Computers in Human Behavior 26 (2010) 452–463

Contents lists available at ScienceDirect

Computers in Human Behavior

journal homepage: www.elsevier.com/locate/comphumbeh

Building an interaction design pattern language: A case study

Stefan L. Pauwels *, Christian Hübscher, Javier A. Bargas-Avila, Klaus Opwis

University of Basel, Faculty of Psychology, Department of Cognitive Psychology and Methodology, 4055 Basel, Switzerland

ARTICLE INFO

Article history:
Available online 18 January 2010

Keywords:
Design patterns
Pattern languages
Interaction design

ABSTRACT

Interaction design patterns are a proven way to communicate good design. However, current pattern collections are not sufficiently powerful and generative to be used as a guide for designing an entire application such as those used in complex business environments. This study shows how we built and validated interaction design patterns that serve as the specification for the redesign of an application. Additionally, they were integrated into a pattern language, as a ruleset for human–computer interaction (HCI) non-professionals to continue development of the application. We demonstrate how individual phases in the redesign of an application can be matched with the process of creating an interaction design pattern language. To facilitate the writing of individual interaction design patterns as well as the development of the pattern language as a whole, a combination of user interviews, controlled experiments and analytical methods has been applied successfully.

1. Introduction

1.1. Interaction design patterns

Design patterns describe good solutions to recurring design problems in specific contexts. The concept of design patterns was originally developed by Christopher Alexander as a method of capturing and communicating good architectural design (Alexander, 1979). Each of Alexander's design patterns has a unique name, a numerical ID and gives an overview of the pattern's context and what the solution is about, mostly in the form of a short summary and a picture or figure (see Fig. 1 for an example). The overview is followed by a detailed description of the problem, how to implement the solution, a rationale why the solution is good and in what context the design pattern should be applied (Alexander et al., 1977).

The differentiation between problem and context in the detailed pattern description seems noteworthy: Multiple design patterns can solve the same problem for different contexts. Consequently, multiple design patterns can have similar or even identical statements as their problem attribute. Many pattern authors use the term "forces" to describe the constraints that define a problem. By contrast, the context attribute should make clear, when to choose one pattern over the other. This crucial attribute can be defined by a list of conditions that must apply for a pattern to be justified or by a careful description of the context.

The design pattern concept was later adopted by software engineers. Gamma, Helm, Johnson, and Vlissides (1995) described a collection of influential software design patterns which are now widely used. Software design patterns differ from Alexander's design patterns in an important aspect: Software design patterns are developed by and for professionals whereas Alexander's architectural design patterns were specifically designed to give non-professionals the power to create good design (Alexander, 1979).

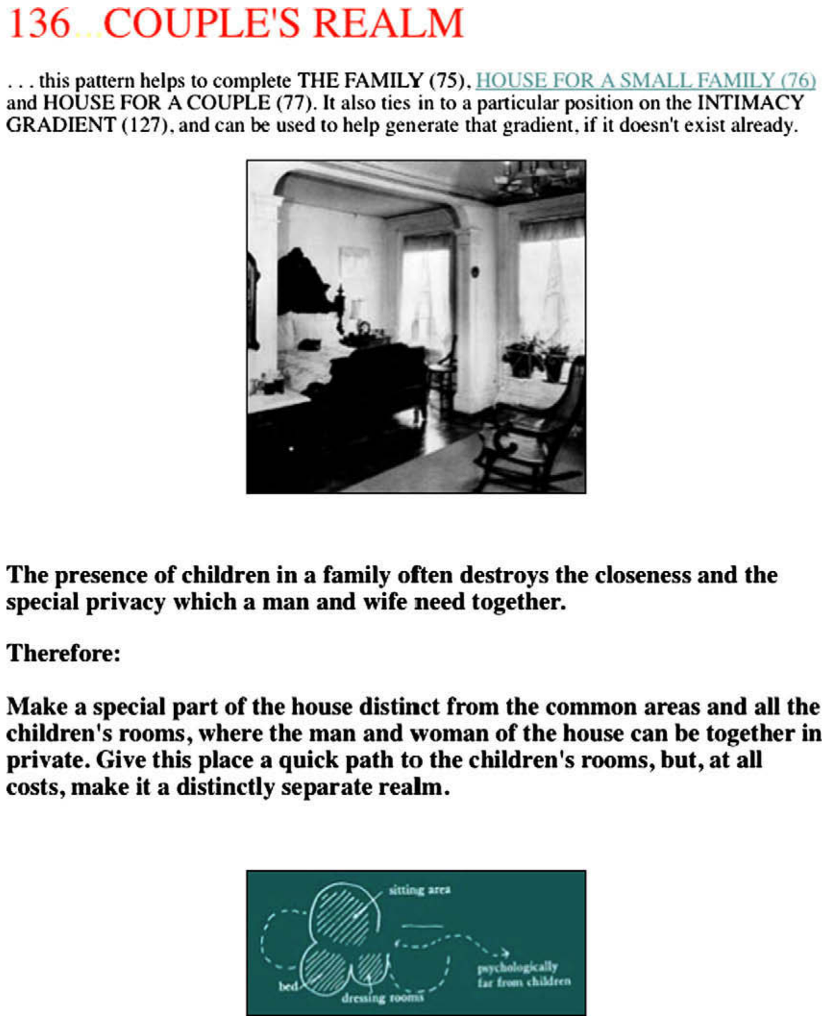
In recent years, design patterns have found their way into the field of human–computer interaction (HCI). Early HCI-related patterns appeared at the pattern languages of programming (PLoP) conference and pattern workshops began emerging at the computer–human interaction (CHI) conference (Bayle et al., 1998). Since then many pattern libraries have been published (Tidwell, 2005; Van Duyne, Landay, & Hong, 2007; Van Welie, 2008; Yahoo! Inc., 2006) and more are appearing each year. Fig. 2 shows an example of a typical interaction design pattern. In the different implementations of the design pattern concept in HCI, the internal structure of a pattern has mostly stayed true to Alexander's pattern form; the attributes' names however vary among implemented design pattern collections. Table 1 shows an overview of typical design pattern attributes used in HCI.

E-learning is another emerging area of application for design patterns that is closely related to HCI. Recent studies analyzed the design of learning environments by evaluating solutions to various problems as design patterns (Diggelen & Overdijk, 2009) whereas other studies looked into techniques of finding and writing e-learning and collaborative learning design patterns (Kohls & Uttecht, 2009; Winters & Yishay, 2009).

Dearden and Finlay (2006) proposed the term interaction design pattern to define design patterns in the HCI field because they state solutions in terms of perceived interaction behavior of an interface. This enables a clear distinction between interaction design patterns

* Corresponding author. Tel.: +41 61 2673568.
E-mail address: stefan.pauwels@unibas.ch (S.L. Pauwels).

0747-5632/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.chb.2009.12.004



136 . . . COUPLE'S REALM

. . . this pattern helps to complete THE FAMILY (75), HOUSE FOR A SMALL FAMILY (76) and HOUSE FOR A COUPLE (77). It also ties in to a particular position on the INTIMACY GRADIENT (127), and can be used to help generate that gradient, if it doesn't exist already.

The presence of children in a family often destroys the closeness and the special privacy which a man and wife need together.

Therefore:

Make a special part of the house distinct from the common areas and all the children's rooms, where the man and woman of the house can be together in private. Give this place a quick path to the children's rooms, but, at all costs, make it a distinctly separate realm.

Fig. 1. Overview of Christopher Alexander's design pattern *Couple's realm* (Alexander, 2001).

used in interface design and software design patterns whose solutions focus on source code and software structures.

Similar to Alexander's original design patterns, interaction design patterns are written for professionals and non-professionals alike. Interface design often involves people from a broad, interdisciplinary field of designers, developers, business analysts, researchers and users (Borchers, 2001) who need to have a common understanding of design problems and solutions in order to cooperate effectively. Interaction design patterns enable the communication of design solutions among co-workers of various fields (HCI, IT, business) or users for participatory design (Dearden, Finlay, Allgar, & McManus, 2002).

Design patterns are essentially a way of structuring knowledge and not a method to find new solutions to problems. Solutions described in design patterns need not be new or original but should be proven to work in practice. Consequently, design patterns are not derived from theory but identified as invariant aspects of solutions that emerge as best practices. The identification of these invariants is often referred to as *pattern mining* (Dearden & Finlay, 2006).

Successful use of interaction design patterns is reported for example by Lin and Landay (2008), who have used design patterns as a central part of a prototyping tool. They showed that designers who made more use of the available design pattern language were able to produce better results than those using the patterns less or not at all. Borchers (2001) reports another successful interaction design pattern case: Interaction design patterns were created based on results of a user-centered design (UCD) project and were successfully reused later in similar interface design projects. Apart from using interaction design patterns directly for the design process, Hughes (2006) proposes using them to conserve knowledge gained from usability studies.

1.2. Pattern languages

A single design pattern has a small impact on the design process of a graphical user interface (GUI). To leverage the design pattern concept it is usual to integrate multiple related patterns into a pattern library. Some pattern libraries have been published either as books (Tidwell, 2005; Van Duyne et al., 2007) or online (Van Welie, 2008; Yahoo! Inc., 2006). Public pattern libraries such as the above-cited are collections of interaction design patterns of varying size and scope that the respective authors have observed or applied themselves time and time again.

In order to connect the individual design patterns of a library, an important aspect of a design pattern is its relation to other patterns. Thus, rules for a design pattern's use – its context – can consist of references to other patterns. GUI design solutions can be encapsulated through design patterns that inherit from or contain each other, not unlike classes in object-oriented programming. The connection between design patterns is already aparent in Alexander's patterns (see Fig. 1). Typical interaction design pattern collections link individual design patterns in a "related patterns" section, where alternative solutions to similar contexts or patterns, which