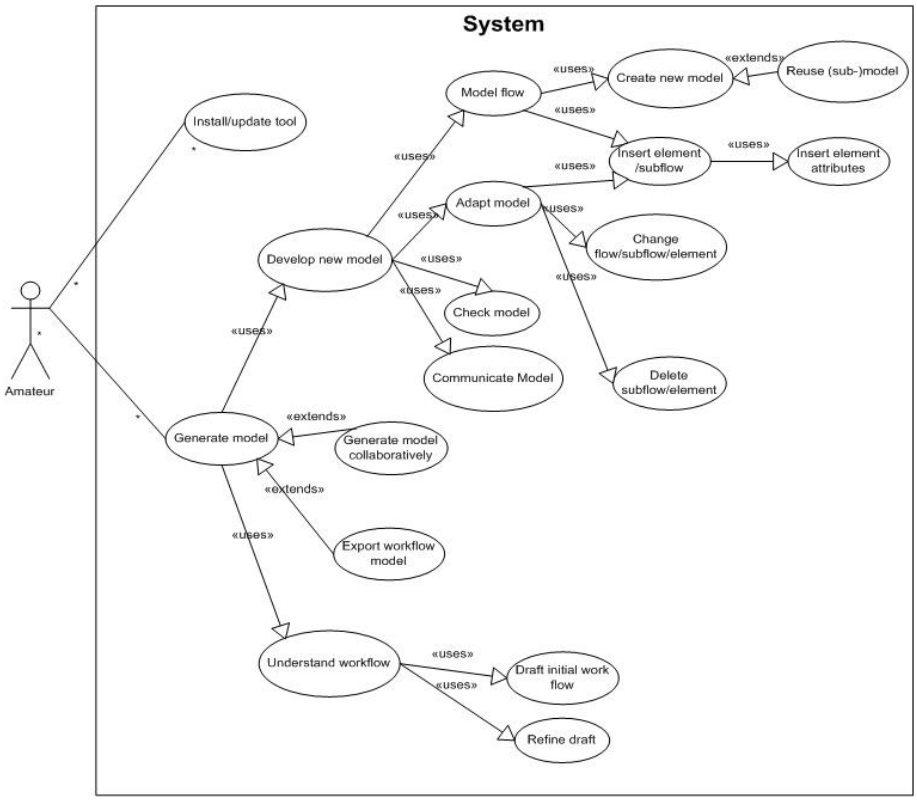

Figure 2: Use-case diagram for workflow modeling tools

After having described each use case in detail, we added usability requirements. These usability requirements were derived from two kinds of sources: Quality models (e.g., ISO9126; ISO9241) and the "Cognitive Dimensions Framework" [12] (cp. Table 1). The latter recommends a set of criteria for the evaluation of notations, programming environments and data visualization.
The result of step 1 was a complete specification of functional and usability requirements for a workflow modeling tool. The next step was to evaluate existing tools against these requirements.

Step 2: Expert Evaluation

In the next step, we explored four existing tools (cp. Table 2) with regard to their capability to accomplish the requirements. For this purpose usability experts from Fraunhofer IESE evaluated two scenarios. In each, they modeled a complete workflow. In Scenario 1, a large and complex workflow was modeled and in Scenario 2, it was a short workflow that was easy to oversee. The scenarios were chosen in order to capture each of the use cases introduced in Figure 2.

Table 1: Quality criteria used to derive usability requirements

Usability Attributes from ISO 9126/ 9241	**Cognitive Dimensions [12]**
Understandability	Viscosity: resistance to change
Learnability	Visibility: ability to view components easily
Operability	Premature commitment: constraints on the order of doing things
Attractiveness	Hidden dependencies: important links between entities are not visible
Usability compliance	Role-expressiveness: the purpose of an entity is readily inferred
Customizability	Error-proneness: the notation invites mistakes and the system gives little protection
Error tolerance	Abstraction: types and availability of abstraction mechanisms.
Conformity with user expectation	
Self descriptiveness	
Efficiency	

For each use case, the fulfillment of the functional requirements, the usability requirements, and the nonfunctional requirements derived from the "cognitive dimensions framework" was evaluated. Table 2 lists the analyzed tools. We will not rate the tools, since our purpose was not to compare the tools, but to identify the best solutions for our (tool-independent) requirements.

Table 2: List of the evaluated workflow modeling tools

Tool	**Description**	**Source**
Oracle BPEL Process Manager 2.0	Infrastructure for creating, deploying and managing BPEL (standard for assembling process flows) business processes.	www.oracle.com
Microsoft BizTalk Server 2004	The MS Visio-Add-In "Orchestration Designer" makes it possible to model business processes for execution in MS BizTalk Server 2004.	www.microsoft.com/biztalk
Essential Business	EBM 1.5 is a tool that combines proven techniques for modeling processes,	www.essmod.com

Modeler 1.5	enabling Model Driven Development of Enterprise Architectures.	
IBM WBI Workbench	IBM WebSphere Business Integration Workbench V4.2.4 is a process modeling tool that makes it possible to test, analyze, simulate, and validate business process models	www-306.ibm.com/software/integration/wbimodeler/workbench/

Step 3: Identification of the best solution

For each positively evaluated functional requirement and for each positively evaluated nonfunctional requirement, the design solution implemented in the tool was analyzed. Every good solution that met our requirements would be a candidate for a workflow-usability pattern. We will give an example for such a best solution in Table 3:

Table 3: Example for a requirement and its corresponding best solution

Requirement	In the use case “create new model“, we phrased the nonfunctional requirement “enable a condensed representation of the complete process”. This requirement was derived from the dimension “visibility” according to the “Cognitive Dimensions Framework” and refers to the ability to view components easily. A complex workflow model can become too large to fit on a single screen. In order to get an overview of every component, a condensed view was required.
Solution	One of the tools offered an elegant solution to this requirement, the Condensed View Feature, which can always be utilized to gain an overview.

For each best solution, we derived a pattern by abstracting from the concrete solution and describing the principles of that specific solution. Table 5 demonstrates an example of one of the workflow patterns.

Step 4: Description of usability pattern

In this way, we were able to identify 40 different patterns. The patterns were classified into several types of patterns as listed in Table 4. Some of these patterns seem to be useful in other domains as well. The basic patterns, for instance, are applicable to almost any kind of application, while the drawing patterns should be found especially in graphic tools. The complete set of categories is probably applicable to any graphical modeling tool.

Table 4: Overview of the identified workflow modeling patterns

Basic Pattern	Autosave Templates
Business Process Pattern	Scopes; Complement attributes; Automatic coupling; Unambiguous attribute names; Unambiguous types of elements; Define types of elements; Facilitate connections; References to other process flows;
Collaborative Work Pattern	Automatic matching of different versions; Color markup; General markups; Multi-user-developing;
Create / Debug Pattern	Auto alert; Auto-correction; Automatic insert; Drop-down boxes; Isolated element deletion; Decisions on demand; Compare screens; Add comments; Context menu; Simulate and test; Sketch; Search; Name symbols; Validate logic;
Documentation & Help Pattern	Documentation & tutorials; Help for attributes; Online help;
Drawing Pattern	Rulers; Conformity to graphic tools;
Format Pattern	Export; Import; Reports
View Pattern	Abstraction levels; Layer; Condensed view; Visibility; Full screen view; Zoom;
Workspace pattern	Adapt workspace; Insert workspace; Notations and working modus; Systematic divisions; Unlimited workspace;

For a detailed description of all patterns, see [13]. To get an impression of the pattern description, in Table 5 we present the view pattern “Condensed View”.

Table 5: Example for the pattern "Condensed View"

Name	Condensed View
Category	View Pattern
Related to	<related pattern names, not only from the workflow pattern collection>
Problem	User wants to gain an overview, or wants to navigate within the workflow model. The model has too many elements and levels of abstractions and cannot be represented on a single screen.
Forces	Condensed representation of the workflow model (visibility); the exact position to insert a new element has to be identified; a specific position within the workflow has to be found; the position of an erroneous element within the workflow has to be identified; easy cognitive walkthrough activity.
Context	Workflow does not fit on a single computer screen and is smaller than 400 symbols.
Solution	Present the complete (downsized) process in a separate window, without scrollbars. Upon double-clicking on a specific spot, center the same spot in the main screen showing the details.
Known Uses	<name of the tool with the best solution>

3 Pattern Evaluation

Usually, pattern descriptions end up in libraries without any empirical validation. Few empirical results are published about the usage of user interface patterns in general [4] [14]. Very few pattern authors set up rules to assure a certain level of quality for their patterns [15]. For example, at “Yahoo!“ [16], a solution has to be used in at least two software systems before it becomes a “pattern”.
To validate the usefulness of the pattern in our project, we wanted to go far beyond this. We formulated the following research hypotheses as a foundation of our investigation:

H: The identified patterns support the software developers in improving the usability of their application.

In order to elaborate this hypothesis we divided it up into the following sub-questions:

> Q1: Do the identified patterns match the design challenges in the domain of workflow modeling tools?
>
> Q2: Do software developers understand the pattern description?

Q3: Does the solution proposed in the pattern description solve the problem stated in the pattern description? (Internal consistency of the pattern)

Q4: Can the solution proposed in the pattern description be transferred to a concrete solution in a software system? (Concretization)

Before exploring each of the hypotheses in more detail by stressing their meaning and showing the results of the case study concerning the questions, we will explain the general design of the case study.

3.1. Design of the case study

The case study was conducted in three steps. We will elaborate the steps while referring to Figure 3 to clarify the rationale of the case study design. Figure 3 illustrates the relationship between the usability problems and the pattern description, as well as the hypothesis/questions that drove the case study.

1. We did a usability test to identify the current weaknesses of the workflow modeling tool. The test was performed with three test users. As part of the test, they had to modify and extend a given workflow, presented in the graphical environment of the workflow modeling tool. In total, 37 usability bugs were identified as result of the usability test. Figure 3 shows the usability problems as little circles in the software.
   The purpose of the usability test was to identify the weaknesses in the current design that could be solved by using the design patterns. Neither the person conducting the usability test nor the test users knew the pattern before.

2. In a second step, we matched the usability problems to problem descriptions in our pattern collection. This step is represented as an arrow with the title “pattern matching”.

3. Afterwards, we conducted an expert evaluation by running guided interviews with three experts. Two interview partners were usability experts. The third one was the lead software engineer of the workflow modeling tool company. Involving experts from both areas was important, because we believe that for the different research questions listed above, expertise from different areas is necessary. The “Understandability” is especially important from the view of the software developer, whereas the question of whether a pattern solution solves a pattern problem should be validated by a usability expert (this question refers to the arrow “concretization” in Figure 3). During the interviews, the 10 patterns were presented one after the other to our interview partners. For each pattern, we investigated Q2, Q3, and Q4. Additionally we asked whether we matched the patterns correctly.

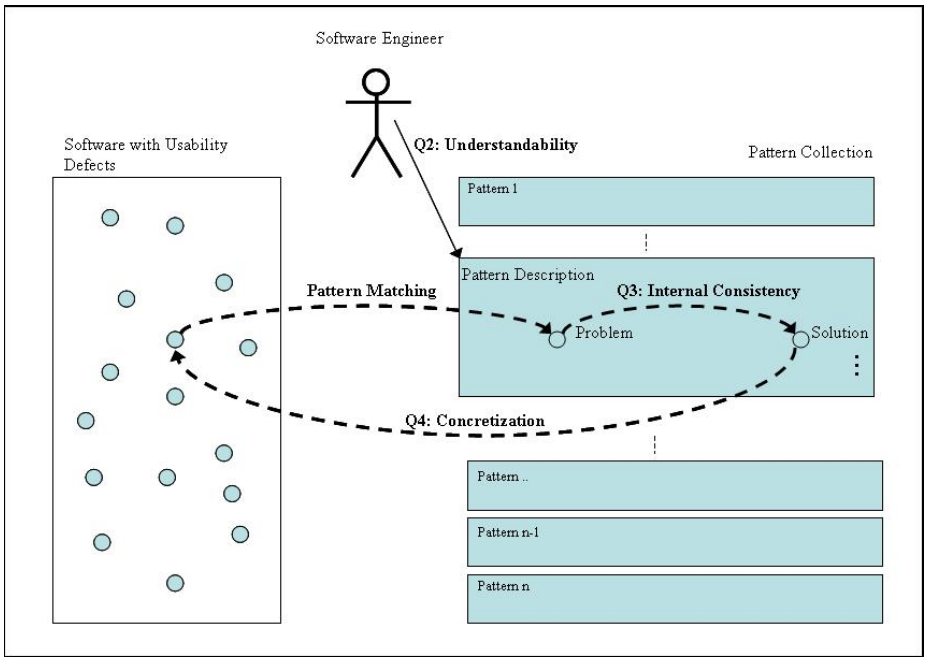

Figure 3: Research questions of the case study

3.2 Contribution to design challenges in the domain of workflow modeling tools

We investigated the question of whether the identified patterns match the design challenges in the domain of workflow modeling tools. Only if the proposed patterns solve challenges in that domain is the pattern collection of any value.
Our case study showed that 10 of the 40 identified patterns matched one or more of the 37 usability bugs. Some bugs matched more than one pattern. In summary, 12 bugs could be linked to patterns. Those patterns not matching any of the identified design problems either do not cover tasks that were set up in the usability test, or the system contained already a usable solution. For example, none of the tasks conducted in the usability test covered the task of debugging a workflow for execution or working on a workflow collaboratively, but 6 of the 40 patterns support these tasks.
For one of the 10 patterns our experts judged the matching between usability defect and pattern as wrong. Of the remaining nine, seven were judged to be valuable contributions to the domain of workflow modeling tools.
The following investigation refers to the 9 “matching” patterns of our collection. The case study has to be extended in order to make a statement concerning the remaining 30 patterns.

3.3 Understandability of patterns

We investigated the question of whether software developers understand the pattern description. The appropriate wording is the precondition for their usage by software engineers.
8 of the 9 pattern descriptions were judged to be understandable. Nevertheless, our interview partners gave us hints onto improve the description for all nine patterns. Terms from the domain of workflow modeling tools have to be worked out more properly in order to improve the understandability of the current description. Also, the names of the pattern should be made more specific and recognizable.

3.4 Internal consistency of the patterns

We wanted to find out whether the solutions described as part of the pattern descriptions really solve the problems stated in the problem description of the pattern. We call this internal consistency of the pattern. Only if the internal consistency is given for a pattern, it can improve the usability of a system.
8 of the 9 patterns under investigation were judged to be internally consistent. For one of the patterns, we got a suggestion to improve the solution, and for one pattern, an additional, alternative solution was proposed.

3.5 Applicability of abstract pattern solution to a concrete software design solution

Even if a pattern is internally consistent, its description of the solution is understandable, and the selected pattern could be matched to a given usability problem of the software system, it might happen that the pattern cannot be transferred to a usable solution in the software system. What remains as a possible pitfall is the step of concretization, which means the transfer of the abstract pattern description to a concrete solution in the software system. This step has to be made by the software developer when applying the pattern. Figure 4 elaborates this problem in more detail.
The solution <s> given in a pattern description <P> is an abstraction of a concrete solution <s*> found in another software system X. For example, layout or color details as well as very detailed interaction steps were not specified as part of the pattern description. When applying pattern P to software system Y, the solution <s'> is a concretization of <s>. <s'> may differ in many details from the original solution <s*>. If the pattern description is not complete or leaves design decisions open that are important to address Problem <p'>, concretization might fail and end up in a solution <s'> that does not improve the usability of system Y.

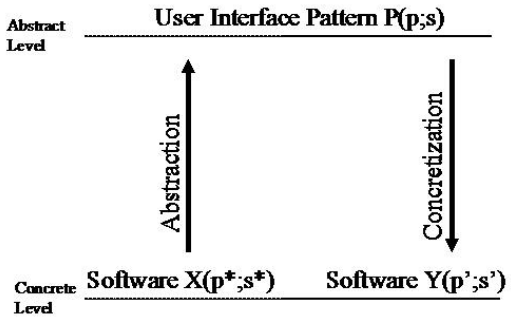

Figure 4: Abstraction and concretisation of patterns

Our experts judged 8 of the 9 patterns to be suitable for deriving a concrete design solution. For a second pattern, one expert proposed a more concrete description.
We believe that we need to run additional experiments to gain more insights into the problem of concretization. A guided interview is not well suited for investigating this. Running usability tests on the next version of the software that contains implementations of the suggested pattern could show us whether the concrete solutions really improve usability.

4. Conclusion and Future Work

The findings support our main hypothesis: For 7 of the 9 patterns, the evaluation showed that the identified patterns support the software developers in improving the usability of their application. As a positive side effect, we found a lot of valuable hints to improve the pattern descriptions. The positive judgment of the patterns under investigation can be interpreted as evidence for the quality of the process we used to identify the patterns.
The quality requirements for user interface patterns – like understandability, internal consistency, and ability for concretization - worked out in the design of the case study, gave us further ideas how to improve the process of pattern identification. With our future work, we want to enrich the process with guidelines for pattern description in order to improve step 4 of the pattern identification process. The guidelines should formalize the consistency between solution and problem and the process of abstraction. As a consequence, the probability of deriving "high quality" pattern collections will increase.
Controlled experiments should investigate the contribution of our patterns to the quality of the end product in terms of statistically valid data. We are also thinking about evaluating the identified pattern in a "design from stratch"-experiment. This experiment will investigate how patterns not only improve a given design, but also support the new design of a system.
In future projects, we want to extend our research to process guidance for software developers in finding the right pattern description in a given pattern collection or

library. Only if this step is sufficiently well supported can user interface patterns support software developers in improving user interface design in their daily work.

A**cknowledgements:** The authors wish to acknowledge the contributions of Steffen Hess with respect to the process of pattern identification and Jörg Grimm in conducting the pattern evaluation study.
The project was performed with financial support from „European Regional Development Fund“ and the state “Rheinland-Pfalz” (Förderkennzeichen: MWVLW, Az.: 8315 38 51 04 IESE, Kapitel 0877 Titel 892 02)

References

[1] D. Kerkow, K. Schmidt, and F. Wiebelt, "Requirements for the Integration of UE Methods in SE Processes from the Perspective of Small and Medium-sized Enterprises (SMEs)," presented at INTERACT Workshop: Integrating Software Engineering and Usability Engineering, Rome, 2005.
[2] R. N. Griffiths and L. Pemberton, "Don't write guidelines write patterns!," vol. 2006.
[3] A. Dearden, J. Finlay, and L. A. B. McManus, "Using Pattern Languages in Participatory Design," presented at Participatory Design Conference, Palo Alto, 2002.
[4] N. L. O. Cowley and J. L. Wesson, "An Experiment to Measure the Usefulness of Patterns in the Interaction Design Process," presented at Tenth IFIP TC13 International Conference on Human-Computer Interaction (INTERACT), Rome, Italy, 2005.
[5] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, Design Patterns: Elements of Reusable Object Oriented Software: Addison-Wesley, 1995.
[6] A. Seffah, M. Desmarais, E. Metzker „HCI, Usability and Software Engineering Integration: Present and Future” in Human-Centered Software Engineering. Edited by A. Seffah, J. Gulliksen and M. Desmarais, Springer, 2005
[7] J. Tidwell, "COMMON GROUND: A Pattern Language for Human-Computer Interface Design," vol. 2006, 1999 (last updated).
[8] M. v. Welie, "Patterns in interaction design," 2003.
[9] I. Graham, A Pattern Language for Web Usability. London, 2003.
[10] A. Richter, "Generating User Interface Design Patterns for Web-based E-business Applications," presented at Interact Workshop: Software and Usability Cross-Pollination: The Role of Usability Patterns, 2nd IFIP WG13.2 Workshop on Software and Usability, 2003.
[11] J. Borchers, "Interaction Design Patterns: Twelve Theses. Position Paper, Workshop ``Pattern Languages for Interaction Design: Building Momentum"," presented at Workshop "Pattern Languages for Interaction Design: Building Momentum", CHI 2000, The Hague, Netherlands, 2000.
[12] T. R. G. Green and M. Petre, "Usability Analysis of Visual Programming Environments: A 'Cognitive Dimensions' Framework," Journal of Visual Languages and Computing, vol. 7, pp. 131-174, 1996.
[13] K. Kohler, D. Kerkow, S. Hess, and K. Schmid, "Best Practices und Usability Pattern für Geschäftsprozess-Modellierungswerkzeuge," 060.05/D, 2005.
[14] J. Wessen and L. Cowley, "Designing with Patterns: Possibilities and Pitfalls," presented at Interact Workshop: Software and Usability Cross-Pollination: The Role of Usability Patterns, 2nd IFIP WG13.2 Workshop on Software and Usability, 2003.

[15] E. Todd, E. Kemp, and C. Phillips, "What makes a good user interface pattern language?," presented at Proceedings of the fifth conference on Australasian user interface - Volume 28, Dunedin, New Zealand, 2004.
[16] M. Leacock, E. Malone, and C. Wheeler, "Implementing a Pattern Library in the Real World: A Yahoo! Case Study," 2005.

Questions

Gerrit van der Veer:

Question: The presented approach is very systematic, based on (1) finding a problem, (2) analyzing the problem, (3) finding a solution and (4) validating the solution. Unfortunately the approach has not been applied and tested in different domain. This raises the issue of its generality.

Answer: There is no doubt that the approach and the presented ideas should be tested in different domains as well. This may be part of future work.

Peter Forbrig:

Question: How does the pattern specification relate to workflow specifications?

Answer: The solution part of the patterns may (informally) entail information which can be used to derive (in part) a workflow specification.