

# User Centered Design Patterns and Related Issues – A Review

**Sara Imtiaz**

*Department of Computer Software Engineering  
National University of Sciences and Technology,  
Islamabad, Pakistan.*

*saraimtiazcheema@gmail.com*

**Arif Raza**

*Department of Computer Software Engineering  
National University of Sciences and Technology,  
Islamabad, Pakistan.*

*arif\_raza@mcs.edu.pk*

---

## Abstract

A design pattern describes possible good solutions to common problems within certain context. This is done by describing the invariant qualities of all those solutions where good patterns improve with time and widespread use. In this research paper some existing user centered design patterns and their issues are discussed. We have studied many user centered design patterns; however most of them do not provide diagrammatic solutions which can be implementable. It is observed that there is a need of a design pattern which can address issues specifically related to Open Source Software (OSS) users.

**Keywords:** Open Source Software (OSS), Human Computer Interaction (HCI), Users, User Centered Design, Design Patterns.

---

## 1. INTRODUCTION

A pattern describes possible good solutions to a common design problem within a certain context, by describing the invariant qualities of solutions [1]. A design pattern language is a collection of design patterns which are structured so that patterns cross-reference other patterns in the set [2]. Christopher Alexander posits that good patterns improve with time and widespread use [1]. A good design pattern consists of several different components. There are the content elements, which encapsulate the description of the problem, context and solution. Relationships between patterns are essential part to be benefitted from the full reuse potential promised [2]. User involvement for the user satisfaction in designing interface is a necessary part. Like all languages, a pattern language has vocabulary, syntax, and grammar. The difference is that the language is applied to some complex activity other than communication. [3]

In OSS projects usability is one of major issues besides software functionality to make it a good usable product [4]. There are architectures and patterns including HCI patterns, which are used in software development.

Following sections consist of a detailed review of HCI oriented design patterns, their applications and relevant issues. A general discussion is presented in Section 2 about HCI oriented design patterns. In Section 3, issues related to some user centred design patterns are discussed.

## 2. HCI ORIENTED DESIGN PATTERNS – IN GENERAL

According to [5], diagrams included in software design patterns help in understanding the patterns. In HCI oriented design patterns, the focus of diagrams is on user point of view. However the pattern language is not specific to the single user interface, style or guidelines. The authors state that designers should communicate with users to understand their requirements. They propose MoLIC (Modeling Language for Interaction as Conversation) for representing

interaction. MoLIC is the language which is used for interaction between users and the designer. MoLIC takes views of users in order to assist designer to incorporate them in the design.

Wilde et al. [6] state that there are many interesting questions related to HCI design patterns of pervasive computing systems. In this new area it is difficult to create a HCI design pattern which is also universally accepted. The important question is how to test the interaction patterns to validate time and expenses; and how to assess the process of using them. The HCI patterns are very helpful because they motivate designers of pervasive systems to improve the quality of interaction among the systems. As a result the authors consider that pattern collection is in need and will be more widespread in the future. So the developers of pervasive systems will pass on HCI pattern collection and continue effort to classify the upcoming interaction patterns. [6]

A survey [7] says that HCI is an important factor in software development. Using the concept of work reusability, two researchers, Ward Cunningham and Kent Beck use the work of Christopher Alexander based on the concepts of object-oriented-programming (OOP) and user interface (UI) implementation. In their work they present five patterns for designing window-based user interfaces in Smalltalk. These design patterns become beneficial to the software engineering community when a book "Design Patterns: Elements of Reusable Object-Oriented Software" by Gamma et al. was published. This book becomes one of the bestselling books in software engineering. These researchers with their innovative thoughts have established a series of design patterns and pattern languages in software engineering. A first design pattern of user-centered interface design is published by Coram and its group fellows. Basic intention was to provide high level patterns with which user interface designers could build graphical user interfaces which are pleasurable and productive to use. First Computer Human Interaction (CHI) workshop on pattern languages in user interface design was organized in 1997. The participants focused on the usage of pattern languages in HCI/user interfaces to.

Kruschit and Hitz [7] describe that the relationships between content elements and sharing design knowledge are the most important concepts in reusing the design patterns. Mostly HCI patterns describe a brief relationship of connections to other patterns. A proper relationship is needed to be described among patterns so that design problems are reduced and the patterns communicate with other patterns successfully. The tools used in the patterns are the major assets because they describe the reuse of HCI design patterns. There are many tools which are used such as "online libraries/catalogs, pattern management tools, and pattern-based UI design tools." The interface tools are helpful for user interface designers to understand user requirements instead of gathering it again and again. Design patterns exist at a higher level of concept than UI toolkits and are not dependent on source code for specific implementation.

At a Human Computer Interaction workshop in 2003 the standardization approach was presented [7]. PLML (pronounced "pell mell") is the standard pattern form for HCI patterns proposed by Fincher et al. Its purpose is to provide standard patterns for authors and users to use the patterns. Later on, an improved version of (PLML v 1.2.) is developed by Deng et al. to overcome many problems. Moreover the patterns are written in such a way that they give users a possibility to change the specific widgets according to their choice of solution. [7]

### **3. USER CENTERED DESIGN PATTERNS AND THEIR ISSUES**

#### **3.1 Alexander's Notion of Pattern**

Alexander et al. originates their work from an observation "Most of the wonderful places of the world were not made by architects but by the people [8-a]". Alexander uses patterns as a guide to construction at human levels of scale, from towns and neighborhoods down to houses and rooms. Patterns go hand-in-hand with an underlying process. Alexander relates three properties of patterns that have become staples of software development. First, the process of building with patterns is incremental. User cannot achieve in a single leap. Second, it is grounded deep in human sensibilities, more so than in just engineering practices; and third, the process is iterative. [8-b]

The patterns are like elements of a language, and the constraints define the language. Indeed, the total system of patterns is called a pattern language. The notion of process is fundamental to patterns' place in system evolution. We grow a system one pattern at a time. While patterns are elements of form, it is also useful to think of them as functional transformations on a system that map the system from one state to another. [8-b]

Alexander et al. prefer users rather than technical persons as their consultants. It might be complex to develop an enriched user friendly product, but it would be easy in usage. User will not need any technical skills to work and handle a product. [8]

According to [9] every project has some details and description to construct. By observing that description one can easily set out what user wants and what is the possibility to construct in it.

### 3.2 Jan Borchers

Description of Jan Borchers' pattern is considered a standard. The pattern describes the solution of a problem by analyzing its context of the problem using examples and diagrams [10].

Pattern in Urban Architecture (e.g. Alexander's pattern for construction homes, buildings etc), patterns in Software Engineering (e.g. small talk UI, etc), patterns in HCI (e.g. apple's human interface guidelines etc) and patterns in Application domain (e.g. pattern mapping etc) are different pattern dimensions. [10]

Jan Borchers proposes a pattern language. Each pattern language is acyclic directed graph and it has nodes and edges. Each node represents a pattern and every two nodes form an edge. Each edge is a reference and every reference has some context. So each set has its name, ranking, illustration, forces, problem, solution and diagram [11].

### 3.3 Hybrid

The hybrid pattern is based on Alexander's notions. It has the basic elements of the Alexandrian pattern in an augmented form. Major change is involvement of the latest technologies based on user input. Thus hybrid pattern is a mixture of concepts of Alexander's and the working approach of technical and the user requirements. [12]

Example of this pattern is Toolbox in HTML. Further sub examples are:

- a. Shape palettes in PowerPoint, Illustrator, MacDraw, etc.
- b. Toolbar buttons for font styles and text alignment in Word, Netscape Composer, etc [12].

There are some disadvantages of this pattern too. It provides a single item solution. It is based on rush over technology and is mainly dependent on the developers experience. The pattern describes a single entity description/ solution. The problem addressed is theoretical, and is not supported by diagrammatical construction or through any development language. Practical implementation emergencies are not discussed in the pattern as well. [12]

### 3.4 User Interface Design Patterns

Sari's collection of User Interface Design Patterns explicitly states that *"we have not tried to apply the format of Alexander's design patterns."* The collection *"tries to outline the recurring design problems faced when trying to create good design."* In an opening section each pattern has a description of some characterizing features and an indication of use. This is then followed by several examples, which are screenshots. Finally, there is a reference section.

It is based on Model Driven Architecture (MDA), Unified Modeling Language (UML) and Platform Specific Model (PSM). The concept of UI design pattern can be enhanced by incorporating user requirements. The major innovative idea of this pattern is reusability for upcoming projects [9].

### 3.5 Pattern Language

Although *Hughes* [13] describes his work as a pattern language, he only gives a single example of a pattern form with no larger structure, organizing principle or "grammar". His form has some familiar components such as Trigger, Context, Forces and the unusual "Claims Analysis" which can contain negative consequences for deploying the pattern. This goes against the more common assumption that a pattern is a positive way of resolving a design problem. [13].

### 3.6 A pattern Language for Web Usability

Graham [14] has proposed a pattern language for web usability. He states that *"each pattern is presented using the same layout, semantic structure and typographical conventions. These are very closely based on the structure pioneered by Alexander et al."* An interesting feature of WU (web usability) however, is that it distinguishes different type of patterns depending on their placement within the collection. The types are Abstract, Concrete, and Terminal. *"Patterns being terminal does not mean that design thinking stops with them - merely that the language considers the further design issues as beyond its scope or ambitions."* It is an interesting concept which hints towards the possibility of separate but inter-locking collections. [14].

The main weakness is that the concept of reusability has not been considered. The areas which are not addressed in this pattern are software development, reusability and distributed development environment. [14]

### 3.7 30 card deck / card sorting

30-card deck has been constructed to assist long-term (5-, 10- or 20- year) design incorporating environmental sustainability [15]. Each card has:

- A sensitizing example on the face (they call this an "evocative image"),
- A name (they call this "the title of a concept"),
- A brief description,
- An activity "to assist a design team in considering that particular concept."

They are collected into four suits of "four critical envisioning dimensions": Stakeholders, Time, Values and Pervasiveness. [15].

There are some disadvantages of this pattern as well. The pattern does not mention any criteria of selecting the priority of cards. It does not mention what happens if the cards exceed the limit of deck as well. There is no interlinking between the decks. The main weakness is that the pattern has theoretical resemblance with card sorting and less implementation approach for large products. The pattern does not address interlink between decks as well [15].

## 4. DISCUSSION & CONCLUSION

It is evident from the above discussion that no design pattern provides a comprehensive solution. Alexander did marvelous work in theory; however it does not propose a pattern in concrete form. The 30 card deck pattern lacks interlinking of the decks. The architecture of hybrid pattern can be used to merge multiple patterns.

In OSS development, the main concern is design. Lack of expert designers or almost absence of expert designer results a low standard HCI conformance while developing an OSS product. Nielsen's usability engineering life cycle model is one of the comprehensive solutions to conform the usability in a project. World beat project is one of the implementation of the pattern language.

Various issues exist in HCI oriented design pattern that need to be addressed. Less user's involvement in design, no style guidelines, ignoring experience designers, less focus on essential values and design invariants that can be encoded in software, are some of the examples. The patterns mentioned in Section 3 highlight different issues. For example, Alexander et al. do not propose any fixed solution to project type. The hybrid pattern does not provide a fixed solution of development. And the 30 card deck pattern does not mention any criteria of selecting the priority of cards.

To overcome these issues a new design pattern is needed that should incorporate user requirements. We need a comprehensive practical oriented design pattern which focus on users' needs and expectations. A user centered design pattern is the need of the hour. Some practical and diagrammatical solution need to be presented in the proposed pattern so that with the help of the diagrammatical solution developers follow these steps and develop the project accordingly. End users should be fully involved. Similarly involvement of HCI experts at every phase, from requirement to design phase, is also recommended. Based on the recommendations and observations of HCI experts, developers could follow guidelines to address usability issues.

## 5. REFERENCES

- [1] T.Jenifer., " A Pattern Language for Human-Computer Interface Design.," 17 May 1999. USA: MIT PRESS REALEASE.
- [2] D. D. Lisa Battle. Lisa.B, *Usability Body of Knowledge*, 2010. Publisher: The Usability Professionals' Association (UPA)
- [3] "<http://c2.com/ppr/ui.html#1>. ", accessed on 02 September 2012
- [4] "<http://www.cs.kent.ac.uk/people/staff/saf/patterns/gallery.html>. ", accessed on 02 September 2012
- [5] M. Paula and S. Barbosa, " Bringing interaction specifications to HCI design patterns.," in Workshop Perspectives on HCI Patterns: Concepts and Tools, workshop CHI 2003.
- [6] A. Wilde, P. Bruegger and B. Hirsbrunner, "An Overview of Human-Computer Interaction Patterns in Pervasive Systems.," in IEEE CONFERENCE PUBLICATIONS, International Conference on User Science Engineering (i-USEr) ., University of Fribourg, Switzerland, 2010.
- [7] K.Christian, H.Martin. "The Anatomy of HCI Design Patterns" ,IEEE Computer Society Washington, DC, USA ©2009, Vols. 978-0-7695-3862-4 , no. Proceeding COMPUTATIONWORLD '09 Proceedings of the 2009 Computation World: Future Computing, Service Computation, Cognitive, Adaptive, Content, Patterns. Pages 202-207, November 15 2009.
- [8] J.Borchers, "Teaching HCI Patterns: Experience from Two University Courses, position paper for "Patterns in Practice: A Workshop for UI Designers",," in International Conference

on Human Factors of Computing Systems , Apr. 21–25, 2002.

- [9] B. B. Russell Beale, " Using modelling to put HCI design patterns to work & quot," in presented at (2005) HCI International. 11th International Conference on HumanComputer Interaction,, Las Vegas, 2005.
- [10] M. W. M. Z. ,. B. Christian Remy, "A Pattern Language for Interactive Tabletops in Collaborative Workspaces.," in Proceeding EuroPLoP '10 Proceedings of the 15th European Conference on Pattern Languages of Programs, RWTH Aachen University, 52056 Aachen, Germany., 2010.
- [11] J. Borchers, "A pattern approach to interaction design.," in Proceeding: DIS '00 Proceedings of the 3rd conference on Designing interactive systems: processes, practices, methods, and techniques, Department of Computer Science Darmstadt, University of Technology Alexanderstr. 6, 64283 Darmstadt, Germany..2000.
- [12] M. H. Christian Kruschitz, " Analyzing the HCI Design Pattern Variety.," in 1st Asian Conference on Pattern Languages of Programs (AsianPlop'2010), Department of Informatics-Systems, University of Klagenfurt,Klagenfurt, Austria. , 17th March 2010.
- [13] M. Hughes, "A Pattern Language for User Assistance.," Published by ACM., vol. Volume 14 , no. Issue 1, Pages 27 - 29 , January + February 2007,.
- [14] I. Graham, " A pattern language for web usability. SBC Warburg," Addison-Wesley Professional Copyright: 2003, no. ISBN-10: 0201788888 ISBN-13: 9780201788884 , 2003.
- [15] E. Hutchins., "The Cognitive Consequences of Patterns of Information Flow", Department of Cognitive Science, University of California, San Diego," Intellectica., vol. 1:30, pp. 53-74.