Enhancing Business Software through Fun-of-Use: A Pattern-based Approach

Christian Graf, Sabine Niebuhr, Kirstin Kohler

Fraunhofer-IESE, Kaiserslautern, Germany

{christian.graf | sabine.niebuhr | kirstin.kohler}@iese.fraunhofer.de

http://www.iese.fraunhofer.de

Abstract. Buying decisions for software products can be influenced by arousing positive emotions, as they increase acceptance of the software product. Therefore we will present a way, how these positive emotions can be generated within business applications. By capturing design knowledge about joyful user interfaces in interaction pattern, we provide systematic guidance on engineering fun. Our patterns base on psychological theories dealing with intrinsic motivation and derived from principles known form game design and e-learning. Implementing the patterns in real business applications we made experiences amongst others in describing them properly and their feasibility in practice.

Keywords: user-interface pattern, design methodology, joy-of-use

1 Introduction

Successful software products should arouse positive emotions, since this is another factor next to the product's reliability and functional range which influences buying decisions - and it is assumed, that its relative importance will increase within the next years. Especially business applications seldom differ in functionalities, why adding hedonic qualities and generating positive emotions is even more important for the acceptance of this kind of applications.

Therefore, in our FUN-project (http://www.fun-of-use.de - a research project supported by the German government) we focussed on the question, how business applications can be designed triggering positive emotions to the user and how this can be supported by software engineering methods.

In recent years more and more the necessity has been stated for systematic guidance to realise user experience (e.g. Shneiderman 2004). Within the FUN-Project we identify interaction patterns to enhance joyful experiences in order to increase the user acceptance.

Patterns as source of design knowledge are getting more common in recent years, since they were originally introduced for architecture (Alexander 1977). Describing approved solutions for often occurring problems in a structured way, they are now well suited in software engineering areas like design (e.g. Gamma

et al 1995), analysis (e.g. Fowler 1997), or architecture (e.g. Shaw 1995). Tidwell (2005) and van Welie (2000) described large pattern collections for interaction and user interface design problems, which became famous in the area of Usability Engineering. Although many pattern descriptions followed, those describing hedonic aspects are still missing.

Starting with identifying these patterns our main challenges have been, where to extract them from, how to describe them properly, and how to apply them in practice. In the following we describe, how we dealt with these questions.

The goal of this paper is to present our approach for identifying patterns and how we implement them with practitioners for watching their feasibility in practice and to test their effects in following experiments.

We will present the identified patterns' theoretical background, where we focused on e-learning and game design. Since they have their findings from psychology, sociology, and didactics, we also try to find our theoretical foundations in these fields. Figure 1 depicts these correlations between the pattern's theoretical background down to the design recommendations given by them. Beside the background and the distilling process we will describe the challenges we had while we implemented some of the patterns with industry partners, as well as the experiences we made in this process.

2 Theoretical Background of Fun-Patterns

Humans experience joy from being kept by a thrilling exercise that they are able to master with some effort. For example, these experiences occur while being involved in games or well-designed learning environments.

Motivation is a key element to engage in actions and enjoy such experiences. Typically the motivation is intrinsic, coming from the subject or activity itself and not from rewards offered externally. It may be the general interest in a topic, expected pleasures, the inner drive to accomplish some goals, pure curiosity, and thirst for knowledge. Intrinsic motivation depends on the acting person itself and not on the quality or quantity of outer properties. In contrast extrinsic motivation is alien to the system of acting person and activity. Most times it consists of materialistic or social rewards (e.g. money, association with a group) that have to be increased in quality or quantity with time to yield the same effects as before. To put it simple: Extrinsic motivation is "expensive" in terms of resources needed but easy to set-up, intrinsic is "cheap" but hard to put in place.

Mixing intrinsic and extrinsic motivation was initially said to undermine intrinsic motives such that extrinsic motivation replaces intrinsic motivation (Deci 1971, Lepper et al. 1982). This view was challenged by several authors (e.g. Cameron & Pierce 1994, Weaver at al. 2003, Harrington and Wallace 2005) that were in turn re-challenged (e.g. Deci et al. 1999). Other researchers argue that extrinsic and intrinsic motivation may work together in harmony and may even amplify each other (Cameron et al. 2001, Reinholt 2006). As the effects of the combination of both types of motivation are not completely clear we will concentrate on intrinsic motivation as the more lasting and rewarding source of joyful experiences. Lindenberg (2001) coined the term "enjoyment-based intrinsic motivation". Concentrating on enjoyment-based motivation means extra

challenges when producing fun-enhanced software. Instead of relying on extrinsic motivators, properties of the activity that the software should support have to be identified. Properties can then be changed, added or deleted to possibly gain a more motivating and joyful interface that provides a positive experience to the user.

In general, intrinsic motivation is regarded as one key aspect for having fun (Csikszentmihalyi 1975). Facilitating intrinsic motivation is the key to let people enjoy fun during their work. This is the main approach behind Fun-Patterns.

3 Fun-Patterns to Support Motivation

Fun-Patterns are meant to create environments to support and foster intrinsic motivation. Under this premise we began identifying potential candidates for Fun-Patterns by examining game design and learning environments. Both subjects are viewed from a perspective from computer science, meaning that we gave special attention to concepts from computer games and e-learning. Each of the topics has a different approach to view motivation and involves a manifold of adjacent topics, like sociology, didactics, psychology and human factors. We analysed literature elaborating on such diverse topics like motivation, engagement, joy-of-use, creativity and experiences on the question what concepts are connected with the drive of people doing things and enjoying it. From this analysis we gathered recommendations for interaction design that were distilled to a selection of user-interface pattern to trigger fun (schematically visualised in Figure 1).

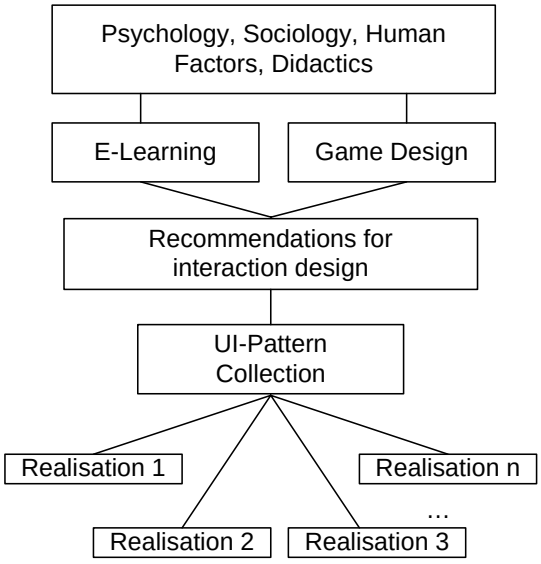

Fig. 1. Top-down process of the pattern development: from established high-level concepts to distilled patterns and concrete realisations

3.1. Patterns from E-Learning

In the field of learning it is crucial to motivate the learner to engage voluntarily in the learning activity, especially in the case of e-learning because it is mostly unsupervised. Positive emotions improves learning to Damasio (1993) showed that learning improves if the learner is emotionally touched. Examining literature about e-learning brings up a diversity of recommendations to foster and support motivation of the learner (e.g. Malone & Lepper 1987, Hops 2003, Sim et al. 2005).

One often found is to accommodate the learner in all possible situations. Thus learning software is often highly adaptable to the learner's needs and skill levels. The user should be supported in achieving his goal and keeping him from being either asked too much. The more the learner knows the more difficult becomes the content. This can be transferred to interface design. An expert interface with a wide range of functions is neither enjoyable nor usable for a novice user. A professional user on the other hand might be frustrated by an interface made for beginners. Thus the user interface must fit the abilities of the user. We coined this proposed pattern "competence dependent UI". It is described here in an abbreviated form that only shows the main categories of a typical pattern description (see for example Fincher 2003):

Name	Competence dependent UI
Problem	An user's competences evolve or users with different competences use the system.
Forces	Novice users are scared off by highly complex interfaces. Experienced users do not want their skills being limited by too elementary interfaces.
Solution	Either detect the skill level of the individual user or give users the change to explicitely change their skill level. Then adopt the user interface.

3.2. Patterns from Games

Like mentioned above, computer games are a kind of software used voluntarily. They are designed to be so exciting, that many users keep on playing a game over hours and days. Games contain hedonic qualities, not task-based aspects like novelity or originality, as Hassenzahl et al. (2000) discribe, and states, that these qualities increase the user's acceptance. So for us it was a logical step to look in game design literature to learn from experiences and research in this field.

For developing a successful game, Shelley suggests to “create a range of different gaming experiences in a number of different ways, all within the same game” (Shelley 2001). He argues with the broad audience of computer games. Business software also has a broad audience with different user goals and expectations. Each user has a different attitude towards fulfilling a business goal and realising own goals. Therefore the approach of a “range of game experiences” could be mapped to a “range of working experiences” by a number of different ways fulfilling a given task. The proposed pattern – we called it “target area” – is depicted in the following:

Name	Target area
Problem	Users have different experiences and different attitudes in fulfilling tasks and goals.
Forces	The task or the goal should be able to be reached in several nuances or different occurences.
Solution	Give users a range of possibilities fulfilling a task or reaching a goal.

If you are going to design a presentation for example, you have different possibilities to fulfil this task – depending on how much you are going to embellish it: You can keep it very simple and write down black coloured text on white slides, or you can add animations to objects, you can use different colours or design playful changes between the slides. The general result of this task therefore is a range of different results in the target area “presentation”.

4 Realisations of Fun-Patterns

Up to this stage of the project we and our project partners have identified two to three Fun-Patterns to be put into some reference application that will be tested in an laboratory environment. Those two reference applications are typically for the domains of our project partners and can be viewed as prototypical instances of later commercial products.

During this practical work we have identified two main challenges for which we will propose solutions:

1. The quality of a set of patterns is only as good as the quality of each item and its description of the relations to others. Such interrelated sets are often called languages. Thus we need a language of patterns that can be used for interaction patterns and that might be easily customized.

2. Patterns of any kind by definition are only abstract solutions to certain classes of problems. Thus the solution presented in one pattern must be customized for the concrete context of the problem. Otherwise it might happen that the implemented solution does not have the expected positive consequences or that it might conflict with other patterns and thus thwart the intended effect.

These challenges were tackled by different means.

1. We were looking at existing pattern description templates and examining existing pattern languages in terms of fulfilling our needs in describing patterns. Thus we identified the Pattern Language Mark-Up Language (PLML) as the most recent and thorough work to start with. This language and the corresponding template encompass many concepts found in software or architectural patterns transferred to interaction design (Fincher 2003). We are about to extend PLML by some concepts that were missing, especially to support the application and customization of the pattern for the later user, be it a programmer or a designer. Thus the pattern description may help the programmer when implementing the concept, but the designer has the opportunity to model these concepts into the application from the start on. Thus anticipating the results and consciously shaping the user experience is at hand.
2. By doing workshops with the project partners we learned about their domain, the context and the application area of the to-be-enhanced-software. We used methods like interviewing, creativity elicitation through associative thinking and brainstorming. Through close cooperation with our partners a set of patterns were identified that bear the potential of being successfully implemented in the reference applications. Our partners will implement these patterns and we will test those implementations. As only the emotional side of the interaction will be influenced we can learn what patterns have a positive effect on the motivation of the user.

5 Lessons Learned

From our work we condensed recommendations for interaction design of joyful interfaces. But not every potential candidate was identified as an appropriate pattern. To be a Fun-Pattern it had to satisfy three conditions:

1. It must be an applicable solution to a certain class of problems – not to abstract (like high-level principles), not to concrete (e.g. a practical realisation for a specific problem in a specific context).
2. It must be a description of means to reach a goal-state and not a mere description of that desirable state.
3. It must be unique in the way that no other patterns encompass the same idea in the same manner. General ideas to evoke fun and realisations of recommendations are not patterns.

Applying a pattern in the collection to real world situations means identifying a specific problem and abstracting it to a generic problem description. That description is then matched to the problem descriptions in each pattern. After having identified a matching pattern, it has to be customised to the specific problem and the overall goal of interaction striven for in the specific context. The last step is to check if the specific problem is suspended by the specific solution (the realisation of the pattern). The whole process is illustrated schematically in Figure 2.

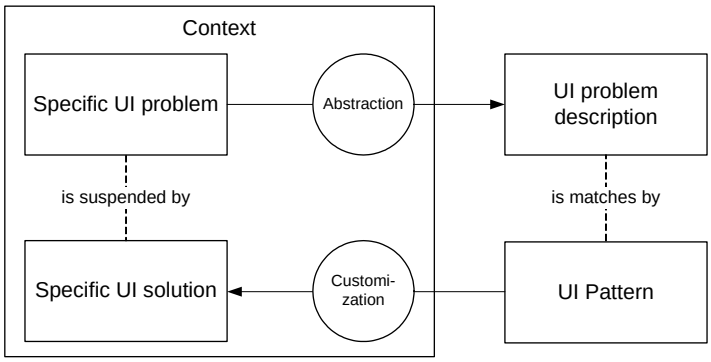

Fig. 2. Fun-Patterns in use: Abstraction of the problem and customization of the pattern by the developer

As we have seen the solution described in a Fun-Pattern has to be customised to the context of the problem environment to form realisations that practically solves the problem (see Figure 2). In contrast to patterns in software-development this step cannot be automated yet, as the diversity of constraints, the inter-relations of patterns and humans' reactions to it are to complex (Dearden & Finlay 2006).

6 Summary and Future Work

In context of the fun-project we are engaged in how positive emotions can be triggered with software and how this knowledge can be used in the software developing process. We decided to express our experiences in interaction patterns and try to realize fun-of-use in business applications with this approach. Therefore we searched in game and educational literature for existing approaches, guidelines and heuristics. A further source had been literature about motivation. Based on this we extracted a first set of patterns.

Our future work will be developing a comprehensive pattern language for interaction patterns especially describing fun-of-use, implementing selected patterns in test applications for analyzing their effect and relevance, and redesigning the tested patterns if necessary.

In a first step, some patterns will be implemented and their effect will be evaluated in a lab-setting. The pattern's effect on users will be measured by a

comparison of two versions of a software system – the system with implemented pattern compared with a version of the system without pattern. This comparison will contain the user's interaction behaviour as well as his subjective appraisal and the result we expect to achieve with this pattern.

During the second project phase we will apply the pattern approach for interaction design in the industrial setting of our partners within their development projects. The goal of this phase is to investigate and prove the usefulness of patterns for software developers.

The emotional side of interaction can be designed by using patterns without the need to have a background in psychology, game design, or e-learning. This bears the hope that in the future the users will meet more enjoyable user interfaces of business applications and therefore their acceptance will increase.

References

Alexander, C. (1977). A Pattern Language. Oxford University Press. New York.

Cameron, J.; Banko, K. & Pierce, W.D. (2001). Pervasive negative effects of rewards on intrinsic motivation. The myth continues. *The Behavior Analyst, 24*, 1-44.

Csikszentmihalyi, M. (1975). Beyond Boredom and Anxiety: Experiencing Flow in Work and Play. Jossey Bass Publishers, San Francisco, Washington, London.

Dearden, A. & Finlay, J. (2006). Pattern Languages in HCI: A Critical Review. *Human-Computer Interaction*, 21, 49-102.

Deci, E. L. (1971). Effects of externally mediated rewards on intrinsic motivation. *Journal of Personality and Social Psychology, 18*, 105-115.

Deci, E. L., Koestner, R., & Ryan, R. M. (1999). A meta-analytic review of experiments examining the effects of extrinsic rewards on intrinsic motivation. *Psychological Bulletin, 125*, 627-668.

Fincher, S. (2003). Perspectives on HCI Patterns: Concepts and tools (introducing PLML). CHI 2003 Workshop Report. *Interface*, *56*, 26-28.

Fowler, M.(1997) Analysis Patterns—Reusable Object Models. Addison-Wesley Object-Oriented Software Engineering Series. Addison-Wesley.

Gamma, E. & Helm, R & Johnson, R. & Vlissides, J. (1995) Design Patterns – Elements of Reusable Object Oriented Software. Addison-Wesley.

Harrington, S. & Wallace, M. D. (2005). *Effect of Reinforcement Schedules on Intrinsic Motivation and the Overjustification Effect.* Paper presented at the 31st Annual Convention of the Association for Behavior Analysis, Chicago, IL. May 2005.

Hassenzahl, M., Platz, A., Burmester, M., and Lehner, K.. Hedonic and ergonomic quality aspects determine a software's appeal. In CHI '00: Proceedings of the SIGCHI conference on Human factors in computing systems, pages 201--208, New York, NY, USA, 2000. ACM Press.

Hops, D. (2003). Konzeption und Entwicklung einer Motivationskomponente für E-Learning Systeme. Diplom-Thesis, Department of Computer-Science, RWTH Aachen University.

Lepper, M. R., Sagotsky, G., Dafoe, J. L., & Greene, D. (1982). Consequences of superfluous social constraints: Effects on young children's social inferences and subsequent intrinsic interest. *Journal of Personality and Social Psychology*, 42, pp. 51-64.

Lindenberg, S. (2001). Intrinsic Motivation in a New Light. *KYKLOS, 54*, pp. 317-342

Malone, T. & Lepper, M. (1987). Making learning fun: A taxonomy of intrinsic motivations for learning. In: R. E. Snow & M. J. Farr (Eds.), *Aptitude Learning and Instruction: III. Conative and Affective Process Analysis.* Hillsdale, NJ: Lawrence Erlbaum.

Reinholt, M. (2006). No More Polarization, Please! -Towards a More Nuanced Perspective on Motivation in Organizations. SMG Working paper, nr. 2006-009. Center for Strategi og Globalisering. København, 2006.

Shaw, M. (1995). Patterns for Software Architectures. In Coplien, J. & Schmidt, D., editors, *Pattern Languages of Program Design*, volume I

Shelley (2001). Guidelines for developing successful games. http://www.gamasutra.com/features/20010815/shelley_01.htm (last retrieved 21-08-2006)

Shneiderman, B. (2004). Designing for fun: how can we design user interfaces to be more fun?. *Interactions, 11*, 5, pp. 48-50.

Sierra, K. (2005). Never Underestimate the Power of Fun. Creating Passionate Users Blog. http://headrush.typepad.com/creating_passionate_users/2005/12/never_underesti.html (last retrieved 21-08-2006)

Sim, G., MacFarlane, S. & Horton, M. (2005). Evaluating Usability, Fun and Learning in Educational Software for Children. In P. Kommers & G. Richards (Eds.), *Proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications 2005* (pp. 1180-1187). Chesapeake, VA: AACE.

Tidwell, J. (2005). Designing Interfaces. O'Reilly Media.

van Welie, M. (2000). Hallvard Traetteberg. Interaction patterns in user interfaces. 7th Pattern Languages of Programs Conference. Allerton Park Monticello, Illinois, USA. http://www.welie.com/patterns/index.html (last retrieved 21-08-2006)

Weaver, A.D.; Watson, T.S.; Cashwell, C.; Hinds, J. & Fascio, S. (2003). The effects of ability- and effort-based praise on task persistence and task performance. *The Behavior Analyst Today*, *4*, 127-133.

Acknowledgement

This work was supported by the German "Bundesministerium für Bildung und Forschung" (www.bmbf.de) under grant 01 IS E06 A