

**PATTERN-ORIENTED DESIGN FOR
INTERACTIVE SYSTEMS**

HOMA JAVAHERY

A THESIS

IN

THE DEPARTMENT

OF

COMPUTER SCIENCE

PRESENTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF MASTER OF COMPUTER SCIENCE AT
CONCORDIA UNIVERSITY
MONTREAL, QUEBEC, CANADA

SEPTEMBER 2003

© HOMA JAVAHERY, 2003

National Library
of Canada

Bibliothèque nationale
du Canada

Acquisitions and
Bibliographic Services

Acquisitons et
services bibliographiques

395 Wellington Street
Ottawa ON K1A 0N4
Canada

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file *Votre référence*

ISBN: 0-612-83903-6

Our file *Notre référence*

ISBN: 0-612-83903-6

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

Canada

ABSTRACT

Pattern-Oriented Design for Interactive Systems

Homa Javaheery

Patterns are a medium created to capture and disseminate design knowledge and are used extensively in the software engineering community. Human-Computer Interaction (HCI) Patterns focus explicitly on (1) providing design solutions to any problems relating to interactive systems and their users, and (2) aim to help developers with the design of more usable systems. We will address pattern use from a practical standpoint, as a working part of design, and investigate their applicability in different contexts of use. First, the evolutionary use of patterns will be traced from single pattern use to Pattern-Oriented Design. Secondly, the applicability of patterns in redesigning existing systems will be discussed with a practical example of a Bioinformatics web-based system. We will focus on how usability issues in the existing system drove the choice of patterns used in redesigning the site. Finally, we will illustrate the use of patterns in redesigning the user interface of existing systems to different platforms, such as mobile phones and Pocket PCs. This design domain has a number of constraints, such as screen size and image resolution. Design strategies need to be rethought to accommodate the challenges associated with such devices, and we will suggest some new design ideas and patterns.

ACKNOWLEDGEMENTS

The help and support of a number of people made this thesis a reality, and I am grateful to each of them:

Dr. Ahmed Seffah for his guidance and patience throughout my Masters. He gave me the opportunity to participate in a number of challenging and interesting projects as a member of the HCSE (Human-Centered Software Engineering) Group at Concordia University.

FCAR (Le Fonds Québécois de la Recherche sur la Nature et les Technologies) for their financial support of this research project.

Daniel Sinnig, Dr. Peter Forbrig and Daniel Engelberg for their contribution to the *Multiple User Interfaces* effort.

Jovan Strika, who spent countless hours of brainstorming with me on the topic of patterns, and gave me a different perspective when I tended to get off track.

Rozita Naghshin for her input and feedback, in addition to her advice on image creation.

Debbie Maret, Behzad Hadian and Kiana Karimi for their encouragement.

Above all, I am grateful to my family. I thank my parents and my sister, Sima, for their active support, for believing in me, and for putting up with me!

TABLE OF CONTENTS

List of Figures	vii
List of Tables.....	viii
1. Introduction	1
1.1. Overview of User-Centered Design	1
1.2. Overview of Patterns.....	2
1.3. Objective and Scope of Thesis.....	4
2. Tracing the Evolution of Patterns	7
2.1. Origin of Patterns	7
2.2. Definition of HCI Patterns	9
2.3. HCI Patterns as an Alternative Design Tool to Guidelines.....	10
2.4. From HCI Patterns to Pattern Languages.....	12
2.5. Pattern Language for Web Design	16
2.6. MOUDIL: The Montreal Online Usability Digital Library	19
3. HCI Patterns in the User-Centered Design Lifecycle	22
3.1. User-Centered Design Philosophy	22
3.2. User-Centered Design Lifecycle	24
3.3. Patterns in User-Centered Design	29
3.4. Pattern-Oriented Design.....	30
3.5. UPADE Editor: A Tool for Pattern-Oriented Design	36
4. Redesign of a Bioinformatics System with Patterns: Incorporating User Experiences	38
4.1. Background to Bioinformatics Systems.....	38
4.2. Framework used in Redesigning a Bioinformatics IBIS.....	40
4.3. Modeling the Bioinformatician's Behavior through Personae.....	42
4.4. Heuristic and Psychometric Evaluation	45
4.5. Method	48
4.6. Results.....	49
4.6.1 User Characteristics and Behavior	49
4.6.2 Heuristic Evaluation with UI Experts	51
4.6.3 Psychometric User Evaluation	52

4.7. Redesign: Applying Patterns Based on Results	54
5. Migrating User Interfaces across Platforms Using Patterns.....	62
5.1. Overview of Multiple User Interfaces.....	63
5.2. Motivations for using HCI Patterns with MUIs.....	67
5.3. Redesigning User Interfaces with Pattern Mapping.....	69
5.4. The Effect of Screen Size on Redesign.....	70
5.5. Pattern-based Redesign: A Case Study with Navigation Patterns	72
5.6. Patterns in Reengineering User Interfaces	77
5.7. Pattern-Assisted Reengineering	78
5.8. Comparing Reengineering to Redesign	81
5.9. Research Directions	82
6. Conclusion and Future Investigations.....	86
References	89
Appendix A: Pattern Examples from UPADE Web Language.....	93
Appendix B: NCBI Site User Evaluation.....	95
Appendix C: Mapping of Questions to Heuristics	100
Appendix D: Complete Expert Results for NCBI Evaluation	102

LIST OF FIGURES

Figure 1-1: Example of an Alexandrian pattern.....	3
Figure 2-2: Descriptive Notation of the Convenient Toolbar Pattern.....	17
Figure 2-3: An Overview of the UPADE Web Language	19
Figure 3-1: ISO 13 407 Standard	25
Figure 3-2: Human-Centered System Development as per ISO 15504	28
Figure 3-3: The Composite pattern	32
Figure 3-5: The Executive Summary Pattern.....	34
Figure 3-6: The UPADE Editor	37
Figure 4-1: Usability Evaluation and Persona in HCI Pattern Selection	41
Figure 4-2: Questionnaire Results of Novice and Expert Users	53
Figure 4-3: Comparing Novice and Expert Users – Unsatisfied and Unsure	54
Figure 4-4: Current Home page of the NCBI site	56
Figure 4-5: Outline of Redesigned NCBI Home page	58
Figure 4-6: Redesigned Home page of NCBI Site.....	59
Figure 4-7: Redesigned Site Map of NCBI Site.....	60
Figure 5-1: An example of a MUI.....	64
Figure 5-2: HCI Patterns in a MUI Framework	66
Figure 5-3: Redesigning the UI with Pattern Mapping.....	70
Figure 5-4: Patterns Extracted from the CBC News site	74
Figure 5-5: Migration of the CBC site to a PDA Platform using Pattern Mapping.....	76
Figure 5-6: Pattern-Assisted UI Reengineering	78

LIST OF TABLES

Table 2-1: Example of a Design and Organizational Pattern.....	7
Table 2-2: Guideline versus Pattern for the Toolbar.....	12
Table 4-1: A Set of Personae for NCBI Site Users	44
Table 4-2: Definitions of Heuristics.....	46
Table 4-3: Task Use and Interaction Behavior of NCBI Site Users	50
Table 4-4: Enhanced Personae of NCBI Site Users.....	51
Table 5-1: A Description of the Quick Access Pattern	69
Table 5-2: Screen Size of PDAs and Desktops	70
Table 5-3: Examples of HCI patterns.....	73
Table 5-4: Examples of HCI Pattern Transformations for Different Screen Sizes.....	75

1. Introduction

1.1. Overview of User-Centered Design

Interactive systems are increasingly entwined in our daily lives. Whether we are dealing with conventional desktop applications, web-based wireless interfaces, or even wearable-device solutions, it is apparent that a plethora of new challenges exist when designing interactive systems. What is really daunting is that the complexity and diversity of the design process continues to grow far more rapidly than we are able to cope. How should we design interactive systems? More specifically, how can we design the user interface of these systems in a way that makes them both usable and useful? [Erickson 2000]

User-centered design (UCD), as a software development philosophy, focuses specifically on making products usable. The usability of a product is defined in ISO 9241, part 11 as: “The extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use.” [ISO 9241, 1991]. This definition relates to the quality of the interaction between the person who uses the product to achieve actual work and the product or software application itself. The important features of this interaction are:

- Effectiveness – how well the user accomplishes the goals they set out to achieve using the system
- Efficiency – the resources consumed in order to achieve their goals
- Satisfaction – how the user feels about their use of the system

Developing software with the quality of this interaction in mind is the primary goal of UCD. The approach typically entails involving users in the design of the system so that their feedback can be obtained. Prototypes are usually employed to do this and designs are modified in light of user feedback. Following this process, the developed software can result in a number of significant advantages for the software developer, by producing software which [ISO 13407, 1998]:

- Is easier to understand and use, thus reducing training costs
- Improves the quality of life of users by reducing stress and improving satisfaction
- Significantly improves the productivity and operational efficiency of individual users and consequently, the organization

Initially it may seem that the user-centered approach complicates the software development task due to the need to make iterative refinements to the design of the software in light of user feedback. However, the benefits to be gained are considerable. In addition to the advantages listed above, the process promotes communication between users, managers and those developing the software. It also identifies problematic issues early on in the development process when it is much cheaper to implement changes. For these reasons alone, it is worth adopting a user-centered perspective. The question is how can we adopt such a user-centered approach to design? In this thesis, I will demonstrate how patterns can help us design interactive systems with a user-centered approach.

1.2. Overview of Patterns

The architect, Christopher Alexander, introduced the idea of patterns in the 1970s [Alexander et al. 1977]. His idea stemmed from the premise that there was something fundamentally incorrect with the approach taken by twentieth century architectural design methods and practices. He introduced patterns as a three-part rule to help architects and engineers with the design of buildings, towns, and other urban entities. His definition of a pattern was as follows: “Each pattern is a three-part rule, which expresses a relation between a certain context, a problem, and a solution” [Alexander 1979]. The underlying objective of Alexander’s patterns was to tackle architectural-related problems that occurred over and over again in a particular environment, by providing commonly accepted solutions. Figure 1-1 illustrates an example of one of his patterns adapted from Erickson [2000]. The numbers in brackets are identifiers for the patterns.

Street Café (88)

[picture omitted]

...neighborhoods are defined by Identifiable Neighborhood (14); their natural points of focus are given by Activity Nodes (30) and Small Public Squares (61). This pattern, and the ones which follow it, give the neighborhood and its points of focus, their identity.

The street cafe provides a unique setting, special to cities: a place where people can sit lazily, legitimately, be on view, and watch the world go by.

The most humane cities are always full of street cafes. Let us try to understand the experience which makes these places so attractive. We know that people enjoy mixing in public, in parks, squares, along promenades and avenues, in street cafes. The preconditions seem to be: the setting gives you the right to be there, by custom; there are a few things to do that are part of the scene, almost ritual: reading the newspaper, strolling, nursing a beer, playing catch; and people feel safe enough to relax, nod at each other, perhaps even meet. A good cafe terrace meets these conditions. But it has in addition, special qualities of its own: a person may sit there for...

[nine paragraphs of rationale omitted]

Therefore:

Encourage local cafes to spring up in each neighborhood. Make them intimate places, with several rooms, open to a busy path, where people can sit with coffee or a drink and watch the world go by. Build the front of the cafe so that a set of tables stretch out of the cafe, right into the street.

[diagram omitted]

Build a wide, substantial opening between the terrace and indoors-OPENING TO THE STREET (165); make the terrace double as A PLACE TO WAIT (150) for nearby bus stops and offices; both indoors and on the terrace use a great variety of different kinds of chairs and tables-DIFFERENT CHAIRS (251); and give the terrace some low definition at the street edge if it is in danger of being interrupted by street action-STAIR SEATS (125), SITTING WALL (243), perhaps a CANVAS ROOF (244).

[text omitted]...

Figure 1-1: Example of an Alexandrian pattern

Alexander's patterns are written in narrative form. Even if they do not have clearly defined attributes per se, they are all structured in a specific way with a description of the problem, solution, and context. The idea of using Alexandrian-type patterns as a design tool has been quite influential in a variety of domains in the last decade, including software engineering. In recent years, the Human-Computer Interaction (HCI) community has jumped on the bandwagon and adopted the idea of patterns for interactive system design.

Patterns in HCI have been introduced as a tool to capture and disseminate **proven** design knowledge, and to facilitate the design of more **usable** systems. Patterns aim to capture

and communicate the best practices of user interface design with a focus on the user's experience and the context of use. As a result, they are an attractive tool for UCD, with interesting ramifications for designing across a variety of contexts.

1.3. Objective and Scope of Thesis

The starting point of my research was to analyze the use of patterns in interactive system design, and to see if from a practical point, they can be used for design in different contexts of use. In addition, to propose different ways in which patterns can be applied in the design process. Patterns have been vigorously discussed in the HCI community, and there has been a substantial effort in the last few years to define them (various conference workshops such as CHI 2002 and 2003; Interact 2001 and 2003). Rather than defining patterns, my thesis addresses pattern use from a practical standpoint, as a working part of design, and investigates their applicability in different contexts of use.

First, I will discuss how patterns can be combined to design user interfaces for new systems, and I will introduce the concept of pattern-oriented design (POD). I will demonstrate how a specific pattern language (UPADE) can be used to design a web application. Even though my illustration of POD is with relation to internet-based sites and systems, most of the ideas can also be extrapolated to GUI-type applications.

Secondly, I will discuss how patterns can be used to redesign the user interfaces of existing systems. Interactive systems are not static. Over time, changes become necessary either due to modifications in user needs or business requirements. In addition, redesign can be initiated due to usability problems of the original system. Using a case study, I will illustrate pattern use in redesigning the National Center for Biotechnology (NCBI) site, which has a very specific user group. I will focus on how usability issues in the existing system drove the choice of patterns used in redesigning the site.

Thirdly, I will illustrate the use of patterns in redesigning the user interface of existing systems to different platforms, such as mobile phones and Pocket PCs. This design

domain has a number of constraints, such as screen size and image resolution. Design strategies need to be rethought to accommodate the challenges associated with such devices, and I will suggest some new design ideas and patterns.

In summary, the major objectives of my thesis are to investigate pattern-oriented design and see if patterns are mature and comprehensive enough to be used in the different contexts outlined above. My thesis will, in large part, deal with web-based patterns and applications for:

- Design of a new application using Pattern-Oriented Design
- Redesign of an existing application using empirical analysis and HCI patterns
- Redesign of UIs to different platforms by transforming existing patterns

The organization of the thesis is as follows:

Chapter 2 provides a review of patterns, and describes their relevance in interactive system design. The evolution of patterns from single pattern use to pattern languages is traced. In addition, the UPADE Web language will be described.

Chapter 3 discusses patterns in the user-centered design lifecycle, and describes the systematic approach of Pattern-Oriented Design for designing new web applications.

Chapter 4 presents the framework used to redesign an existing bioinformatics web-based system. Personae, heuristic evaluation, and psychometric assessment through questionnaires were used to receive feedback about usability issues with the site. Based on these results, appropriate patterns were used to redesign the site.

Chapter 5 describes pattern use in migrating existing systems to different platforms, or Multiple User Interfaces. Existing patterns from the HCI community were transformed to patterns appropriate for other devices; such as mobile phones and Pocket PCs.

Investigations were mainly surrounding web applications, and a case study where a News site was redesigned for use on a handheld device is discussed.

Finally, **Chapter 6** summarizes my work and presents future avenues for research in this area.

2. Tracing the Evolution of Patterns

2.1. Origin of Patterns

Patterns have been around since the 1970s [Alexander et al. 1977] thanks to Alexander's work. His pattern framework has been applied extensively to object-oriented programming, and inspired a different way of thinking in which design knowledge is captured and reused effectively. Alexander's influence is apparent in Gamma et al.'s [1995] book, "Design Patterns: Elements of Reusable Object-Oriented Software". This book inspired the software engineering community to take a closer look at the concept of patterns as a problem-solving discipline for object-oriented design. Gamma et al. [1995] have documented 23 design patterns in their book, one example being the Observer Pattern. Like all other patterns, the observer pattern is described in a specific format, with consistent attributes. A short description of this pattern is given in Table 2-1.

Table 2-1: Example of a Design and Organizational Pattern

DESIGN PATTERN [Gamma et al. 1995]	ORGANIZATIONAL PATTERN [Coplien 1995]
<p>Name: Observer Intent: Define a one-to-many dependency between objects. When one object changes state, all its dependents are notified. Applicability: (1) A change to one object requires changing other unknown objects, (2) Object should be able to notify other objects, but you don't want them to be tightly coupled Participants: Classes are Subject, Observer, Concrete_Subject, and Concrete_Observer Consequences: (1) Abstract for broadcast communication, (2) Support for broadcast communication, (3) Unexpected updates Related Patterns: Mediator, Singleton</p>	<p>Name: Review The Architecture Problem: Blind Spots occur in the architecture and design Context: A software artifact whose quality is to be assessed for improvement Forces: (1) A shared architectural vision is important, (2) Even low-level design and implementation decisions matter, (3) Individual architects and designers can develop tunnel vision. Solution: All architects should review all architectural decisions. Architects should review each other's code. The reviews should be frequent and informal early in the project. Resulting Context: The intent of this pattern is to increase coupling between those with a stake in the architecture and implementation, which solves the stated problem indirectly. Related Patterns: Mercenary Analyst, CodeOwnership</p>

What is notable about design patterns is that they are both concrete and abstract at the same time. They are concrete enough to provide sound solutions to design problems, which can be put immediately into practice. On the other hand, they are abstract enough to be applied to different situations [Li 2001]. This new way of thinking, in which there is far less inclination towards technology, and a greater focus on documenting and disseminating best practices, has prompted other software-related domains to adopt the idea of patterns. These domains are as diverse as developmental organization and process, teaching, and software architecture. For comparative purposes, Table 2-1 also illustrates an organizational pattern called *ReviewTheArchitecture*, by Coplien [1995]. Organizational patterns are also documented in a specific format, with consistent attributes. Although these attributes may differ from those used to describe design patterns, the principle is similar.

In software engineering, patterns exist at different levels. They can be related to either the organization as a whole, the process of software development, the people involved, or the product being engineered. Figure 2-1 is illustrative of the *Pattern Mania* that presently exists in the world of software engineering. As you can see, the idea of patterns is no longer restricted to object-oriented design or organizational practices.

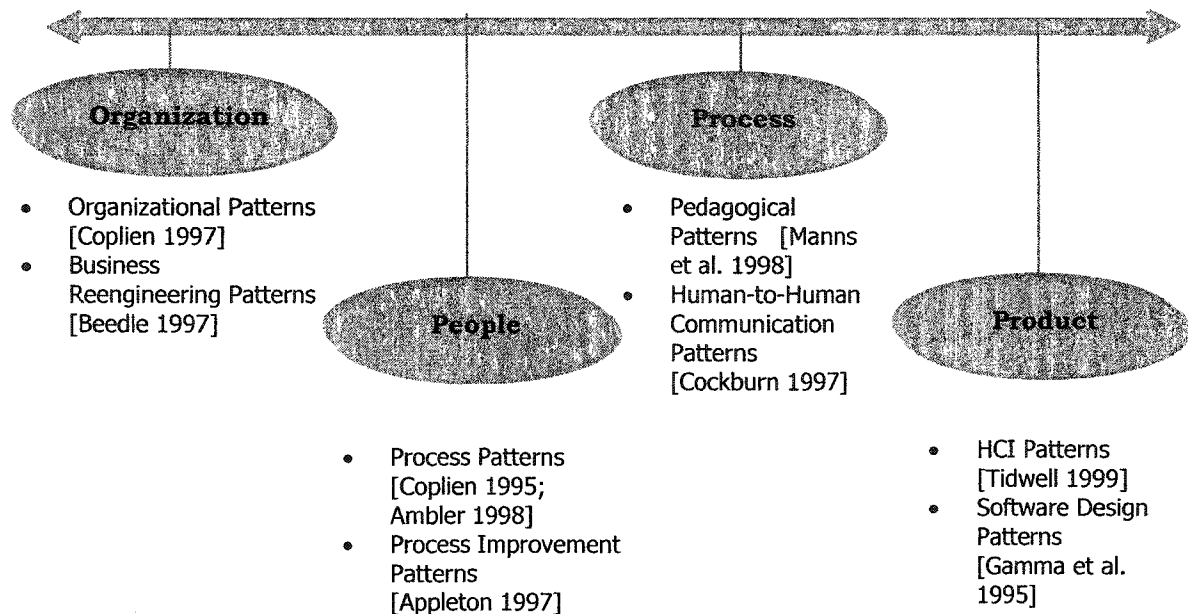


Figure 2-1: Pattern Mania in Software Engineering

2.2. Definition of HCI Patterns

To this end, during the last three years, the HCI community has been a forum for vigorous discussion on patterns for HCI and interactive system design. Although there may be a number of differing terminologies, for the duration of this thesis, I will call these patterns HCI Patterns. HCI Patterns focus explicitly on providing solutions to any problems relating to interactive systems and their users. They help with the design of more usable systems; by providing well-known solutions to UI and usability-specific problems, and capturing best user experiences. They are applicable to different levels of abstraction such as the user-task model, the navigation model or the concrete presentation of the user interface. HCI patterns are extremely useful for designers because they have the potential to drive the UI design process [Borchers 2000; Lafreniere 1999; Javahery and Seffah 2002].

One important remark that I would like to make at the forefront of this chapter is that patterns are a great source of interest not necessarily because they provide novel ideas to the software engineering community, but because of the way that they package already-available design **knowledge**. This way of presenting information to software designers and developers allows the reuse of best practices, and avoids reinventing the wheel each time. In addition, HCI patterns are a great way of incorporating usability best practices into software development. In light of this, pattern use has not just gained popularity amongst software engineers, but is of great interest to usability engineers and specialists who are concerned with the construction of usable systems.

In *Notes* [Alexander 1964], Alexander describes the reasons surrounding the failure of existing design practices. One argument he makes is that traditional design practices fail to create products that meet the real needs of the user, and are ultimately inadequate in improving the human condition. His patterns were introduced in a hierarchical collection with the purpose of making buildings and urban entities more usable and pleasing for their inhabitants. Interestingly enough, this idea can be extrapolated to HCI design, where the primary goal is to make interactive systems that are usable and pleasing to users.

2.3. HCI Patterns as an Alternative Design Tool to Guidelines


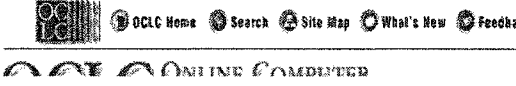
Before HCI patterns became so popular, existing resources for UI designers and usability specialists were scarce. Besides various databases and repositories of information, the main resources available to designers were **Guidelines**. Like patterns, guidelines aim to provide information about the design of interactive systems and to disseminate, or distribute, this information. They concentrate most often on the physical design attributes of the user interface. This information is important for both novice and experienced designers, allowing them to have a point of reference for design practices. Good examples are the *Macintosh Human Interface Guidelines* and the *Java Look and Feel Design Guidelines* (<http://java.sun.com>).

Although guidelines are useful, they have a number of disadvantages. First, they are numerous, and it is therefore difficult to select the appropriate guideline to apply to the design problem in question. This is especially true for novice designers. Secondly, at times, guidelines contradict each other. One guideline may indicate one approach, while another guideline is indicative of a different approach. Consequently, the designer may have to figure out the best solution by guessing, and in some cases, the design problem will not be solved. Thirdly, and most importantly, most guidelines give general information, and are not problem-oriented. In cases where guidelines are problem-oriented, they do not promote reuse because they are too tailored to a particular toolkit or technology [Borchers 2001].

Patterns alleviate many of the shortcomings associated with guidelines. Above all, they are a good alternative to guidelines because they are problem-oriented, but not toolkit-specific. They are more concrete and easier to use for novice designers. Guidelines can be quite abstract, whereas patterns are more structured and the knowledge is placed in a context. The designer is told when, how and why the solution can be applied. Since patterns are context-oriented, the solution is related to a specific activity. Furthermore, patterns promote reusability of design solutions for two reasons: (1) they are toolkit independent, and (2) they are presented in a specific format with defined attributes.

Table 2-2 compares a guideline and a pattern that addresses the same problem: Helping the user to find frequently used commands or pages. What is interesting to note is that the pattern version of the description gives very detailed information about the context in which the solution can be applied. If the context changes, the pattern changes. In addition, to describe a HCI pattern, a specific format is used. First, each pattern is given a distinct name. Secondly, the actual definition of the pattern is structured into distinct attributes, which depend on the pattern language being used. As an example, some attributes used in our UPADE Web Pattern Language, are: Context, Problem, Solution, and Example [Javahery and Seffah 2002].

Table 2-2: Guideline versus Pattern for the Toolbar

Guideline	Pattern
<p>A toolbar is a collection of frequently used commands or options that appear as a row of toolbar buttons. Toolbars normally appear horizontally beneath a primary window's menu bar, but they can be dragged anywhere in the window or into their own window. Toolbars typically contain buttons, but you can provide other components (such as text fields and combo boxes) as well. Toolbar buttons can contain menu indicators, which denote the presence of a menu. Toolbars are provided as shortcuts to features available elsewhere in the application, often in the menus.</p>  <p style="text-align: right;">Toolbar</p> <p>Source: Java Look and Feel Design Guidelines: http://java.sun.com/products/jlf/ed2/book/</p>	<p>Pattern Name: Convenient Toolbar Pattern Description:</p> <ul style="list-style-type: none"> • Context – assist the user to reach convenient and key web pages at any time <ul style="list-style-type: none"> ○ User: Novice and expert ○ Workplace: Website, using desktop browser • Problem – help the user find useful and “safe” pages that need to be accessed from any location on the site, regardless of the current state of the artefact • Solution – group the most convenient action links, such as home, site map, and help • Other attributes – Forces, Related Patterns, ... • Specific example – 

2.4. From HCI Patterns to Pattern Languages

So far, I have discussed how patterns can be used to capture and encapsulate design knowledge. There are, however, two important pieces to the puzzle: Capturing the knowledge, and the **dissemination** of the knowledge. Without the latter, there is not much use in collecting patterns in the first place.

Pattern **languages** aim to disseminate the knowledge contained in patterns. They are a collection of interrelated patterns, and can be used as a lingua franca for design [Erickson 2000]. There exist two essential criteria that have to be fulfilled for a pattern language. Firstly, the language has to contain a standard pattern definition. One format for defining patterns was described in the previous section. Secondly, a relationship must exist

between patterns. In other words, the language must depict interrelationships between different patterns, as well as logically group patterns.

Successful designs require individuals to communicate their concepts and ideas, and to build a common forum for the discussion of already-available design practices. As in any culture or society, the HCI community needs a common ground for such communication and dissemination of knowledge. Thomas Erickson [2000] proposes using pattern languages as a *Lingua Franca* for the design of interactive systems. He discusses the potential of a lingua franca as a way of making communication in design a more “egalitarian process”. There are many stakeholders, including users, who may be part of the design process. It is important to have a common language that all stakeholders can use to communicate points or ideas about design. A lingua franca can act as a communicative resource to facilitate discussion, presentation, and negotiation for the many different individuals who play a role in designing interactive systems.

More importantly than being a communicative tool, pattern languages guide software designers through the design process, by providing solutions to a set of common design problems. In addition, they can provide a framework for designers to connect patterns into complete architectures for interactive systems.

Many groups have devoted themselves to the development of HCI pattern languages. Welie’s Interaction Design Patterns [Welie 2003], Experiences [Coram and Lee 1998], and Tidwell’s UI Patterns and Techniques [Tidwell 2002] play a major role in the HCI field and wield significant influence. In addition, Jan Borchers’ book entitled, “A Pattern Approach to Interaction Design” [Borchers 2001] introduces a pattern language for the specific domain of interactive music exhibits.

As an example, the *Experiences* pattern language, developed by Coram and Lee [1998] concentrates on the user’s experience within software systems. The main focus is on the interactions between the user and the interfaces of software applications. Patterns are grouped according to different focus areas and user interface paths such as interaction

style, explorable interface, and symbols. The interrelationships between the patterns are clearly mapped, and each pattern has a pattern description in natural language.

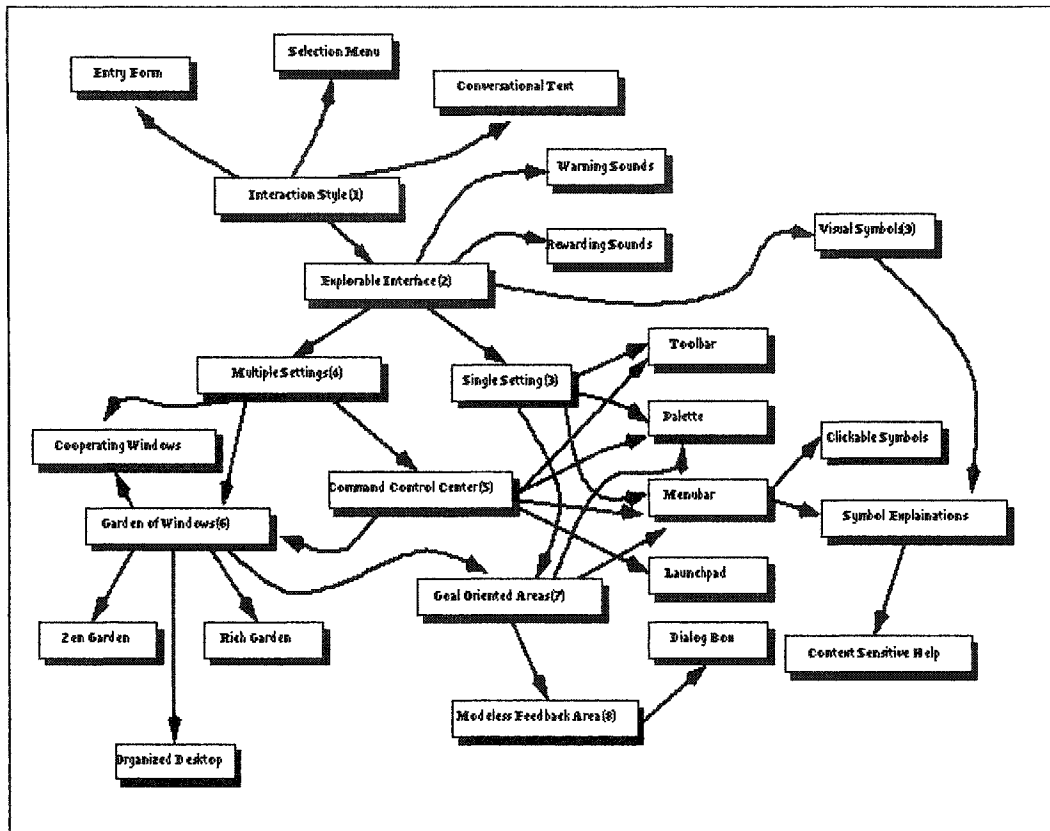


Figure 2-2: The Experiences Pattern Language

With pattern languages, the concept of a “relationship” between patterns brings us one step closer to being able to use patterns effectively and with efficacy to solve problems in HCI and interactive system design. Pattern languages can include a method to help connect patterns to design whole architectures for a particular domain. *Experiences* showed the beginnings of thinking along these terms, but regrettably was not developed in its entirety. In chapter 3, I will go one step further and discuss how to link patterns to the user-centered design (UCD) lifecycle, and will demonstrate how to connect patterns to design an entire architecture for a web-based application.

There are certain problems associated with current pattern languages. To begin with, there are no standards for the documentation of patterns. The Human-Computer

Interaction community has no uniformly accepted pattern form. Current pattern languages resort to narrative text formats, which is the case for the *Experiences* example described above. Moreover, both the Welie and Tidwell collections describe their patterns in narrative text, although they do organize their descriptive format with certain attributes such as: *Use when, Why, How, Examples*.

Secondly, many pattern languages do not have clearly defined relationships between their patterns. This is, by far, the most serious drawback of current languages. In the case where interrelationships or groupings are described, they are often incomplete and not context-oriented. This is a major limitation since the conditions underlying the use of a pattern is related to its context of use. Since patterns deal with different levels of abstraction within design, if languages are not structured logically, it can be confusing for individuals trying to work with them. *Experiences* describes interrelationships between some of its patterns, and a few languages categorize patterns into groups (such as the Welie and Tidwell collections), but none of them address both issues so as to effectively capture the relationship between individual patterns. A taxonomical classification is definitely missing in the HCI pattern community.

Thirdly, when patterns are documented, there are no tools to formally validate them. There should be some formal reasoning and method behind the creation of patterns, and in turn, pattern languages. One way would be to tie in patterns to a process, which would highlight their need more precisely. As mentioned previously, *Experiences* showed the beginnings of thinking along these terms, and in the next chapter, I will go one step further and discuss how to link patterns to the UCD lifecycle.

Finally, a language in computer science is described as having some sort of syntax and semantic. None of the current pattern languages follow this principle, and there are no universally applicable rules in the HCI community with regards to how to document patterns and structure pattern languages. This has implications with regards to how usable current pattern languages are for designers [Seffah and Javahery 2002]. Such issues are currently being explored and tackled by various HCI conference workshops every year.

2.5. Pattern Language for Web Design

The problems with current pattern languages motivated us to develop the UPADE Web Language. UPADE is an acronym for Usability Patterns-Assisted Design Environment, and is an ongoing research project in the Human-Centered Software Engineering (HCSE) Group dealing with HCI-related pattern use and supporting tools for design. The UPADE web language and its environment is one such tool to assist software developers with the design of internet-based information systems.

Each pattern in the UPADE web language provides a proven solution for a common usability and HCI-related problem occurring in a specific context of use for web applications. The language follows certain rules, with regards to pattern documentation and language structure. An example of the descriptive notation (consisting of nine attributes) used for each pattern is given in Figure 2-2, using the Convenient Toolbar pattern which was introduced in Section 2.3.

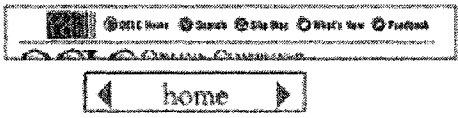
Pattern Name: Convenient Toolbar Pattern Type: Navigation Support	
Context Use:	User: Novice or Expert Task: Assist the user to reach convenient and key web pages at anytime
Workplace:	Web applications
Usability Problem:	The user can easily find useful and "safe" pages regardless of the current state of the artefact. The user can reach these pages promptly.
Usability Factor:	Factor: Efficiency, Safety Criteria: Consistency, Minimal Action, Minimal Memory, User Guidance, Helpfulness
Example:	
Design Solution:	Group the most convenient action links, such as home, site map, help and etc. Use meaningful metaphors and accurate phrases as labels. Place them consistently throughout the web site.
Other Language Attribute:	Design Principle Related Usability Patterns Reading

Figure 2-2: Descriptive Notation of the Convenient Toolbar Pattern

The UPADE web language is comprised of a set of HCI patterns, grouped into three categories:

- Architectural patterns
- Structural patterns
- Navigation support patterns

In addition to the above categories, individual patterns have defined interrelationships between them: Superordinate, subordinate, competitor, and neighboring [Li 2001]. These will be described in detail in the context of Pattern-Oriented Design (Section 3.4).

Architectural patterns describe different architectures for organizing a web site's entire contents. This group of patterns describes different schemes for organizing the content of a web application across separate pages and can even extend across multiple servers.

Structural patterns define the layout of content suitable for presentation on the web, with the aim of establishing a consistent physical and logical screen layout of pages. These patterns can be further grouped into two different categories: Page Managers and Information Containers. The former cover elements such as the home page and utility pages (ex: search and help). The latter suggest methods for displaying and grouping content into cognitively accessible segments of information.

Navigation support patterns portray techniques for browsing and navigating within a set of information segments, as well as between interrelated pages. Many of these patterns are not necessarily new, and have been defined in other pattern languages [Welie 2003; Tidwell 2002].

The figure below illustrates the UPADE web language and all of its patterns.

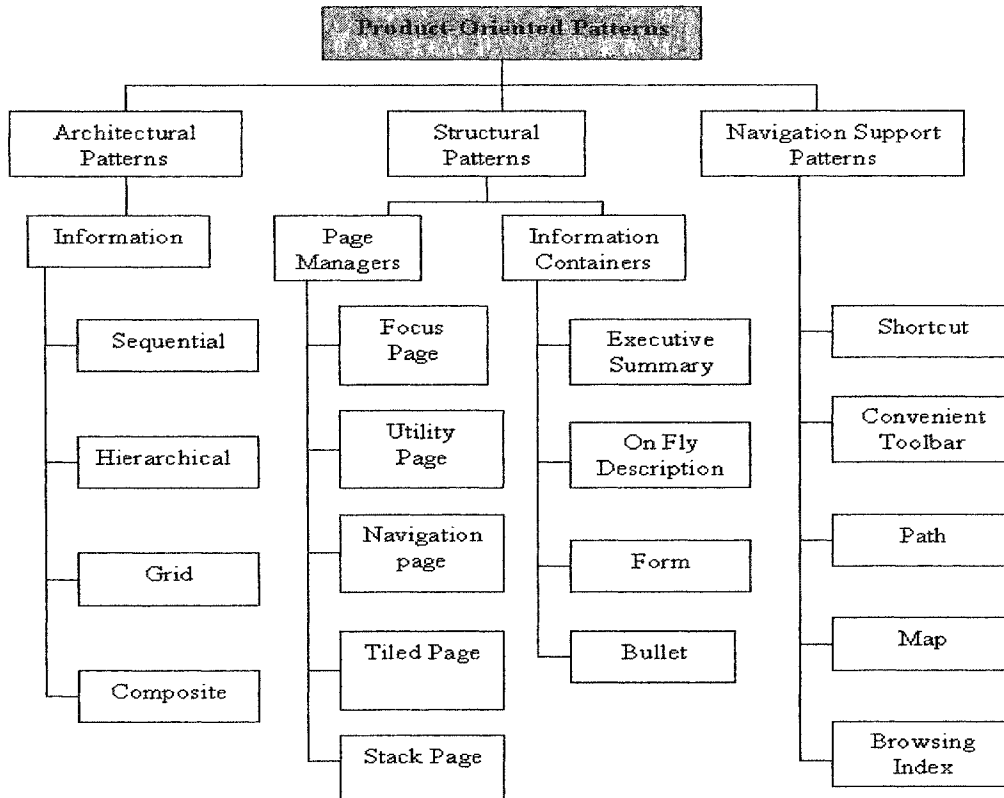


Figure 2-3: An Overview of the UPADE Web Language

2.6. MOUDIL: The Montreal Online Usability Digital Library

In order to improve the UPADE web language and to better understand pattern use for the UPADE project, we decided to develop an online digital library for HCI patterns. MOUDIL, the Montreal Online Usability Digital Library, is targeted for software developers and usability engineers. MOUDIL was designed with two major objectives: First, as a service to UI designers and software engineers to share HCI pattern information for UI development. Secondly, as a research forum for understanding how patterns are really discovered, validated, used and perceived [Gaffar et al. 2003]. It is hoped that all this information can be helpful in further developing our understanding of patterns and can provide feedback for the UPADE project.

MOUDIL is still being developed. Some of its key characteristics are [Gaffar et al. 2003]:

- It has been designed to accept proposed or potential patterns in many different formats or notations. Therefore patterns in versatile formats can be submitted for reviewing. Although originally the pattern database included only our UPADE web language, it is hoped that other patterns and pattern languages will be included.
- There will be an international editorial board for reviewing and validating patterns. Before publishing, collected and contributed patterns must be accessed and acknowledged by the editorial committee. We are inviting HCI pattern practitioners and researchers to join this committee.
- A Pattern Ontology editor will capture the HCI community's understanding of pattern concepts and relate them to one another. A pattern taxonomy is a first step, and as mentioned in section 2.4, such a taxonomy is missing in the HCI community.
- A Pattern Editor will allow semantic information to be attached to patterns. Based on this information and our ontology, patterns will be placed in relationships, grouped, categorized and displayed.
- A Pattern Navigator will provide different ways to navigate through patterns or to locate a specific pattern. The pattern catalogue can be browsed by pattern group or searched by keyword. Moreover, a pattern wizard will find particular patterns by questioning the user.
- A Pattern Viewer provides different views of the pattern, adjusted to the preferences of the specific user.

So far, I have traced the evolution of patterns in the HCI community, and discussed pattern languages in detail. In addition, I have outlined the UPADE web language and an

additional pattern-based tool called MOUDIL. I will now turn my focus on how we can go one step further with patterns: How to combine them for design.

3. HCI Patterns in the User-Centered Design Lifecycle

3.1. User-Centered Design Philosophy

Simply stated, UCD philosophy entails involving users in design so that their feedback can be obtained, therefore resulting in more usable interactive systems. In this section, I will describe some general principles to clarify the UCD philosophy. The next section will detail the UCD lifecycle and associated standards.

The main principle of UCD, and its key strength, is the active involvement of users [ISO 13407, 1998]. The extent of this involvement depends on the precise nature of the design activities, but generally speaking, the strategy is to involve individuals who have real insight into the context in which an application will be used. Involving such end-users can also enhance the acceptance and commitment to the new software. Users should feel that the system is being designed in consultation with them, rather than being imposed on them.

IBM's Dr. Clare-Marie Karat, a researcher at its Thomas J. Watson Research Center, has proposed a new set of user rights "to transform the culture, in which information technology systems are designed, developed and manufactured," and to ensure that all future products are precisely what the customer expects. This set of user rights is as follows [Karat 1998]:

1. **Perspective:** The user is always right. If there is a problem with the use of the system, the system is the problem, not the user.
2. **Installation:** The user has the right to easily install and uninstall software and hardware systems without negative consequences.
3. **Compliance:** The user has the right to a system that performs exactly as promised.
4. **Instruction:** The user has the right to easy-to-use instructions (such as user guides, online or contextual help, and error messages) for understanding and utilizing a

system to achieve desired goals and recover efficiently and gracefully from problem situations.

5. Control: The user has the right to be in control of the system and to be able to get the system to respond to a request for attention.
6. Feedback: The user has the right to a system that provides clear, understandable, and accurate information regarding the task it is performing and the progress towards completion.
7. Dependencies: The user has the right to be clearly informed about all system requirements for successfully using software or hardware.
8. Scope: The user has the right to know the limits of the system's capabilities.
9. Assistance: The user has the right to communicate with the technology provider and receive a thoughtful and helpful response when raising concerns.
10. Naturalness: The user should be the master of software and hardware technology, not vice-versa. Products should be natural and intuitive to use.

Some other important principles of UCD, which incorporate the user's perspective into software development, are:

- **An appropriate allocation of function between user and system**

Determining which aspects of a job or task should be handled by users and which should be handled by the system is of critical importance. This division of labour should be based on an understanding of human capabilities and their limitations, as well as a thorough grasp of the particular demands of the task. The input of end-users to determine this allocation is crucial to ensure user acceptance.

- **Iteration of design solutions**

Iterative software design entails the feedback of end-users following their use of early design solutions. These may range from simple paper mock-ups of screen layouts to high

fidelity prototypes. The users attempt to accomplish “real world” tasks using the prototype; their feedback is used to further develop the design.

- **Empirical measurement of product use**

Product use should be empirically measured so as to provide both quantitative and qualitative data for markers such as user satisfaction and user behaviour. UCD entails observation and measurement of user behaviour, careful evaluation of feedback, insightful solutions of existing problems, and strong motivation to make design changes. User-derived feedback about ease of use and ease of learning is collected directly and/or indirectly from users, and then transformed into design recommendations and decisions.

- **Multi-disciplinary design teams**

User-centered software development is a collaborative process that benefits from the active involvement of various parties, each having insight and expertise to share. The development team should be made up of representatives from all groups having a stake in the proposed software, and also depends on the potential importance of each contribution. Depending upon the circumstances, the team may include managers, usability specialists, training and support staff, software engineers, quality assurance representatives, and of course end-users themselves, i.e. the people who will use the final product.

3.2. User-Centered Design Lifecycle

The UCD Lifecycle and the stages involved can be described by the ISO 13407 [1998] standard on human-centered processes for interactive systems. The figure below describes the main activities as suggested by this standard:

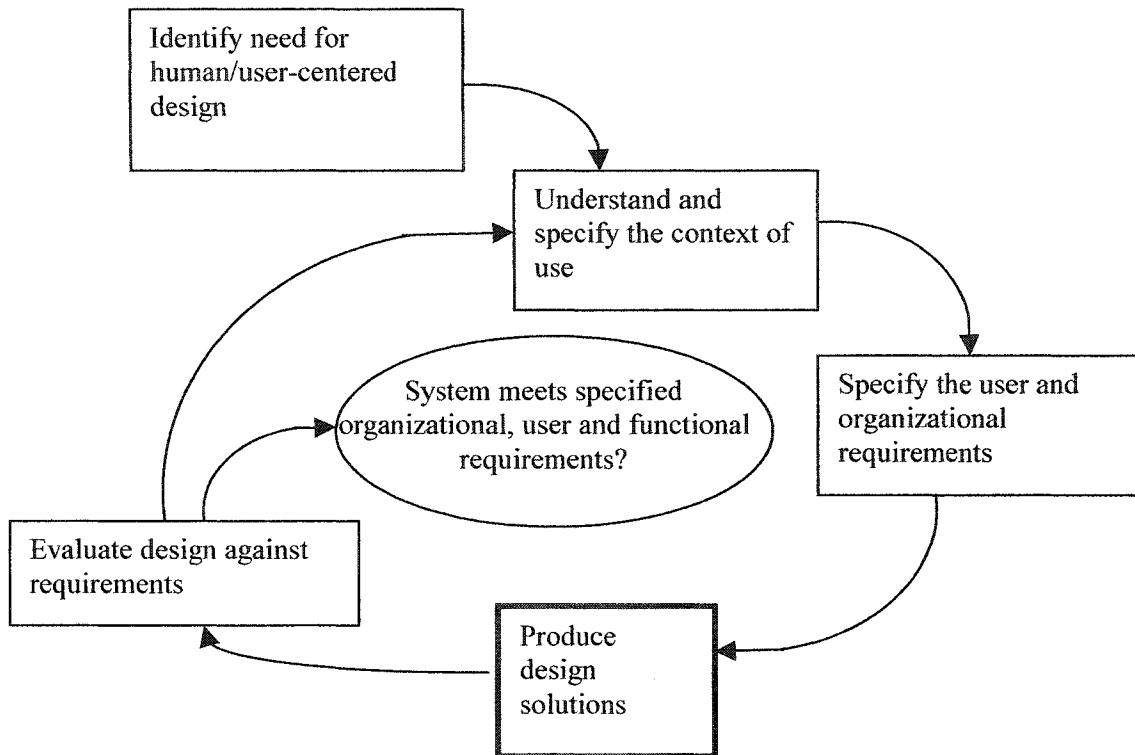


Figure 3-1: ISO 13407 Standard

A more detailed explanation of each step is as follows:

Identify need for human/user-centered design

Everyone concerned in the development process should commit to the user-centered design philosophy, and help create a design plan whereby there is ample time and opportunity for engaging in user requirements elicitation and testing, as well as the more technical aspects of development. The design plan is a working document which is first produced as an outline and which is then reviewed, maintained, extended and updated during the design and development process.

Understand and specify the context of use

During this step, the following information is gathered, analyzed and documented: (1) The characteristics of the intended users; (2) The tasks the users will perform; (3) A hierarchical breakdown of the global task; (4) The overall goals of use of the system for each category of user; (5) The characteristics of tasks that may influence usability in typical scenarios, such as frequency and duration of performance; (6) The environment in which the users will use the system.

It is important at this early stage to set down some markers as to what the minimal, as well as the optimal system requirements should be, with the intention to user-test in these environments before release. Relevant characteristics of the physical and social environment should also be considered.

Specify the user and organizational requirements

In most software development lifecycle models, a major activity is to specify the functional requirements for the system. For user-centered design, it is essential to extend this activity to create an explicit specification of user and organizational requirements, in relation to the context of use description. This specification should be described with the following considerations: (1) The quality of the UI and workstation design; (2) The quality and content of the tasks of the identified users. This includes the allocation of tasks between different categories of users, as well as user comfort, safety, health and especially motivation; (3) Effective task performance especially in terms of the transparency of the application to the user; (4) Required performance of the new system in relation to operational and financial objectives.

Produce design solutions

The next stage is to create potential design solutions. Please note that this is the part of UCD in which we are interested for the remainder of this chapter. This stage has the following objectives:

- Using existing knowledge, such as standards, guidelines, and **patterns**, to develop a proposed design solution;
- Making the design solution more concrete (using simulations, paper prototypes, mock-ups etc.);
- Showing the prototypes to users and observing them as they perform specified tasks;
- Using user feedback to improve the design;
- Iterating this process until design objectives, which include usability-related objectives, are met.

Evaluate design against requirements

Two different approaches are used for evaluation: Formative and Summative. Formative evaluation provides feedback that can be used to improve design. Summative evaluation assesses whether user and organizational objectives have been achieved during design. Whichever evaluation approach is used, it is important to understand that the results are only as meaningful as the context in which the system is tested. In addition, long-term system use should be monitored and evaluation results should be reported.

So far, our discussion has surrounded ISO 13407 [1998], which is one standard used to describe UCD for interactive systems. Another standard which should be investigated is ISO 15527 [2001]. Figure 3-1 summarizes a process model of human-centered system development. The seven processes in the model, which are defined by a set of base practices, are conformant to ISO 15527 [2001].

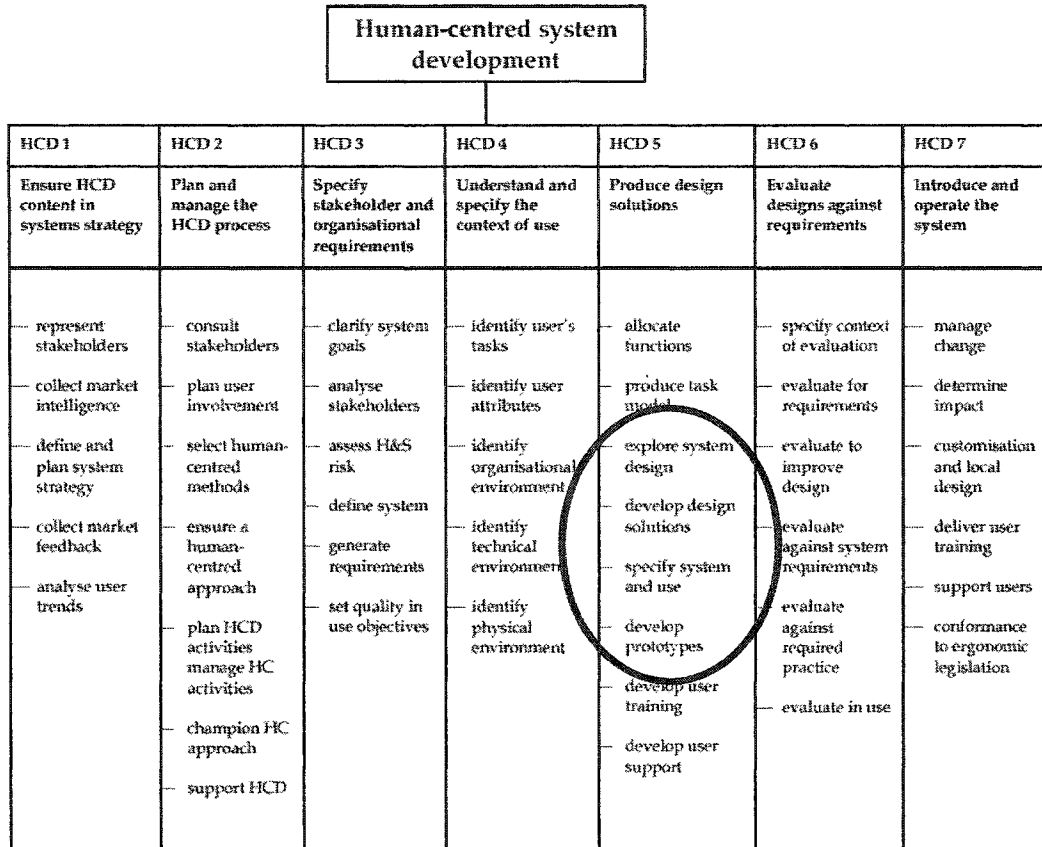


Figure 3-2: Human-Centered System Development as per ISO 15527

In both standards, the patterns that I will discuss for the remainder of this chapter come into play during the stage of producing design solutions. The relevant practices have been outlined in both Figures 3-1 and 3-2, indicating the stages of UCD development that I will concentrate on for the remainder of this chapter, as well as for chapters 4 and 5. In future avenues in this thesis, I will discuss Pattern-Supported Integration (PSI), which also deals with using patterns in other stages of development, such as requirements and testing.

3.3. Patterns in User-Centered Design

Granlund and Lafrenière's [1999] Pattern-Supported Approach (PSA) was a starting point for linking patterns to UCD. PSA aims to support early system definition and conceptual design through the use of HCI patterns. HCI patterns are used to describe business domains and processes, tasks, structure and navigation, and GUI design [Borchers 2001]. In PSA, the scope of patterns deals with all of the practices in HCD 5 of design, as illustrated in Figure 3-2. The main idea that can be drawn from PSA is that HCI patterns can be documented according to the UCD lifecycle, and they can give us knowledge as early on as during system definition. In other words, during system definition and task analysis, depending on the context of use, we can decide which HCI patterns are appropriate for the design phase. PSA shows the beginnings of associating patterns with the UCD lifecycle, however the concept of linking patterns together to result in a design is not tackled, which brings us to our Pattern-Oriented Design approach.

Pattern-Oriented Design, or POD is a design approach we developed [Javahery and Seffah 2002]. Investigations are based on several years of web design development and informal ethnographic interviews with software developers. These investigations reported best practices for bridging design practices and software tools, in particular for web interactive applications. The different interviews highlighted that in order to render HCI patterns understandable by novice designers and software engineers who are unfamiliar with UCD and usability engineering, patterns should be presented to developers as part of the design **process**. Therefore, Pattern-Oriented Design consists of understanding when a pattern is applicable during the design process, how it can be used, as well as how and why it can or cannot be combined with other related patterns.

Simply put, POD is an approach using patterns to achieve design solutions for UCD. It is important to highlight this so as to distinguish between HCI patterns as a component of design and POD as a method of taking these components to achieve design. POD will be discussed in further detail in the following section.

3.4. Pattern-Oriented Design

The medium (e.g. natural language or narrative text) generally used to document patterns, coupled with a lack of tool support, compromises the potential use of HCI patterns. These preliminary observations motivated us to investigate a **systematic approach** for incorporating HCI patterns to achieve design solutions [Javahery and Seffah 2002]. An underlying goal of our investigations was to promote pattern-based development among software developers who are unfamiliar with HCI design and usability engineering techniques. POD involves transferring the knowledge gained by usability experts to software engineers through a systematic approach facilitated by tool support. Our motivation is to help novice designers apply patterns correctly and efficiently. Tool support for pattern-oriented design should enhance the pattern user's understandability, decrease the complexity of a pattern, and eliminate terminological ambiguity. At the same time, the pattern language should be put into practice in a real context of use, which is a critical issue for making pattern languages a cost-effective vehicle for gathering and disseminating best design practices among software and usability engineering teams.

To better understand the process of pattern-oriented design, we interviewed a number of developers using our patterns from the UPADE web language. These interviews revealed that in order for these patterns to be useful, developers need to know how patterns can be combined to create a complete or a partial design. Providing a list of related patterns, as is the case for most HCI pattern languages, is insufficient to effectively drive design solutions. Relationships between patterns are key notions in the understanding of patterns and their use. We suggest four different types of relationships between patterns: superordinate, subordinate, competitor, and neighbouring.

- A superordinate pattern is superior to the described pattern, and can contain both the target pattern and other patterns.
- A subordinate pattern is a pattern that can be embedded in the described pattern.
- A competitor pattern provides the same solution as the described pattern or alternatively, provides a competitive solution that can solve the same problem.

- A neighbouring pattern belongs to the same pattern category (family) and design step as the described pattern.

In Appendix A, two detailed examples of patterns with their respective relationships are presented. In what follows, I illustrate how these relationships come into play when combining patterns from our UPADE web language. I will also demonstrate how patterns can be connected to the iterative design process for an internet-based information system (IBIS).

The first step of the IBIS design process begins with the description of the *architecture* of the entire web site. During this step, three basic patterns can be combined to organize the content of a complex web site. These patterns are the Sequence, Hierarchical and Grid patterns. The simplest architectural pattern is the Sequence pattern which organizes web application content as a sequence, or a linear narrative. The Hierarchical pattern is a tree-based hierarchy, and is one of the best ways to organize complex bodies of web information. Hierarchical organization schemes are particularly well suited to organizing a complete web site. Finally, the Grid pattern should be used when topics and contents are fairly correlated with each other, and there is no particular hierarchy of importance. Procedural manuals, lists of university courses or medical case descriptions are often best organized using Grid patterns. However, for a large and complex IBIS, its content will require a combination of these basic patterns, which is called the Composite pattern. Figure 3-3 illustrates the Composite pattern. Among the many relationships that exist between these three basic patterns, we note that the Composite pattern is superordinate to the Sequence, Grid and Hierarchical patterns.

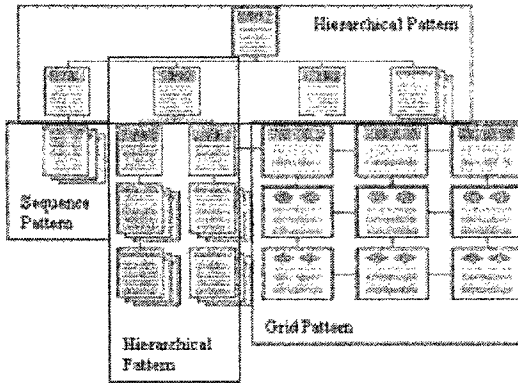


Figure 3-3: The Composite pattern

During the second step, the designer applies *Structural patterns* to establish a consistent physical and logical screen layout for each page that was defined in the previous step. This step involves applying *Page Manager* patterns, which are a type of structural pattern: Some of the patterns that may be used here include Focus, Utility, Navigation, Tiled and Stack Pages:

- Focus page is the fountainhead and center of a website. It must balance aesthetics and practicality to attract the user, such as the home page.
- Utility page provides extra information or assistance without interrupting the workflow, such as with a pop-up information window.
- Navigation page groups all information into a page, and each item directs the user to the appropriate content, such as a geographical map.
- Tiled page structures and presents contents to the user from more general to specific by dividing the page into several surfaces.
- Stack page groups content into categories which have no obvious hierarchy; this is done by designing several surfaces stacked together and labelling them appropriately.

Different relationships exist between these patterns, and even between these patterns and those used in the previous design step. As an example, all the Structural patterns are subordinate to the Architectural patterns from the last step. In addition, to further illustrate some relationships, Tiled and Stack patterns are competitors (see figure 3-4).

This means that if you choose the Tiled pattern as a basic model for your home page, you cannot use the Stack pattern for any of the subsequent pages. Moreover, the Navigation page pattern is highly suitable for organizing the content of the first page of a Sequence pattern. However, it cannot be used for subsequent pages. Such knowledge can be critical for pattern users because if it is not taken into consideration during design, it can compromise the benefits of the pattern.

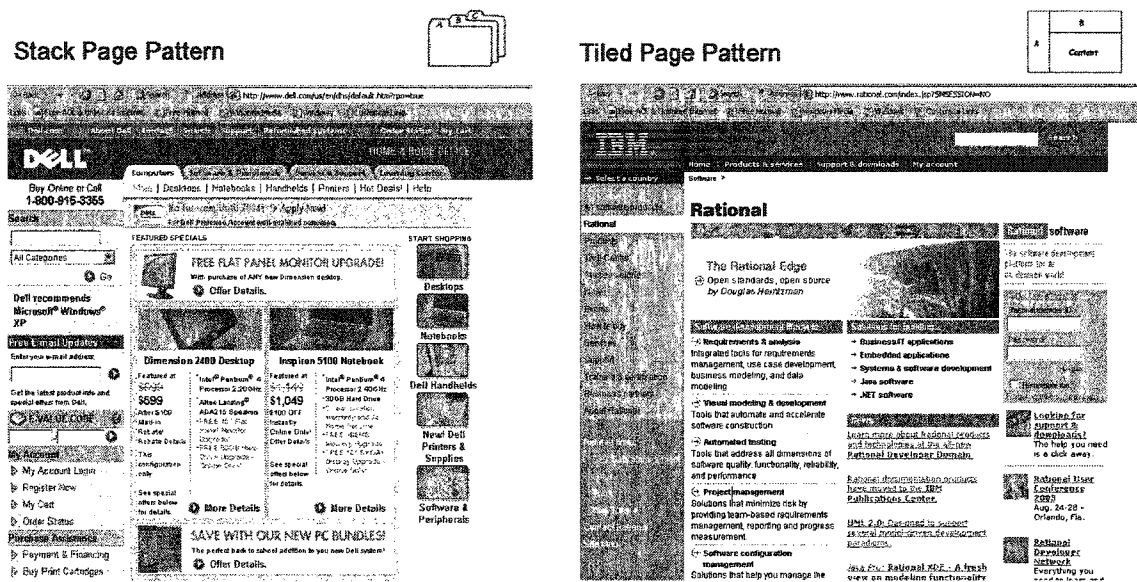


Figure 3-4: Comparison of Stack and Tiled Page Patterns

The third step of the IBIS design process involves employing *Information Container* patterns, the second type of Structural patterns, to quickly "plug in" an information segment for each page. Long before the Web was invented, authors of technical documents discovered that users appreciate short segments of information. Such design practices should be embedded in the design process and presented to the designer. For example, for users, how long does it take to determine if a large document contains relevant information? This question is a critical design issue. The patterns which can be applicable here are as follows:

- On Fly Description provides the user with a short description of the object when the mouse hovers over it.
- Form pattern collects input information from the user, such as with an online registration form.
- Bullet pattern collects a small amount of information from the user, usually with preset choices. This way, the user need only click on the radio button, such as with a survey questionnaire.
- Executive summary provides an information preview or summary for a certain topic of choice.

Although all of the above patterns can be applied independently, one of the main strengths of the POD approach is that developers can exploit pattern relationships and the underlying best practices to come up with concrete and effective design solutions. As an example, the Executive Summary pattern, combined with the Map or Index Browsing patterns from Navigation Support, allows users to preview information about a certain topic before spending time to download, browse and read different pages (Figure 3-5). Executive Summary should be weighted as a highly recommended subordinate pattern when pattern users try to use the Map and Index Browsing patterns.

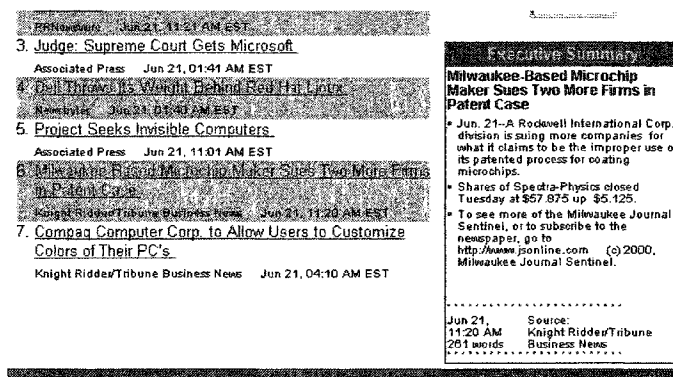


Figure 3-5: The Executive Summary Pattern

The fourth and final step consists of building the navigation support. It is possible to consider navigation elements earlier in conjunction with other patterns, as mentioned during the third step of the process. *Navigation Support patterns* suggest different models for navigating between information segments and pages. The following is a selection of patterns as proposed by our UPADE web language:

- The Convenient Toolbar pattern (mentioned in previous sections) assists the user to reach convenient and “safe” pages regardless of the state of the artefact.
- The Shortcut pattern helps the expert user reach favourite and frequently-visited pages or documents, and is generally located on the home page as a list box.
- The Site Map pattern collects links to all web pages so that the user can easily acquire the skeleton of the whole site.
- The Dynamic Path pattern indicates the user’s entire path starting from when the web application was initially accessed, and is similar to “breadcrumbs” in other pattern languages.
- The Index Browsing pattern allows the user to easily and promptly navigate amongst important content pages, and is located consistently throughout the website.

To illustrate an example of existing relationships, the Index Browsing and Dynamic Path patterns are considered neighbouring since they belong to the same design step. Although they are both used for navigation support, they are not used to solve the same usability problem and are applied in different contexts. Dynamic Path is used to navigate between pages in an already-taken path, and gives the user a sense of safety and control. Index browsing is generally used to navigate amongst important content pages, and allows the user to reach these pages safely.

Knowledge about context-oriented relationships, as described above, can be very useful to pattern users. They can be a guide in choosing the best solution for a specific user problem based on a particular context. If such information were embedded in an editing tool that supported pattern-oriented design, it would be beneficial for developers during

their prototyping and design activities. The lack of such an editing tool motivated our team to develop the UPADE editor for POD, which is the topic of the next section.

3.5. UPADE Editor: A Tool for Pattern-Oriented Design

The UPADE editor serves as a support tool for pattern-oriented design, and has three key functionalities. First, the most important functionality provided by the UPADE editor is the possibility given for both pattern writers and developers, using existing relationships between patterns, to define new patterns or create a design by combining existing usability patterns. For example, designers can begin their design by selecting the Home Page pattern from the pattern toolbox. They then insert Navigation Support patterns such as Site Map and Index Browsing, as well as Information Container patterns such as Quick Summary in the home page (see Figure 3-6).

Secondly, in order to facilitate pattern combination, the tool supports different hierarchical and pattern-oriented design levels. Three views are possible: Application View, Page View, and Navigation Support View. At the web application level, designers can establish a prototype of an entire web application by combining Content Architecture patterns. The Page level is where designers can develop a design solution for a specific Web page by embedding required page elements on the page. The Page Element level is where designers can pick up certain Navigation Support patterns and Information Container patterns from the product pattern box and combine them to establish a prototype of a page. Designers can start at any of these levels, and are encouraged to select a specific design step from the Process Viewer.

Thirdly, the UPADE editor provides a mechanism to check and control how patterns are combined. Using the different relationships that exist between patterns, UPADE editor automatically examines the compatibility of patterns and offers relevant task-sensitive advice to designers. For example, if the Stack page pattern has been used to organize the structure of a page, the Tiled page pattern cannot be used since they are competitors.

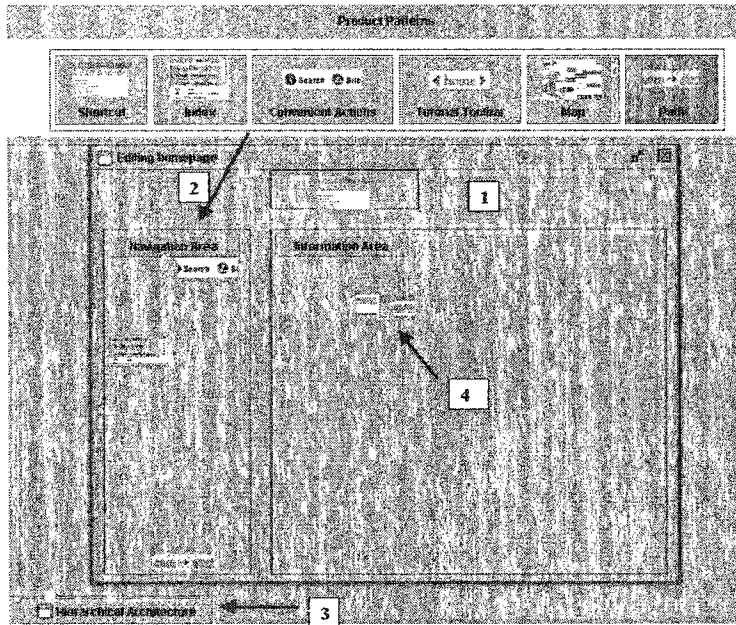


Figure 3-6: The UPAGE Editor

4. Redesign of a Bioinformatics System with Patterns: Incorporating User Experiences

What I have explained thus far in this thesis deals with designing web-based interactive systems from scratch. In addition, the systems I described were not specific to any type of user group or community. In this chapter, I will introduce the idea of using patterns to **redesign** an existing system for a particular target group. My discussions will focus on a case study I performed with a web-based Bioinformatics system. This study consisted of first performing usability evaluations on an existing website, and then applying appropriate HCI patterns to redesign the site based on the discovered usability problems. The purpose of this study was two-fold: Determine if patterns are effective in redesigning an existing system, and how usability evaluation can be used as a complementary technique in redesign.

To investigate patterns and redesign, I was looking for an IBIS with a large amount of content, many users, as well as high functionality. I decided to work with one of the most influential Bioinformatics web-based systems, the National Center for Biotechnology Information at <http://www.ncbi.nlm.nih.gov>. This site is a well-established site, with a large community of users, and a vast amount of information. High user access and a great deal of content can often cause usability problems. I wanted to see what usability evaluations would uncover in terms of problems with the site, and if I could use patterns for improving the design.

4.1. Background to Bioinformatics Systems

Bioinformatics is a discipline at the forefront of the biological and computational sciences. Originally, in the 1980s, it was associated with the analysis of biological sequence data, but is now used to encompass all computer applications in biological sciences. It is an emerging field that has gained much attention in the last few years, and deals with DNA sequence analysis and searching, protein structure prediction, and

biological-based algorithms, amongst other areas. Current research in bioinformatics has necessitated skill sets that are unique in terms of information-gathering, data-mining and knowledge-building [Higgins and Taylor 2000].

According to Higgins and Taylor, “The ultimate aim of bioinformatics must surely be the complete understanding of an organism – given its genome” [Higgins and Taylor 2000, p. ix]. To achieve such a daunting task, it is apparent that tools and resources are needed that are both effective and efficient. The web, as a hyper-media based information system, has to be the single most computational resource that has allowed the quick expansion of information sharing in bioinformatics [Attwood and Parry-Smith 1999]. There are various online initiatives dedicated to providing biocomputing services and holding data repositories. The National Center for Biotechnology Information (NCBI) is one such initiative, and the leading North American information provider.

NCBI was established in 1988 in Maryland, USA, as a division of the National Library of Medicine [Attwood and Parry-Smith 1999]. The role of the NCBI is to advance scientific knowledge of the underlying molecular and genetic processes surrounding health and disease. It attempts to achieve this feat through the development and use of new information technologies and biocomputing power. According to Attwood and Parry-Smith [1999], the NCBI’s specific aims include:

- Creation of automated systems for storing and analyzing biological information
- Development of advanced methods of computer-based information processing
- Facilitation of user access to databases and software
- Coordination of efforts to gather biotechnology information world-wide

The complexity surrounding the NCBI site as an interactive system is two-fold: First of all, we are dealing with a complicated data repository of rich and critical information in a specific field of research, thus with a specific user community. Secondly, the medium for dissemination of this information is the web, which has its own specificities with regards to user interaction.

Informal ethnographic interviews carried out with bioinformatics researchers from three different labs (university-based lab from Queen's University; hospital teaching lab from McGill University; and state-run research facility in France) resulted in the following interesting discoveries:

- The NCBI site is by far the most popular bioinformatics information provider currently in this field.
- It is rich in information and provides access to nucleotide, protein and literature databases.
- It contains computerized information processing methods and tools which are used even on a daily basis by some biomedical researchers.
- However, users also pointed out a number of problems with the site, including: Difficulty to find desired information, poor site organization, information overload, easy to get lost on the site, and frustration because of lengthy waiting times for receiving input.

The results of the ethnographic interviews, although informal, did give me enough information to progress to the next step, and decide that further user evaluation and usability studies are appropriate for this site. It was hoped that this would lead to suitable pattern selection for a possible redesign of the NCBI site.

4.2. Framework used in Redesigning a Bioinformatics IBIS

Understanding and specifying the context of use is an important step towards the development of systems that are human-centric (Figure 3-1). Gathered information about users, their interaction behavior, and their experiences with an existing system (as is the case with redesign), can shed light on how a particular interactive system is perceived and used. This includes factors such as user control, consistency and efficiency, and, most importantly, whether system functionalities are able to effectively support user tasks.

The framework I used to redesign the NCBI site is described in Figure 4-1. The starting point consisted of using personae (explained in Section 4.3) to describe the users of the NCBI site, as well as their basic characteristics. These personae helped to identify potential users for the usability evaluation step of the existing site. The results of users who participated in the usability evaluation step were used as feedback to (1) Further enhance the personae and determine precise interaction behavior. These refined personae, along with context information, gave a clearer picture of the context of use. (2) Determine usability issues and existing problems with the site. Both these points helped me choose appropriate HCI patterns for UI (re)design.

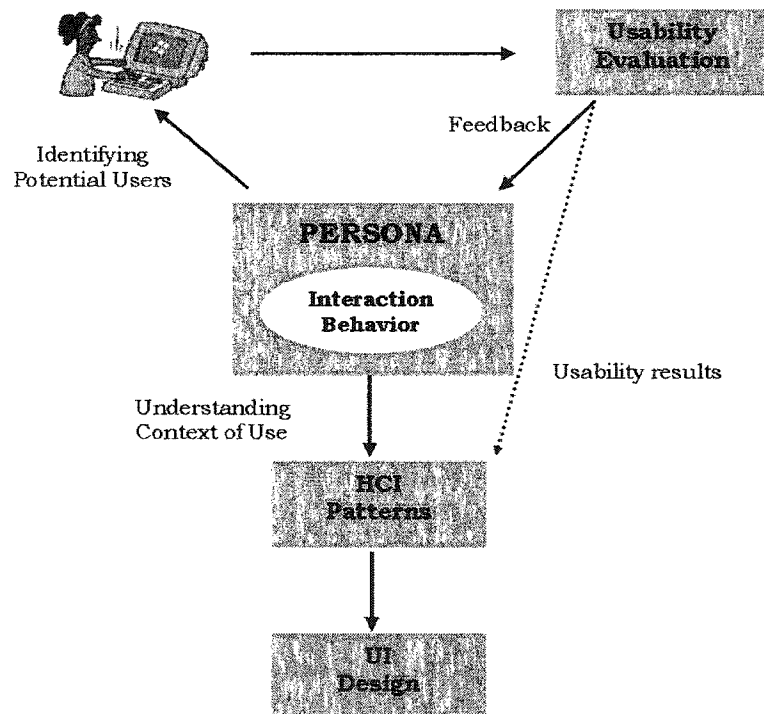


Figure 4-1: Usability Evaluation and Persona in HCI Pattern Selection

4.3. Modeling the Bioinformatician's Behavior through Personae

Persona is an example of a UCD technique that can be used to define the users of the system. A persona has a fictional name, educational background, and description of work habits or personal characteristics. It is intended to help developers better understand both the users and context of use for a planned tool or interactive system. Cooper argues that designing for any one external person is better than trying to design vaguely for everyone [Cooper 1999].

Website interaction predominantly deals with the user experience of moving through the site and interacting with all of its various parts. Web design must incorporate what people do on the site rather than simply how it looks [Nielsen 2001]. For a bioinformatics website, we know very little about user behavior and experiences. More consideration of all aspects of the bioinformatician's experience and interaction with the website are necessary, such as how the site is perceived, learned and mastered. This includes ease-of-use and, most importantly, the needs that the site should fulfill with respect to services and information. Any design or redesign initiative should first focus on the behavior of users. By understanding and analyzing users and their behaviors, we can build personae for the target user community. We can then choose appropriate patterns, and design a site according to this information.

The idea of constructing personae to describe a target clientele has been part of marketing studies for some time [Weinstein 1998]. Their goal is to create a set of personae that best represents their marketing audience, based on statistical and demographic data. By establishing personae, they are able to create a marketing campaign that will have success in exactly the areas that they choose to focus on.

In software design, Alan Cooper, the father of Visual Basic, proposed personae for modifying the design of his product [Cooper 1999] so as to redirect the focus of the development process towards end users and their needs. Alan Cooper believes that for each project, a different set of personae should be constructed. This is because each

project targets different users in different contexts of use. Each persona needs to have a name, an occupation and personal characteristics such as likes, dislikes, needs and desires. In addition, each persona should outline specific goals related to the project. These goals can be personal (e.g. having fun), work-related (e.g. hiring staff), or practical (e.g. avoiding meetings) [Tahir 1997].

The users selected for this study were from the set of three personae in table 4-1 since they represent the main audiences for the NCBI site. To create this sample, I used domain analysis and ethnographic interview results (as described in section 4.1) to postulate the users of the NCBI site, as well as the kind of experiences these users may have. A biomedical expert was advised to provide advice on domain-specific information.

Table 4-1: A Set of Personae for NCBI Site Users

A Set of Personae for NCBI Site Users		
<p>Debbie Smith</p> <ul style="list-style-type: none"> ❖ 24 years old ❖ Masters student in Biochemistry ❖ She lives with roommate away from home ❖ She jogs daily and plays soccer twice a week ❖ She uses the internet daily for e-mail access, and often searches for biological information related to her research ❖ She accesses the NCBI site often from both home and her university lab using a PC with a 17 inch screen and high-speed internet access ❖ She loves giving the image of being intelligent; enjoys intellectual conversation ❖ She doesn't like asking people how to do things; likes to figure it out on her own ❖ Very fast learner and hard worker 	<p>Xin Li</p> <ul style="list-style-type: none"> ❖ 37 years old ❖ Masters in Molecular Biology ❖ Researcher in a pharmaceutical company ❖ He is married with two young children ❖ He plays tennis and squash at times ❖ He uses the internet daily for e-mail, access to the company's intranet and information portal, as well as for information searches related to his work ❖ He accesses the NCBI site weekly from his office using a PC with a 17 inch screen and through a network connection ❖ He doesn't really bring his work or research endeavors home, and only uses the internet at home (56k modem) for surfing and email ❖ He wants to finish work as soon as possible and go home; doesn't like staying at work late 	<p>Dr. Thomas Johnson</p> <ul style="list-style-type: none"> ❖ 57 years old ❖ PhD in Parasitology ❖ University Professor in the Faculty of Agricultural and Environmental Sciences ❖ He is married with 3 children; all of them have moved away from home ❖ He plays golf once a week ❖ Uses internet daily for e-mail access and information searches related to his research ❖ He accesses the NCBI site a few days a week from both home and his office; from home, he uses a 15 inch screen and 56k modem; from his office, he uses a 17 inch screen and a network connection ❖ He has a few graduate students working in Bioinformatics, and needs to stay updated on biocomputing tools and resources ❖ The worst thing anyone can tell him is that he is not fast enough

If constructed effectively, a persona should be sufficiently informative and engaging so that it redirects the focus of the development process towards end users and their needs. However, constructing such an effective persona is not easy. To increase their effectiveness, personae should be supported by user and empirical data. In the next section, I show how one can refine an effective persona with usability evaluation. Similar to this approach, Tahir [1997] suggested creating user profiles through contextual

inquiry. Tahir, along with Cooper [1999], are clear in positioning persona descriptions as the starting point around which usable products can be constructed.

Therefore, to enhance and render the personae more informative, I decided to gather more specific user information from the evaluations with end-users. If anything were to shed more light on user behaviors and characteristics, it would be information gathered from the users themselves!

4.4. Heuristic and Psychometric Evaluation

There exist a number of evaluation techniques in usability engineering that are appropriate for assessing the usability of interactive systems, including internet-based information systems. Usability evaluation techniques include field or laboratory observation, remote testing, the measurement of quantitative metrics (such as error rates or task efficiency), participatory design, heuristic evaluation, and the administration of objective questionnaires (psychometric assessment). Heuristic evaluation and psychometric assessment, the two techniques used in this study, are outlined below.

Heuristic evaluation is closely associated with Jakob Nielsen's usability philosophy [Nielsen 1994]. Nielsen has described heuristic evaluation as a relatively low-cost method for identifying usability problems in an existing program. It is an inspection method where experienced users evaluate a program's UI against a set of accepted principles, or heuristics. Early lists of heuristics were lengthy and difficult to apply. To reduce testing costs, Nielsen came up with a list of ten heuristics that cover what he considered the most important aspects of usability. Some of these principles are redundant, and I came up with a somewhat reduced set of nine heuristics (from the original ten), as described in Table 4-2. In addition, I adapted these heuristics for the web.

Nielsen [2001] claims as few as 3-5 experienced evaluators are necessary for heuristic evaluation, stating that they will be able to detect the majority of usability problems. Since heuristic evaluation is a subjective usability method, it has been used as an

approach for building first a qualitative picture of user experiences. To a certain extent, such a picture was used to tailor the questionnaire I conducted with bioinformaticians.

Table 4-2: Definition of Heuristics [adapted from Nielsen 1994]

Heuristic	Definition
1. Visibility and Navigation	Sections and links should be clearly marked, and users need to know "Where am I?" and "Where can I go next?" System should keep users informed about what is going on, through feedback within reasonable time (e.g. progress indicators).
2. Language and Communication	The system should "speak" in a language familiar to the user (e.g. technical terms should be avoided).
3. Control	The user should always feel like they have a way out of the system and unwanted states, such as with the "home" button
4. Consistency and Standards	The user should have a sense that wording used within content and buttons, as well as system behaviors are consistent (e.g. conform to platform standards, pages should have uniform organization).
5. Error prevention and Recovery	Prevent errors in the first place through good design; give intelligible messages when errors occur.
6. Recognition not recall	Options, actions, and instructions for system use should be easily available (e.g. users should not have to memorize dialogue across pages). Good labels, visibility of path taken, and descriptive links are crucial for user recognition.
7. Efficiency	The system should accommodate users of varying experience (e.g. give shortcuts for experienced users).
8. Minimalist Design	Only relevant and important information should be contained in dialogs. Information that is not relevant simply clutters the interaction between the system and user.
9. Help	Help information should be easy to find, specific to user goals, and should provide concrete steps.

Psychometric assessment, through objective questionnaires administered to users, is another usability evaluation method. Such questionnaires gather user perceptions in a systematic way. They are analogous to structured interviews in that questions are presented in the same way to all respondents.

In the past, we have used the Software Usability Measurement Inventory (SUMI) to assess user perceptions of a software system [Kirakowski & Corbett 1993]. The SUMI is a multi-dimensional inventory questionnaire that evaluates different aspects of user satisfaction. However, for this study, I decided to administer a tailored questionnaire for two particular reasons. First, since we are dealing with a web-based information system, the mode of interaction between users and the system is different from a traditional software system, and therefore necessitates specific web-based scenarios and questions. Secondly, the study entailed participation from a particular user group and community that uses the NCBI site to accomplish specific tasks. I wanted to make sure that the questionnaire was domain-specific, and asked appropriate task-related questions.

The questionnaire created for usability evaluation with end-users consisted of three parts: (1) User Information, (2) User Evaluation of NCBI Site, and (3) General Questions. The following experts were consulted to assure both quality and precision in terms of the quantifiers used: A cognitive psychologist, a senior usability expert, and a biomedical specialist. A sample of the questionnaire can be found in Appendix B. The purpose of the first part of the questionnaire was to gather some demographic and user information, while the third part was aimed at gathering general impressions of the site from the user.

The second part of the questionnaire contained specific questions, which enabled me to quantify user experiences with certain properties of the site. Similar to McKenzie [2000], heuristics were used to describe different facets or properties of the site. In other words, each set of questions was correlated with a particular heuristic. Using the same list of nine heuristics introduced earlier in this section, I was able to cover the most important aspects of usability by asking specific questions. For example, if we are to take the first heuristic from Table 4-2 (“Visibility and Navigation”), the following questions were asked to assess user experiences with relation to the visibility and navigation of the site:

- Do you find it easy to navigate on the NCBI website, especially when performing a new task?
- Is it visually clear what is a link or a button?

- Do you receive feedback and requested information promptly, such as when you perform a BLAST search?
- Is it easy to get lost when looking for information?

The full mapping of questions to heuristics used in the questionnaire is illustrated in Appendix C.

4.5. Method

In total, there were 19 participants for the study: 16 users for the psychometric assessment (questionnaire) and 3 UI experts for the heuristic evaluation.

Participants of the questionnaire were from 4 research groups: A Bioinformatics research group from France, and 3 major biomedical research labs affiliated with Canadian universities (University of British Columbia, Queen's University, and McGill University). In addition, 2 participants of the study were medical practitioners in private practice. All participants were given consent forms to sign before they participated in the study (see Appendix B), and results were recorded in anonymous form.

A small number of UI experts (3) were also used in this study, but this was due more to resource limitations of this project than to acceptance of Nielsen's claim of using 3-5 evaluators [Nielsen 2001]. One evaluator was a senior usability expert, whereas the other 2 were junior usability experts, with at least 2 years of experience in the field. The UI experts were asked to comment on the NCBI site with relation to the 9 heuristics outlined in table 4-2, as well as to give comments and suggestions for improvement of navigation structure, home page, site map, and search tools. In addition, they were given space to write any other comments they deemed relevant that were not covered in the heuristics or specific items asked.

Unlike heuristic evaluation, small samples (e.g., $N = 3$ to 5) are inadequate for psychometric methods such as the questionnaire administered in this study. This is

because results from small samples tend to be statistically unstable and subject to sampling error. Therefore, more representative samples are desired. As a rough rule of thumb, there should be one subject for each item on a questionnaire. My questionnaire had 31 items; thus, a sample of 31 subjects would have been ideal. However, due to resource limitations and the specific user community tackled in this study, I was only able to include 16 participants. This is much better than the 3-5 suggested users by Nielsen, but not as ideal as I would have liked.

4.6. Results

4.6.1 User Characteristics and Behavior

User characteristics and behavior were determined from part 1 of the psychometric evaluation (see Section 4.4) with NCBI site users. The administered questionnaire exposed the following interesting characteristics about participants:

- 63% of participants were male, and 37% were female.
- 63% indicated that English was their first language, whereas 37% indicated other languages including French and Chinese.
- All participants were very familiar with the internet, and had more than 3 years of experience.
- All participants were from biomedical-related fields, including molecular biology, biochemistry, pharmacology and cancer research. Only 1 participant indicated that their actual field of research was bioinformatics, and 1 other participant indicated biology and computer science.
- Highest level of education ranged from B.Sc. to PhD and MD (medical doctor).
- 44% of participants indicated that their current position was a graduate student, while the rest of participants included researchers, Post-Doctoral students, physicians, and 1 database administrator working in the bioinformatics domain.
- Participant leisure activities included a variety of sports, traveling, and reading.

In addition, the following information about user interaction behavior with relation to the NCBI site resulted from the questionnaire:

- 44% of users were using the NCBI site for less than 1 year, while 56% were using the site for more than 1 year. Of these 56%, 78% were using the site on a regular basis for more than 3 years.
- The main tasks users performed were highly dependent on their experience with the NCBI site. Users working with the site for less than 1 year used the site for very different tasks compared to users with more than 1 year of experience (please see table 4-3).

Table 4-3: Task Use and Interaction Behavior of NCBI Site Users

<1 year experience	>1 year experience
<ul style="list-style-type: none"> • Literature and article searches (Pubmed) • Educational and information-gathering • Small amount of advanced tools such as sequence analysis 	<ul style="list-style-type: none"> • BLAST search (protein and sequence alignment) tool • More sophisticated tools such as LocusLink, and a variety of DNA and protein sequence searches • Literature database searches

As a result of the above, the personae were refined to include the new information about user behavior and experiences with the NCBI site. Table 4-4 shows the results of the enhanced personae. These enhanced personae now include only 2 types of users: A novice user and an expert user; which is appropriate with relation to the user characteristics, interaction behavior, and performed tasks based on our sample of bioinformaticians using the NCBI site. By using this newly gathered information, we can now begin the process of pattern-oriented design by closing the gap between actual user experiences and the features offered by such a complex website. We can associate patterns with each type of desired task and user behavior; by combining them, we can build a more usable site with the user at the center of the design process. These selected patterns are highlighted (in bold) in Table 4-4.

Table 4-4: Enhanced Personae of NCBI Site Users

A Set of Personae for NCBI Site Users	
<p>Debbie Smith</p> <ul style="list-style-type: none"> ❖ 24 years old; masters student in Biochemistry; works daily in a lab with other graduate students ❖ She lives with roommates away from home; she jogs daily and plays soccer twice a week ❖ She uses the internet daily for e-mail access, and often searches for biological information related to her research ❖ She accesses the NCBI site 2-3 times a week from both home and her university lab using a PC with a 17 inch screen and high-speed internet access ❖ She recently started doing bioinformatics-based research, and has only been accessing the NCBI site for 6 months (Patterns for Novice users) ❖ She is still unfamiliar with all the menu options and functions (On-Fly Description Pattern) ❖ She is still learning about the NCBI site, and actively reads General NCBI Information and "About NCBI" (Executive Summary Pattern) ❖ She uses the site mainly for literature and article searches (such as Pubmed), educational and information-gathering, and has only started to do sequence alignment searches (Index Browsing and Search Pattern) ❖ She gets lost looking for information after advancing more than 3 layers, and needs to go back to a safe place (Home button on Convenient Toolbar Pattern and Dynamic Path pattern) ❖ She is a hard worker and likes to figure out things on her own (Help button on Convenient Toolbar Pattern) 	<p>Xin Li</p> <ul style="list-style-type: none"> ❖ 37 years old; masters in Molecular Biology; researcher in a pharmaceutical company ❖ He is married with two young children; he plays tennis and squash at times ❖ He uses the internet daily for e-mail, access to the company's intranet and information portal, as well as for information searches related to his work ❖ He accesses the NCBI site almost daily from his office using a PC with a 17 inch screen and a network connection ❖ He has been accessing the NCBI site for 2 years now, and is very familiar with tools related to his research (Patterns for Experts users) ❖ He doesn't really bring his work or research endeavors home, and only uses the internet at home (56k modem) for surfing and email ❖ He wants to finish work as soon as possible and go home; doesn't like staying at work late ❖ English is his second language, and he is not always comfortable with spelling (Index Browsing and Alphabetical Site Map) ❖ He uses the NCBI site for specific tasks, such as secondary structure prediction for proteins (Shortcut pattern and customized MySpace) ❖ Likes to limit his searches to specific species (Advanced search pattern) ❖ Likes to know about recent discoveries and advances in the field (Teaser menu [Welie, 2003] and Executive Summary)

4.6.2 Heuristic Evaluation with UI Experts

The complete results of the heuristic evaluation with the UI experts can be found in Appendix D. A more concise version of results will be given here. All heuristics, except for *Language and Communication*, were found to be problematic. Major problems found by UI experts were (1) Easy to get lost because path or current position is unclear, (2) Difficult to get out of undesired or error states, (3) Inconsistency amongst sites, such as with different menu structures, (4) Information overload, (5) Not enough help and

guidance for novice users, (6) Lack of efficient options for expert users, such as shortcuts.

In addition, UI experts were asked to comment on specific items of the site: Navigation structure, Home page, Site Map, and Search tools. The **navigation structure** was found to be big and fairly complex, so it is easy for users to lose their orientation on the site. The **homepage** was found to be overloaded with links, low in visibility, and no guidance for first time users. 2 out of 3 UI experts suggested that it might be interesting to consider a different home page for different users, based on their experience with the site (i.e. Novice vs. Expert users). In practice, a **site map** is designed to give users who know what they are looking for, fast access to a certain sub-site; and for new users, additional help in locating a page or topic. The site map for the NCBI was found to be complicated and difficult to use for either of these groups of users. **Search tools** on the NCBI site were found to be relevant for more experienced users, but more explanation and control should be given to newer users.

4.6.3 Psychometric User Evaluation

As indicated by the enhanced personae, there seem to be 2 types of users for the NCBI site. This is especially problematic due to the vast amount of information and tools, as well as complicated site structure. This was confirmed by our heuristic evaluation, where UI experts highlighted site problems with relation to two different types of users: Novice and Expert.

When I analyzed the results of the administered questionnaire, once again, these two types of users had differing results with relation to a number of properties of the site: (1) Visibility and Navigation, (2) Consistency and Standards, and (3) Help. The results can be found in Figure 1. The only property of the site that both user groups seemed to be satisfied with was “Language and Communication”.

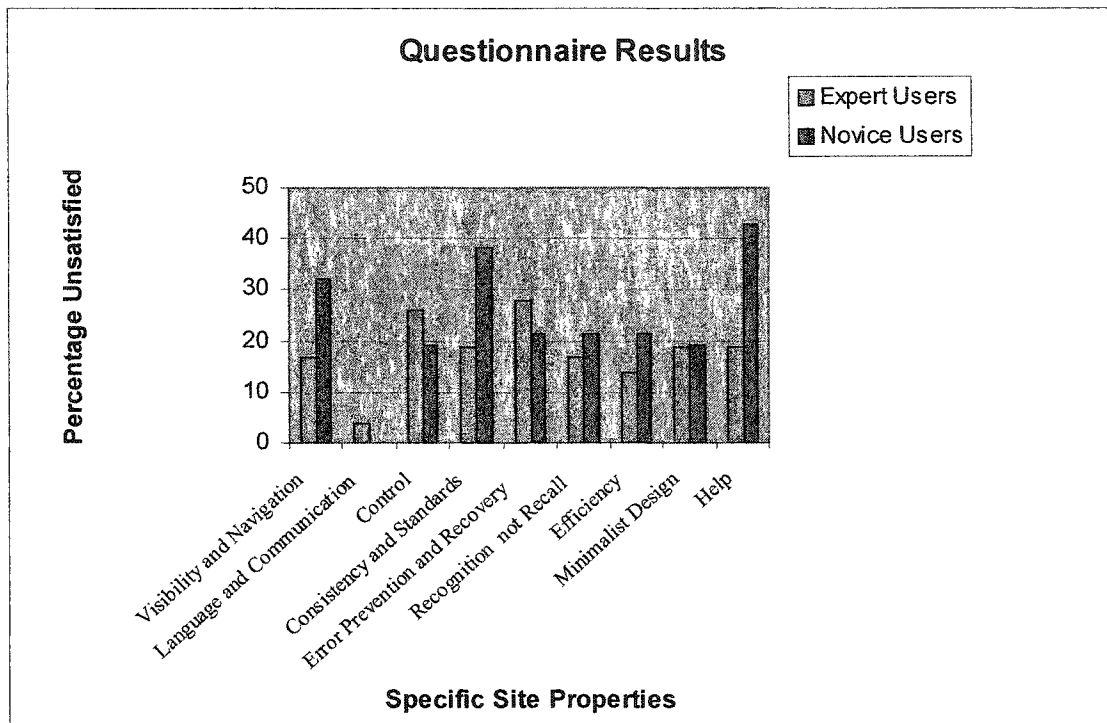


Figure 4-2: Questionnaire Results of Novice and Expert Users

In addition, it was interesting to note that novice and expert users, when taking into account percentage of individuals both unsatisfied and unsure, had noticeably different results (Figure 4-3). It seemed that a greater number of novice users were both unsatisfied and unsure of different properties of the site. This may also be attributed to the questionnaire, in terms of type of questions asked i.e. Novice users may not have easily understood all questions, especially the more detailed ones relating to more advanced tasks.

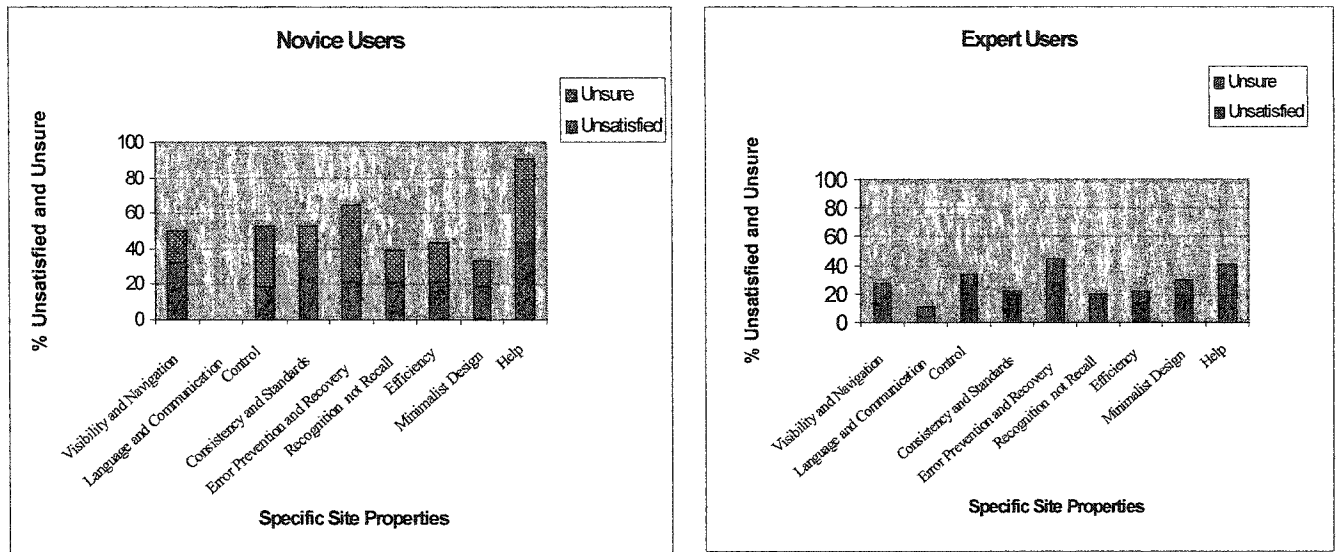


Figure 4-3: Comparing Novice and Expert Users – Unsatisfied and Unsure

4.7. Redesign: Applying Patterns Based on Results

In this section, I will describe the redesign of two pages of the NCBI Site: The home page (Figure 4-4) and the site map (Figure 4-7).

If we start with the **home page**, we notice that there are well-identified zones of content, although overloaded with information. As described in the UI expert evaluations, it is not always clear what is a link. In certain zones such as *Pub Med Central*, there seem to be many individual links but in fact, the whole zone is a single link. This is confusing and frustrating. In addition, the main (left-hand) menu contains textual descriptions of menu items. Aside from being non-standard, this lengthens the menu unnecessarily, so that users have to scroll to find all of the elements. These textual descriptions should be implemented as rollovers (On-Fly Description pattern) for new users. The main menu items are not optimally legible because the font is yellow on a blue background, resulting in reduced contrast and visibility. In summary, the site is overloaded with links, low in

visibility, and first time users will probably give up since there is no guidance. It may be interesting to have a different home page for different users.

The home page is NCBI's face to the world, as well as the starting point for most user visits. Improving a home page multiplies the entire website's usability and increases accessibility and visibility to the many other pages estimated as part of the site (around 5000 pages for NCBI according to our calculations of using menu depth and breadth analysis).

While redesigning the home page, we should keep in mind a number of usability principles applicable for web design. First, the page should be organized for scanning. This should be done in way to facilitate users while scanning down the page, trying to find the area that will serve their current goal. Links are the action items on a homepage, and when each link begins with a relevant word, it is easier for scanning eyes to differentiate it from other links on the page. Secondly, clear affordance of links and navigation elements should exist; their appearance should help users to understand what they represent. The mouse pointer change provided by web browsers, to indicate that the element pointed at is a link, is not sufficient. The designer can use differences in size to establish a hierarchy between links, but HTML text has poor graphic quality and doesn't allow much visual characterization. Differentiation between navigation elements and information is indeed the main affordance problem to be solved. Finally, designers should strive to avoid user errors and facilitate error recovery. Alternative links to enable users to recover quickly and easily from errors, as well as communication in a language familiar to users, are important design considerations.

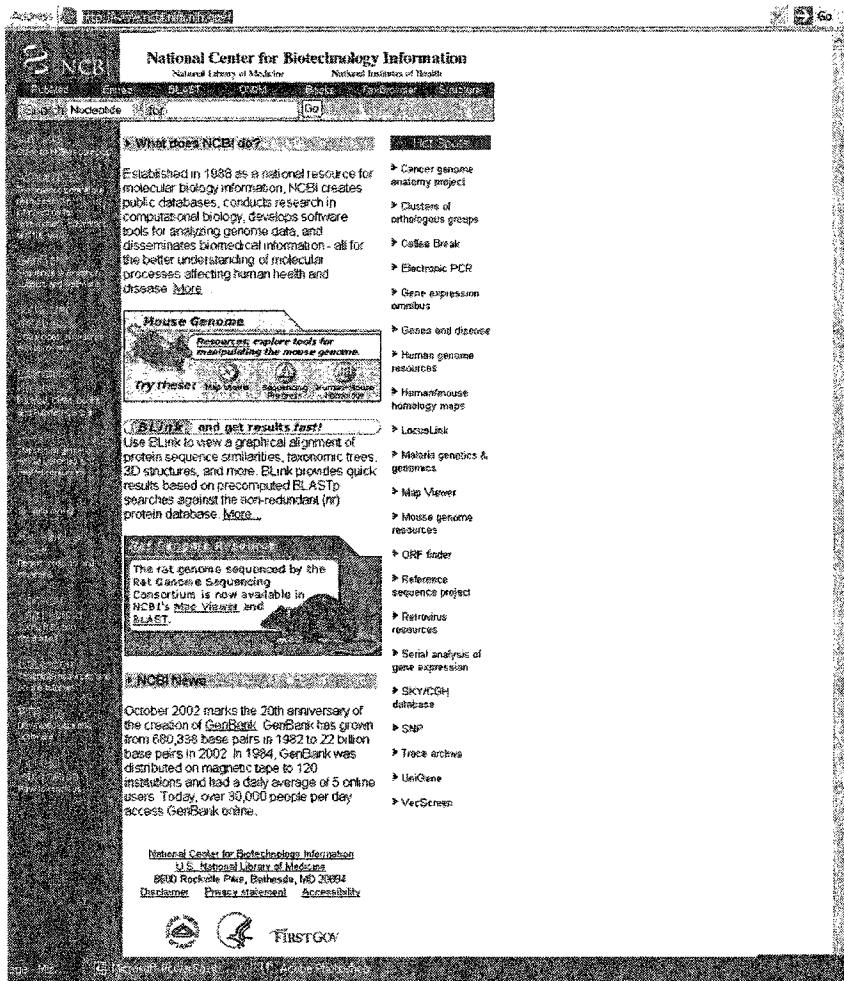


Figure 4-4: Current Home page of the NCBI site

Some important design problems with the current home page, taking into account the usability evaluation results mentioned in this chapter, are as follows:

1. Navigation menu on the left has important site links, however a description of the link is provided under each name; this is only useful for novice users who don't know what kind of information they can find from the links.
2. "Hot Spots" are representative of possible shortcut links that are only useful for expert users who know what they are looking for; these can be updated and changed with time.

3. General information about NCBI, what they do, and their mandate is interesting for novice users; for expert users, this clutters the site
4. NCBI news and newsletter is a good idea for expert users and should be detailed on the homepage for this group; it may be a good idea to place it instead of the general NCBI information. However, for novice users, it will just add additional content and scroll down. A good compromise is to replace it with a banner link, and place the content on another page.
5. The explanation of the three information containers on the page are useful for novice users, but may just be adding extra content for expert users. More usability studies would need to confirm this since the containers are not static. Saying that, it may still be useful to have the existing links, but without such detailed explanations.

An ideal design strategy would be to have separate home pages for novice and expert users. The two different home pages would actually be the result of using different types of patterns for each group of users. An example of this kind of site can be found at www.cinemasguzzo.com (accessed July 2003), where the home page and *look and feel* of the site changes depending on the user group. In our example, this could be achieved by adding a separate page where users can distinguish if they are novice or expert users. Alternatively, a novice user page could be automatically loaded, with a link to the expert page, or vice versa. One other solution could be to add a server-side counter that looks into your cookies to see how frequently you have visited the site, and based on how frequently you have visited, the website will come up with the appropriate page. All these solutions may be ideal for the user community, however they are not all that practical for such a large and complex website, with such a complicated site architecture. It therefore makes more sense to settle for a compromise, and try to make the site usable for both types of users.

Figure 4-5 illustrates the skeleton of the NCBI home page resulting from the redesign exercise that I performed, by using pattern-oriented design. All the patterns are from the UPADE web language, with the exception of the Disclaimer and Teaser Menu patterns [Welie 2003].

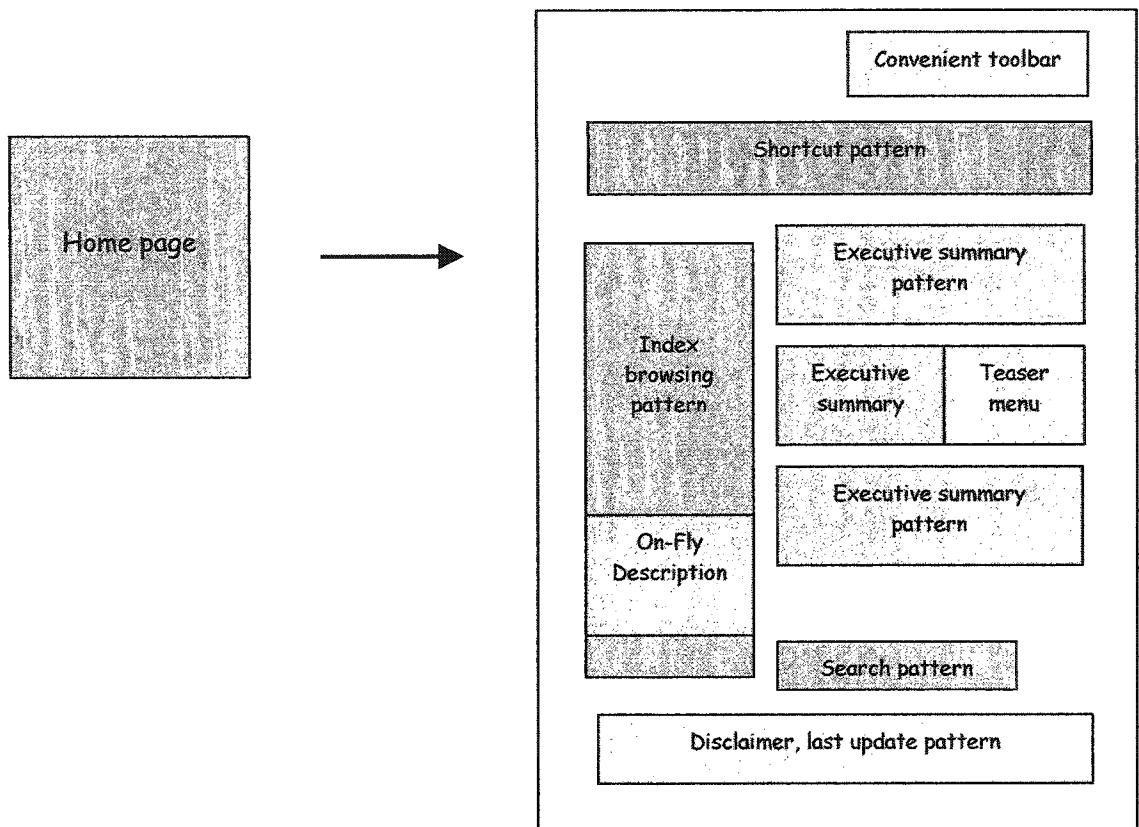


Figure 4-5: Outline of Redesigned NCBI Home page

The image below (figure 4-6) is a screen capture of the new home page that I designed for the NCBI site, taking into consideration all patterns described in the home page skeleton above. It is important to note that from the usability evaluations, it was apparent that users are attached to the site, and therefore it was important not to change the *look and feel* during design.

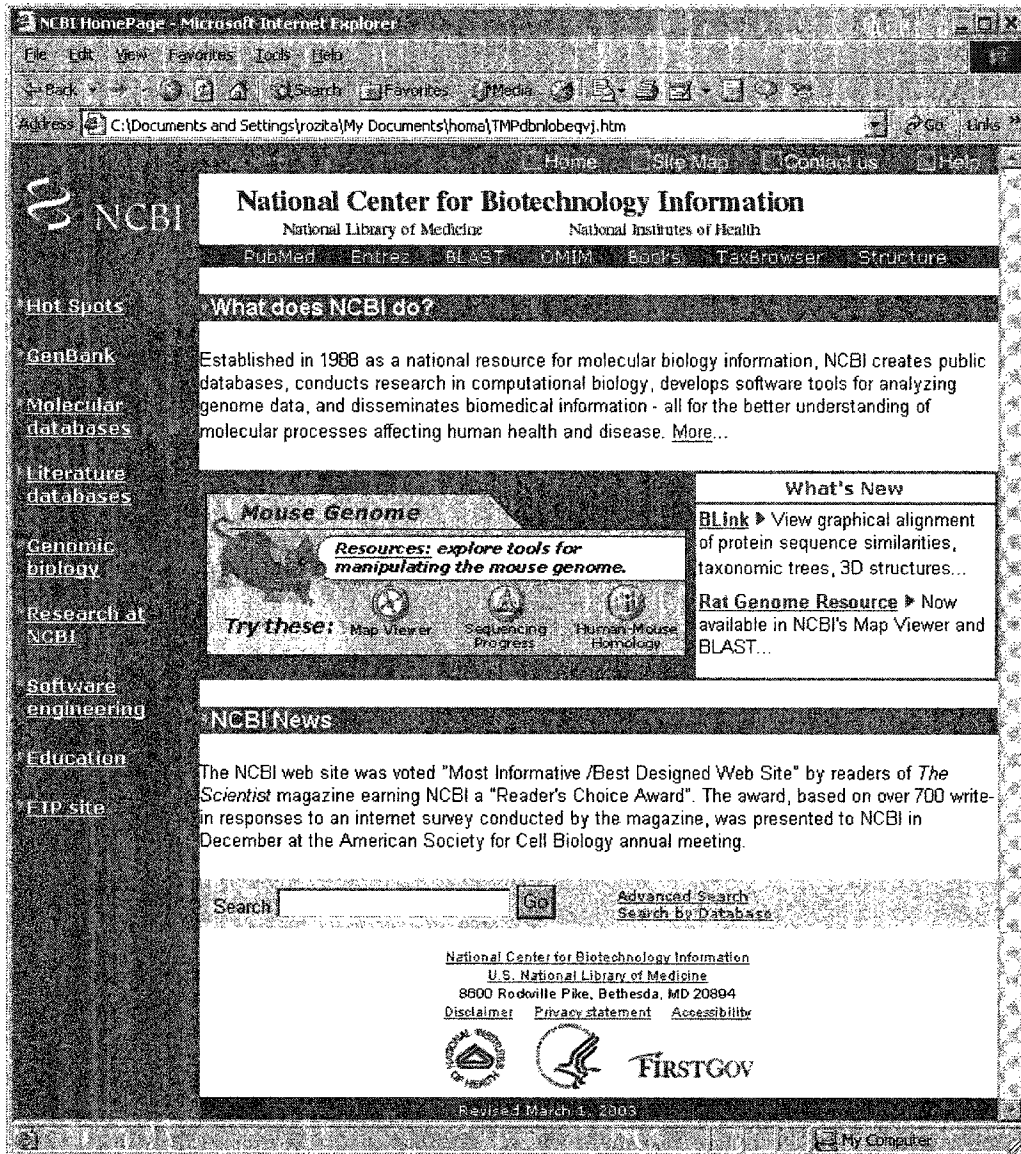
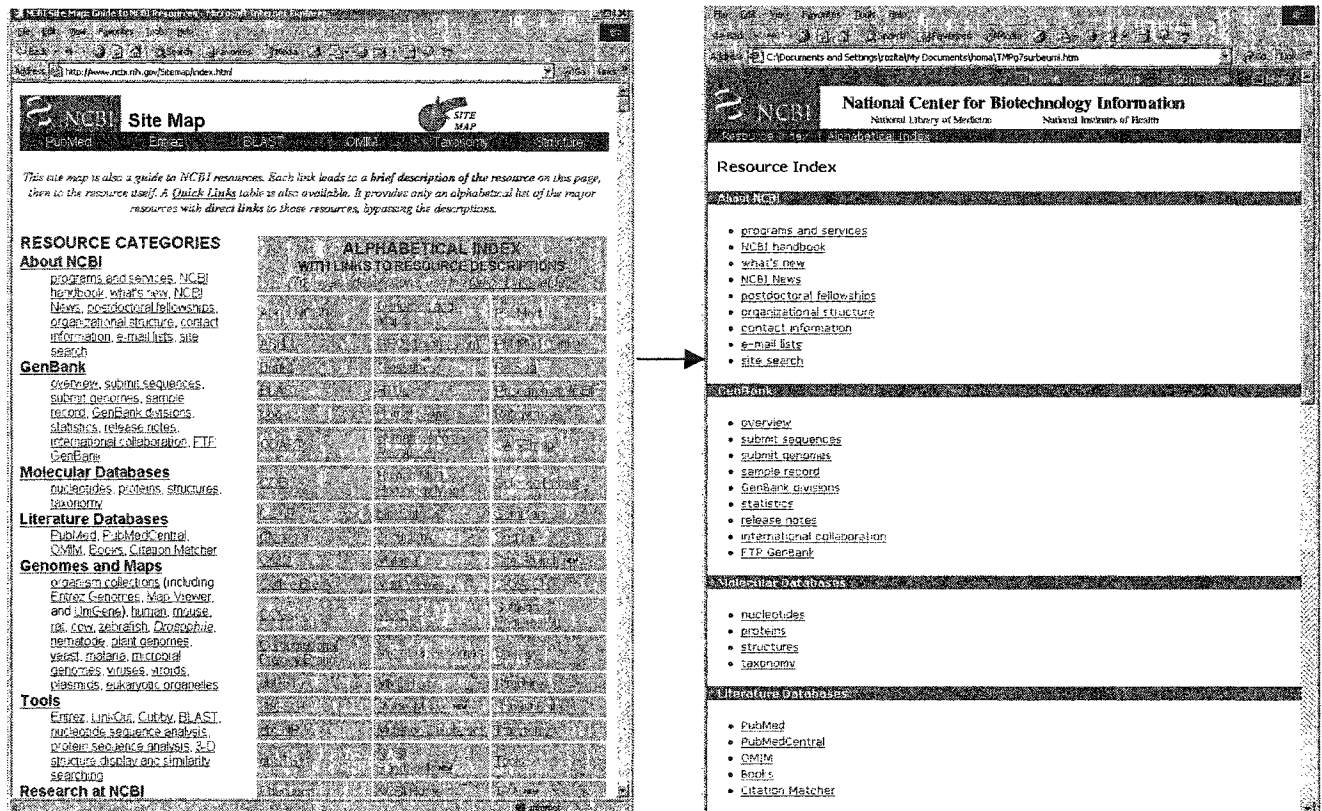


Figure 4-6: Redesigned Home page of NCBI Site

The **site map** was also redesigned, with two different views: Resource Index and Alphabetical Index. The site map pattern was used as a guide. The current site map was excessively long and complicated, and as mentioned by our UI experts in the heuristic evaluations, the site map should include only the basic site structure. The alphabetical index is an excellent idea, but the 3-column format makes it difficult to use. The Resource category structure uses paragraph format with several items on one line and some items straddling two lines. This makes it difficult to scan vertically to find a target,

which is the main use of this type of list. The list items should be in bullet format, one item per line. For new users, the site map is way too crowded and does not provide much help. In summary, the site map is designed to give users who know what they are looking for, fast access to a certain sub-site; and for new users, additional help in locating a page or topic. The site map is complicated, and difficult to use for either of these groups of users. The redesigned site map is illustrated in figure 4-7.



(a) Original Site Map

(b) Redesigned Site Map

Figure 4-7: Redesigned Site Map of NCBI Site

In this chapter, I wanted to demonstrate the potential of patterns in the redesign of existing systems. The same pattern-oriented design approach can be applied to the redesign of other site pages, including:

- Other central pages of a site from which all other pages can be reached (directly or indirectly). The home page is a specialization of such a page. For a large website, we can have more than one central page (e.g. for an academic institution; university, department, research group, personal web sites).
- Navigation pages for directing the user to the proper area of the site containing the information they are seeking.
- Content pages provide the information users are seeking when they visit a site. They may also contain navigational links to give users a sense of location within the site and allow them to progress to more information or return to a previous page.
- Input page (transaction forms, search, feedback) to collect information from users or establish a dialog with the user.
- Utility pages such as help, archive, configuration information.

In addition, if we are sensitized towards the needs of users, the architecture and site structure of a site may need to be redesigned. However, this is a very large feat for a well established website, and difficult to put into practice.

Future directions for the study described in this chapter include a detailed re-evaluation of the new NCBI home page and site map to assess user acceptance. Due to resource constraints, I was unable to perform a more thorough re-evaluation, but as a preliminary step, I carried out informal ethnographic interviews with 3 of the original participants. The main comments from these users were: (1) Home page is much clearer and there is less information overload, (2) Home page no longer requires excessive scrolling, (3) Site map is better organized but has too much white space.

5. Migrating User Interfaces across Platforms Using Patterns¹

A major milestone in interactive system evolution was the shift from text-based interfaces to more complex graphical user interfaces. The web, as a vital medium for information transfer, has also had a major impact on UI design. It emphasized the need for more usable interfaces that are accessible by a wider range of people. Recently, the introduction of new platforms and devices has added an extra layer of complexity to UI system changes. In the migration of interactive systems to these new platforms and architectures, modifications have to be made to the UI while ensuring the application of best design practices. I will demonstrate how this can be achieved through the use of HCI Patterns.

I will be introducing a new term called Multiple User Interface (MUI), which refers to an interactive system that provides access to information and services using different computing platforms. A computing platform is a combination of computer hardware, an operating system and a UI toolkit. Computing platforms include the large variety of traditional office desktops, laptops, palmtops, mobile telephones, personal digital assistants (PDAs), as well as new devices such as interactive television. Conceptually speaking, a MUI provides multiple views of the same information on these different platforms and coordinates the services provided to a single user or a group of users. Each view should take into account the specific capabilities and constraints of the device while maintaining cross-platform consistency and universal usability.

So far in this thesis, I have addressed design and redesign for web applications in a specific context of use, as well as trying to redesign a system for better usability. Part of the context of use is the environment, which was static in terms of the platform in previous examples, i.e. desktop. However, due to the emergence of MUIs, we should see how patterns could be applied to these new environments. Pattern languages should be flexible, and demonstrate that they can adapt to different devices.

¹ This study fits into a broader research project called *Patterns in Multiple User Interface Reengineering* and includes contributions from [Javahery et al. 2003].

Similar to the previous chapter, this chapter also deals with redesign using patterns, but in the context of migrating existing user interfaces between platforms. I will consider two migration methods: Redesign and Reengineering. In redesign, the source interface is migrated to a new platform by directly transforming the HCI patterns used in the interface. Reengineering adds an intermediate step of reverse engineering the source design into a set of abstract HCI patterns and design strategies, from which multiple platform-specific patterns can be inferred. Each approach has its strengths and weaknesses. This chapter focuses in detail on the redesign approach, and offers a prospective outlook on the reengineering approach.

5.1. Overview of Multiple User Interfaces

The concept of MUI was introduced during the HCI-IHM workshop in 2001. A multiple-user interface can be described as follows:

- Allows a single or a group of users to interact with the server-side services and information using different interaction/UI styles. For example, pen and gestures on a PDA, function keys or single characters on a mobile phone, and a mouse for a desktop computer.
- Allows an individual or a group to achieve a sequence of interrelated tasks using different devices.
- Presents features and information that behave similarly across platforms, although some differences may exist. In addition, each platform may have its specific *look-and-feel*.
- Feels like a variation of a single interface, for different devices with the same capabilities. The MUI provides multiple views of the same model, which may reside in a single information repository, or may be distributed among independent systems.

Figure 5-1 shows a MUI to the same Internet financial management system. This interface consists of three views: Desktop, PDA with keyboard, and mobile phone.

Ideally, from the user perspective, these three different interfaces should be as similar as possible. However, this is not realistic because of the capabilities and constraints imposed by each platform. Therefore, the MUI can be seen as a compromise between customized and platform-dependent UIs. The effort required to keep all interfaces consistent and to maintain them increases linearly with the number of interfaces, as the functionality of the underlying financial system is expanding.

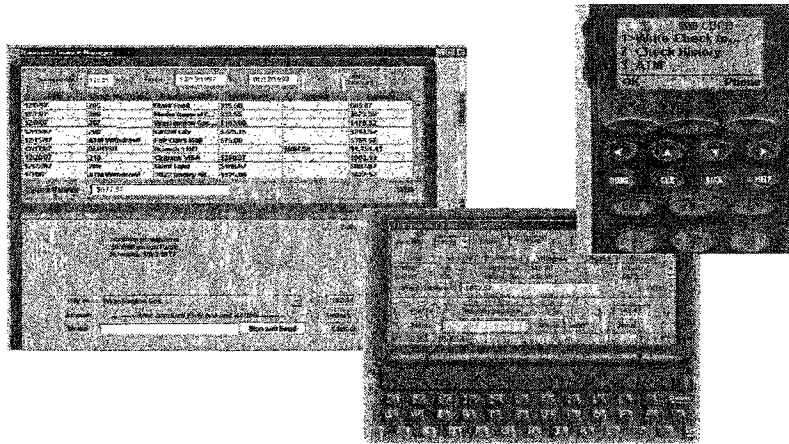


Figure 5-1: An example of a MUI

The following is a scenario that further clarifies the MUI concept and its use:

“You are riding in a car with your colleague who is driving. Suddenly, your **mobile phone** comes on, asking if you can take a video conference call from your team in Canada to discuss a project on which you are working. You take the call and as you communicate with them via the integrated **car video system**, you find out that they need one of the spreadsheets you have saved on your **laptop**, which is in the trunk of the car. Using your wireless technology, you are able to transfer the file to your **PDA**, and then send it to your team. A few minutes later your team will receive and open your spreadsheet, using an **office desktop** and you can start discussing options with them once again via your **interactive television** from your comfortable home.” This scenario is based on [Ghani 2001].

Olsen [2000], Johnson [1998] and Brewster et al. [1998] highlight the design challenges associated with the small screen size of hand-held devices. In comparison to desktop computers, hand-held devices always suffer from a lack of screen real estate, so new metaphors of interaction have to be invented for such devices. Many assumptions that have been held up to now about classical stationary applications are no longer valid for hand-held devices due to the wide range of possibilities currently available. This is because hand-held devices have constantly updated capabilities, exploit additional features of novel generations of networks, and are often enabled for mobile users with varying profiles.

As a starting point, let us take the example of web applications, which are usually designed for a standard desktop computer with a web browser. With the rapid shift toward wireless computing, these web applications need to be customized and migrated to different devices with different capabilities. We need to rethink the **strategies** for displaying information in the context of devices with smaller and lower-resolution screens. As an illustration, an airline reservation system might separate the tasks of choosing a flight and buying the ticket into two separate screens for a small PDA. However, this separation is not required for a large screen. Furthermore, the PDA interface might eliminate images or it might show them in black-and-white. Similarly, text might be abbreviated on a small display, although it should be possible to retrieve the full text through a standardized command. For all these situations, HCI patterns facilitate the transition to different devices while ensuring that constraints are taken into account, and that usability is not compromised.

Figure 5-2 illustrates how HCI patterns can be applied to display the CNN site (www.cnn.com) on different devices. Although the basic functionality and information content are the same for all three platforms (desktop, PDA, and mobile phone), they have been adapted according to the context of use and the limitations of each platform. Depending on the device and the constraints imposed, the presentation of the site will be different. HCI patterns help designers choose appropriate presentations for the design of

each interface. In Figure 5-2, different patterns are used to address the same navigation problem, which is how to assist the user in reaching specific and frequently-visited pages.

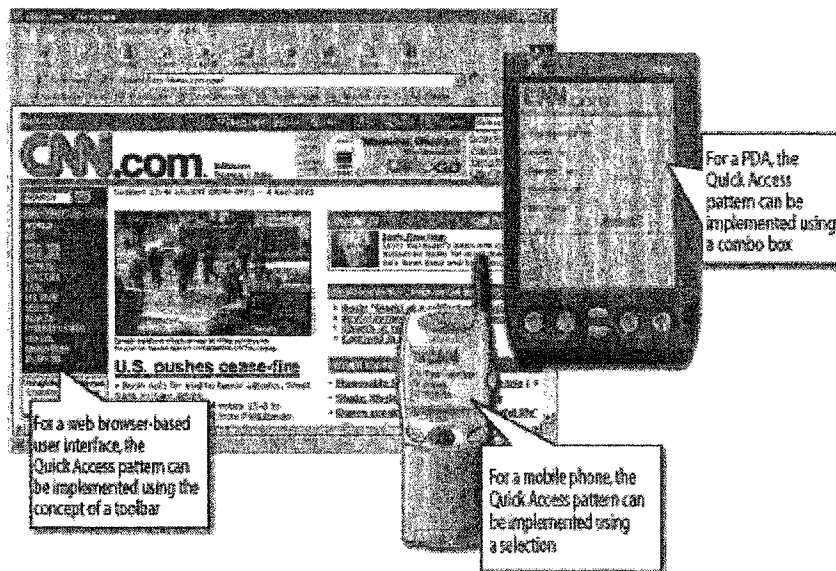


Figure 5-2: HCI Patterns in a MUI Framework

In this chapter, I will address already-existing UIs that require migration to different platforms. As mentioned previously, either Redesign or Reengineering can be used when migrating UIs to other platforms. The emphasis of this chapter is not on the differences between the methods, but rather on how both methods can benefit from the use of HCI patterns:

- Reengineering is a technique that reuses the original system with the goal of maintaining it and adapting it to required changes. It has a fundamental goal of preserving the knowledge contents of the original system through the process of evolving it to its new state. In the process of concretely applying the reengineering, HCI patterns can be used to abstract and redeploy the UI onto different platforms. Reengineering in itself is a complex undertaking, and therefore tools are needed to support the transition so as to limit time and costs.
- Redesign is a simplified version of reengineering, and can be more practical than reengineering in certain contexts. Redesign using patterns involves a direct

transformation of patterns, as an example, from a desktop-based set of HCI patterns to a PDA-based set of patterns. In contrast to reengineering, there is no intermediate step of creating a *platform-independent UI model*. The consequences of this simplification are described later in this chapter.

Remarkably, although research on multiple views and multi-device or multi-user interaction can be traced back to the early 1980s, there are relatively few examples of successful implementations [Grudin 1994]. Perhaps the main cause of this poor success rate is the difficulty of integrating the overwhelming number of technological, psychological, and sociological factors that affect MUI usability into a single unified design.

In the evolution of user interfaces, a *multi-user* interface has been introduced to support groups of devices and people cooperating through the computer medium [Grudin 1994]. A single user in the context of a MUI is what a group of users is for a multi-user interface. The user is asynchronously collaborating with himself/herself. Even if the user is physically the same person, he/she can have different characteristics while working with different devices. For example, a mobile user is continuously in a rush, impatient, and unable to wait [Ramsay and Nielsen 2000]. This user needs immediate, quick, short and concise feedback. The same user, while in the office, can afford to wait a few seconds more for further details and explanations.

5.2. Motivations for using HCI Patterns with MUIs

The following are the motivations for using patterns as a **tool** for redesigning or reengineering an existing user interface:

First, there exist a number of HCI pattern catalogues that carry a significant amount of reusable design knowledge. The use of HCI patterns can facilitate MUI development while increasing their usability. Many groups and individuals have devoted themselves to the development of HCI pattern catalogues and languages, as described earlier [Tidwell 2002; Coram and Lee 1998; Welie 2003]. Some suggest a classification of their pattern

catalogues according to the type of application [Tidwell 2002], while others tailor their catalogue of patterns for a specific platform [Welie 2003]. In addition, Mahemoff and Johnston [1999] propose the *Planet Pattern Language* for internationalizing interactive systems. This language addresses the high-level issues that developers encounter when specifying requirements for international software. It helps developers document and access information about target cultures and shows them how these resources can help them to customize functionality and user interface design.

Secondly, HCI patterns have the potential to drive the entire UI design process [Borchers 2000; Lafreniere and Granlund 1999; Javahery and Seffah 2002]. HCI patterns deal with all types of issues relating to the interaction between humans and computers, and apply to different levels of abstraction. Depending on the type of application, they can be categorized according to different UI facets; such as Navigation, Information/Content, and Interaction (which includes forms and other input components) for web applications. For software developers unfamiliar with newly emerging platforms, patterns provide a thorough understanding of context of use and examples that show how the pattern applies to different types of applications and devices. Some researchers have also suggested adding implementation strategies and information on how a pattern works, why it works (rationale), and how it should be coded [Javahery and Seffah 2002; Welie et al. 2000].

Thirdly, HCI patterns are an interesting reengineering tool because the same pattern can be implemented differently on various platforms. For example, the Quick Access pattern in our Figure 5-2 helps the user reach specific pages, which reflect important website content, from any location on the site. For our news example, it can provide direct and quick access to central pages such as *Top Stories*, *News*, *Sports*, and *Business*. Table 5-1 illustrates the description of this pattern. For a web browser on a desktop, it is implemented as an *index browsing toolbar* using embedded scripts or a Java applet in HTML. For a PDA, the Quick Access pattern can be implemented as a *combo box* using the Wireless Markup Language (WML). For a mobile phone, the Quick Access pattern is implemented as a *selection* [Welie 2003] using WML. Pattern descriptions should

provide advice to pattern users for selecting the most suitable implementation for a given context.

Table 5-1: A Description of the Quick Access Pattern

Pattern Name	Quick access
Type	Navigation support in small and medium Web sites
Context of use	<ul style="list-style-type: none"> • Useful for the novice and expert user • Assist the user to reach specific pages from any page and at anytime • Menu to reflect important website content • Present visible and structured items and sub-items to the user
Consequences	<ul style="list-style-type: none"> • Easy and prompt navigation amongst menu items • Decreases memory (cognitive) load • Increases Web page accessibility • Increases subjective user satisfaction and trust
Solution	<ul style="list-style-type: none"> • Group most important website content links as a menu, such as <i>Top Stories, News, Sports</i>, etc. for a News site • Order of index may be based on a ranking system; should be visible to user • Use meaningful metaphors and accurate phrases as labels • Place it consistently throughout the whole website
Implementation strategy	<ul style="list-style-type: none"> • Implemented as a GUI toolbar (index browsing) for traditional desktop applications, such as a vertical menu on the left • Implemented as a combo-box or a pop-up menu for small size screens such as PDA and some mobile phones (depends on screen size and device capabilities) • Implemented as a selection for mobile phones

5.3. Redesigning User Interfaces with Pattern Mapping

As illustrated in figure 5-3, using the traditional GUI as a starting point, it is possible to redesign the UI for platform migration, by using what I call *pattern mapping*. The patterns of the existing GUI are transformed or replaced in order to redesign and re-implement the user interface. Since patterns hold information about design solutions and

context of use, platform capabilities and constraints are implicitly addressed in the transformed patterns.

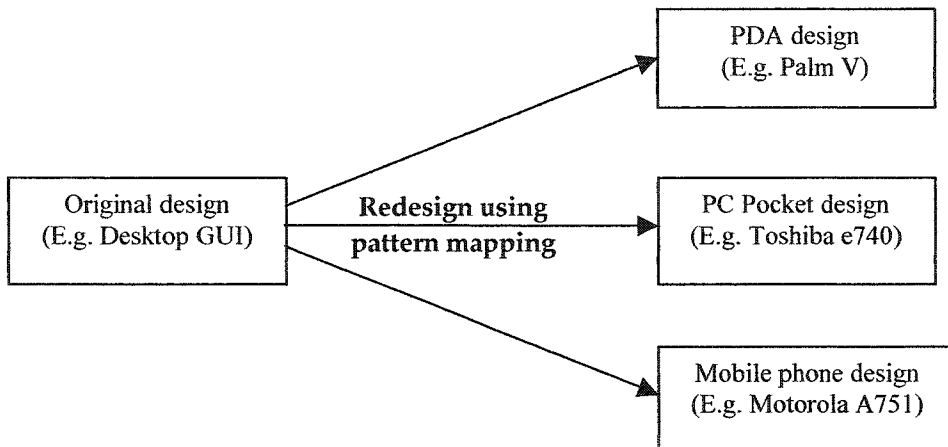


Figure 5-3: Redesigning the UI with Pattern Mapping

To illustrate the use of patterns in the redesign process, in what follows, I will describe the fundamentals of the pattern-based redesign process as illustrated above.

5.4. The Effect of Screen Size on Redesign

Different platforms use different screen sizes, and these different screen sizes afford different types and variants of patterns. In this section I will address how the change in screen size between two platforms affects redesign at the pattern level. I will focus on the redesign of desktop architectures to PDA architectures, as a function of their difference in screen size. This section provides a framework for the redesign of navigation architectures at the presentation layer of design. The amount of information that can be displayed on a given platform screen is determined by a combination of area and number of pixels, as illustrated in Table 5-2.

Table 5-2: Screen Size of PDAs and Desktops

Device	Screen Area (cm ²)	Pixels
Standard PDA	35– 45	25,000 – 100,000
Standard desktop computer	600 – 900	480,000 – 786,000

Comparing area, a standard desktop monitor offers approximately 20 times the area of a typical PDA. If we compare pixels, the same desktop monitor has approximately 10 times the pixels of the PDA.

The total difference in information capacity between platforms will be somewhere between these two measures of 20 times the area and 10 times the pixels. We can conclude that to transform a desktop display architecture to a PDA display architecture, the options are as follows:

1. To reduce architecture size, it is necessary to significantly reduce both the number of pages *and* the quantity of information per page.
2. To hold constant the architecture size (i.e. topics or pages), it is necessary to significantly reduce the quantity of information per page (by a factor of about 10 to 20).
3. To retain the full amount of information in the desktop architecture, it is necessary to significantly increase the size of the architecture, since the PDA can hold less information per page.

The choice of transformation strategy will depend on the size of the larger architecture and the value of the information:

- For small desktop architectures, the design strategy can be weighted either toward reducing information if the information is not important, or toward increasing the number of pages if the information is important.
- For medium or large desktop architectures, it is necessary to weight the design strategy heavily toward reducing the quantity of information, since otherwise the architecture size and number of levels would rapidly explode out of control.

Finally, we can consider transformation of patterns and graphical objects in the context of

the amount of change that must be applied to the desktop design or architecture to fit it into a PDA format. The list is ordered from the most direct to the least direct transformation:

1. **Identical.** For example, drop-down menus can usually be copied without transformation from a desktop to a PDA.
2. **Scalable** changes to the size of the original design or to the number of items in the original design. For example, a long horizontal menu can be adapted to a PDA by reducing the number of menu elements.
3. **Multiple** of the original design, either simultaneously or sequentially in time. For example, a single long menu can be transformed into a series of shorter menus.
4. **Fundamental** change to the nature of the original design. For example, permanent left-hand vertical menus are useful on desktop displays but are not practical on most PDAs. In transformation to a PDA, left-hand menus normally need to be replaced with an alternative such as a drop-down menu.

This taxonomy of transformation types is especially relevant to the automation of cross-platform design transformation since the designs that are easiest to transform are those that require the least transformation. The taxonomy therefore identifies where human intervention will be needed for design decisions in the transformation process. In addition, when building a desktop design for which a PDA version is also planned, the taxonomy indicates which patterns to use in the desktop design to allow easy transformation to the PDA design.

5.5. Pattern-based Redesign: A Case Study with Navigation Patterns

In this section, I will discuss the use of patterns in design transformations from desktop to PDA platforms. The method transforms a core set of patterns based on screen size.

For this case study, I will consider transformations for the following patterns for navigation (Table 5-3). This list is far from exhaustive, but helps to communicate the

flavour and abstraction level of patterns for navigation that we are targeting. Due to space limitations, I can only provide the title and a brief description, rather than the full description format as described in [Borchers 2001].

Table 5-3: Examples of HCI patterns

	HCI Pattern	Definition or comments
P.1	Bread crumbs	Navigation trail from home page down to current page; see [Welie 2003].
P.2	Temporary horizontal menu bar at top	Displayed in a specific context (not permanent). Typically called up by an item in a left-hand vertical menu.
P.3	Temporary (contextual) vertical menu at right in content zone	Called up by a higher-level menu or a link. Might be permanent on a single page, but not repeated across the site.
P.4	Information portal	Broad first and second level on home page. Same principle as the "Directory" pattern [Welie 2003].
P.5	Permanent horizontal menu bar at top	Standard, single-row menu bar
P.6	Permanent vertical menu at left	Vertical menu repeated across all pages of a site. Can have one or multiple levels of embedding.
P.7	Progressive filtering	Allows user to reach target by applying sequential filters [Welie 2003].
P.8	Shallow embedded vertical menu	A single-level menu or a 2-level embedded menu
P.9	Sub-site	Shallow main menu or broad portal leading to smaller sub-sites with simple navigation architectures
P.10	Container navigation	Different levels of menu displayed simultaneously in separate zones (e.g. Outlook Express or Netscape Mail)
P.11	Deeply embedded vertical menu	E.g. file manager menu
P.12	Alphabetical index	Index contains hyperlinks to pages containing or describing the indexed terms
P.13	Key-word search	Search engine
P.14	Intelligent agent	Human-machine interfaces that aim to improve the efficiency, effectiveness and naturalness of human-machine interaction by representing, reasoning and acting on models of the user, domain, task, discourse and media
P.15	Drop-down menu	A menu of commands or options that appears when the user selects an item with a mouse
P.16	Hybrid navigation	Start with key-word search, then present details of search target in menu format

Figure 5-4 illustrates some of the navigation patterns from Table 5-3 as used in the home page of a desktop-based Web portal. Once these patterns are extracted from the desktop-based architecture, they can be transformed and re-applied in a PDA architecture.

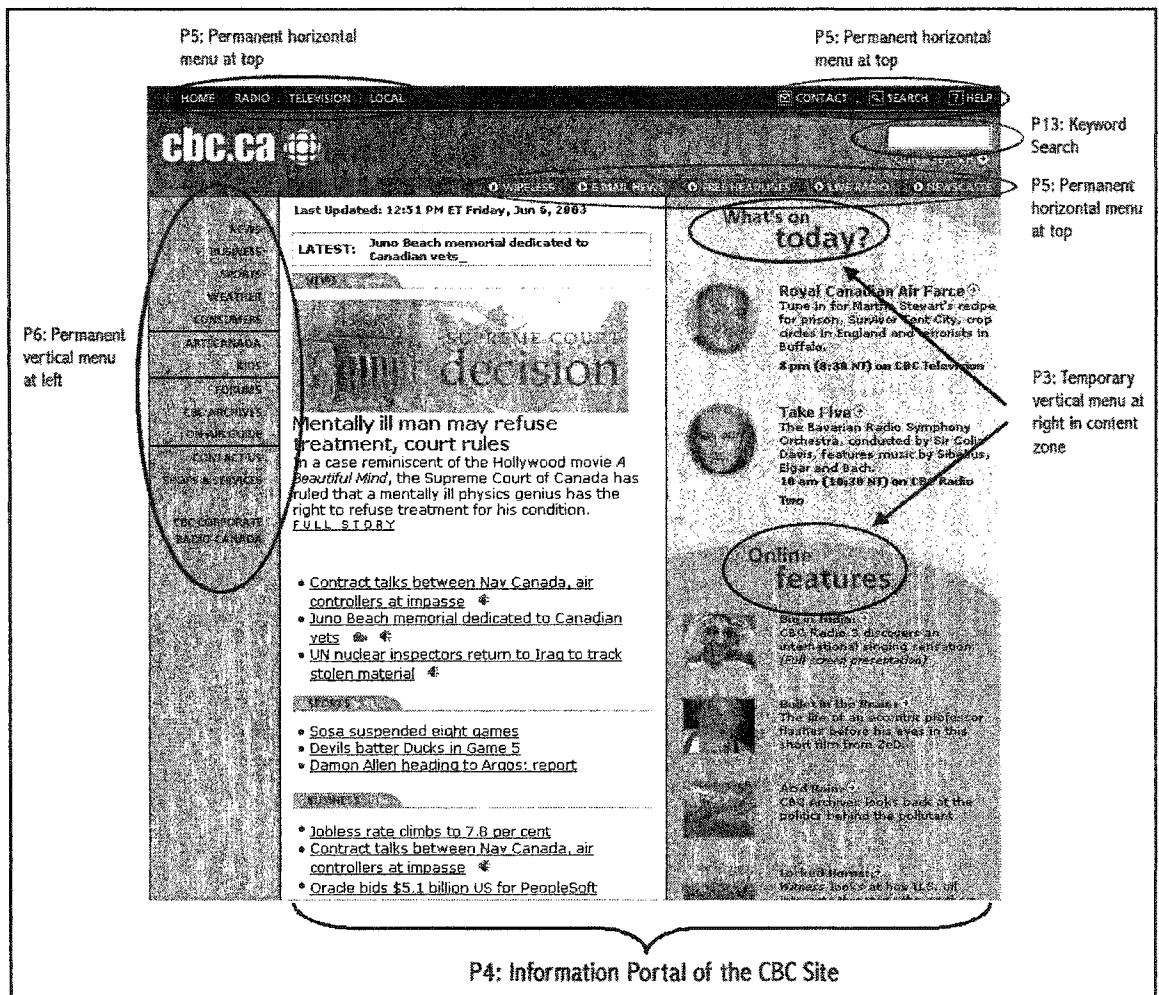


Figure 5-4: Patterns Extracted from the CBC News site

Table 5-4 describes the types of cross-platform transformations that are recommended for the HCI patterns in Table 5-3, and which can be used to redesign the CBC News site. These transformations offer the closest and simplest equivalent in the corresponding platform. In the third column, the suffix “s” after a pattern indicates “scaled (down)”, and the suffix “m” indicates “multiple (sequence)”.

Table 5-4: Examples of HCI Pattern Transformations for Different Screen Sizes

HCI pattern in desktop display	Type of transformation	Replacement pattern in small PDA display
P.1 Bread crumbs	Scalable or fundamental	P.1s - Shorter bread crumb trail; P.15 - Drop-down "History" menu.
P.2 Temporary horizontal menu	Scalable or fundamental	P.2s - Shorter menu; P.6 - Link to full-page display of menu options ordered vertically
P.3 Temporary vertical menu in content zone	Identical, scalable or fundamental	P.6 - Temporary vertical menu in content zone; P.6s - Shorter temporary vertical menu; or P.15 - Drop-down menu
P.4 Information portal	Scalable	P.4s - Smaller information portal
P.5 Permanent horizontal menu at top	Scalable or fundamental	P.5s - Shorter horizontal menu at top; P.6 - Link to full-page display of menu options ordered vertically
P.6 Permanent vertical menu at left	Fundamental	P.15 - Drop-down menu
P.7 Progressive filtering	Identical	P.7 - Progressive filtering
P.8 Shallow embedded vertical menus	Identical or fundamental	P.6m - Sequence of temporary vertical menus in content zone; P.8 - Shallow embedded vertical menus
P.9 Sub-site	Scalable or fundamental	P.6 - Temporary vertical menu in content zone; P.9s - Smaller sub-site;
P.10 Container navigation (3 containers)	Scalable or fundamental	P.10s - Container navigation (2 containers); P.7 - Progressive filtering
P.11 Deeply embedded vertical menus	Multiple or fundamental	P.6m - Sequence of single-level menus; P.8m - Sequence of shallow embedded menus
P.12 Alphabetical index	Scalable	P.12s - Alphabetical index (less items per page, or smaller index)
P.13 Key-word search	Identical	P.13 - Key-word search
P.14 Intelligent agents	Identical	P.14 - Intelligent agents
P.15 Drop-down menu	Identical, scalable or fundamental	P.15 - Drop-down menu; P.15s - Shorter drop-down menu; Hyperlink to P.6 - Temporary vertical menu in content zone
P.16 Hybrid navigation: Key-word search	Identical or scalable	P.16s - Hybrid approach with smaller or less deeply embedded menus

Figure 5-5 demonstrates the redesigned interface of the CBC site for migrating to a PDA platform. The permanent horizontal menus at the top (P5) in the original desktop UI were redesigned to a shorter horizontal menu (P5s). In order to accommodate this change on the small PDA screen, the three different horizontal menus had to be shortened, and only important navigation items were used. The keyword search pattern (P13) remains as a keyword search. The permanent vertical menu at the left (P6) is redesigned to a drop-

down menu (P15). The drop-down menu in the PDA design also includes the menu headings, “What’s on today?” and “Online features” from the temporary vertical menu (P3) in the original desktop design. Finally, the information portal (P4), which is the first thing that captures the user’s attention, is redesigned to a smaller information portal (P4s).

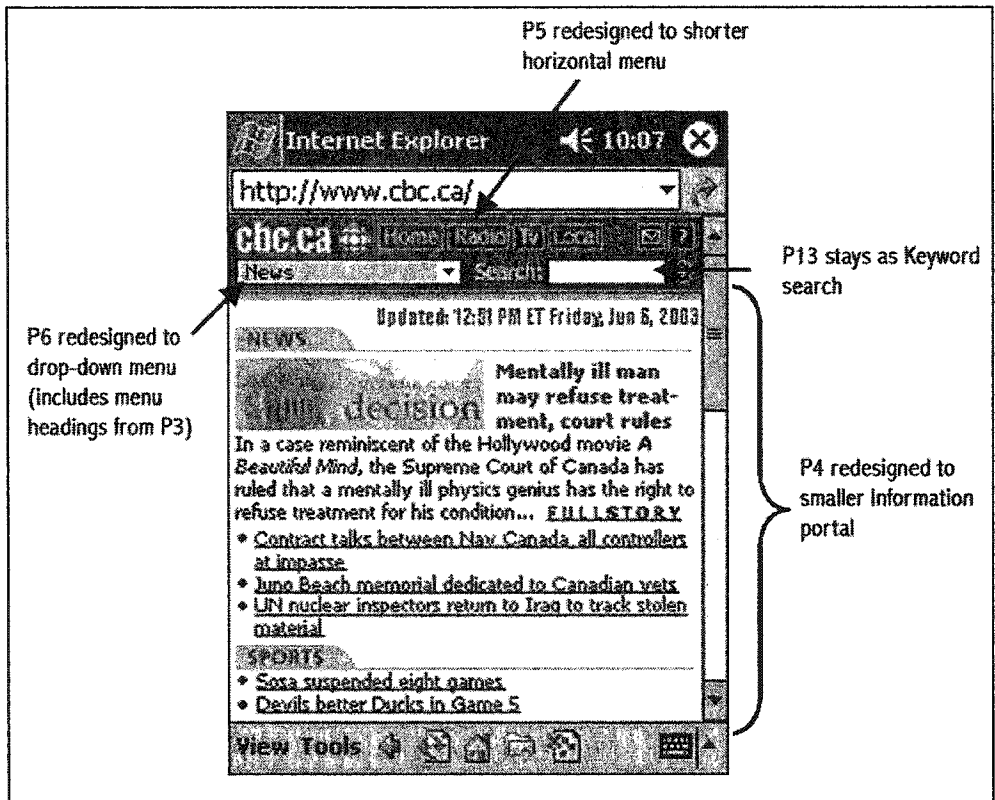


Figure 5-5: Migration of the CBC site to a PDA Platform using Pattern Mapping

Up to this point, I have demonstrated how patterns can be used as a redesign tool for migrating a web portal from a desktop platform to a PDA. The problem with redesign, however, is that the same exercise has to be repeated for each platform. If we were to have a generic UI model of the application or web site, migrating to different platforms would be facilitated. In such a case, design strategies and content-related information would be separate from presentation issues. The next section introduces some ideas for using patterns in UI reengineering to try to come up with a UI model, which can then be

instantiated to different platforms. What I propose in the following section is a future perspective on how patterns can be applied to reengineering user interfaces.

5.6. Patterns in Reengineering User Interfaces

Reengineering consists of a reverse engineering phase, a transformation phase, and a forward engineering phase [Moore 1996]. Figure 5-6 illustrates how reengineering can be performed with HCI patterns. The process begins with a user interface (e.g. desktop in Figure 5-4) that is to be reengineered, or migrated to different platforms. The reengineering steps are as follows:

1. Reverse engineering: Consists of a pattern extraction and abstraction phase, resulting in the creation of a platform-independent UI model.
2. Transformation: Patterns and design strategies in the platform-independent UI model are analyzed, and transformed if appropriate.
3. Forward engineering: The platform-independent UI model is first instantiated to different platforms based on constraints and capabilities; the patterns are then implemented on different platforms.

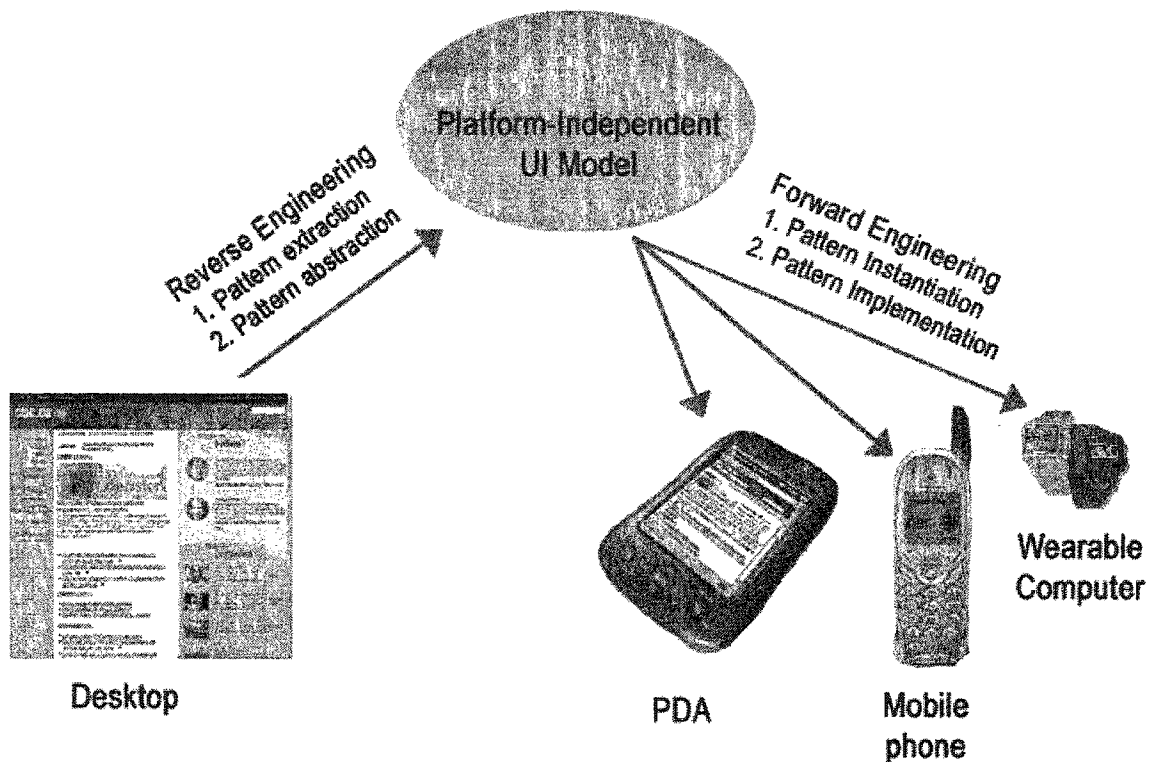


Figure 5-6: Pattern-Assisted UI Reengineering

In what follows, I will further clarify the above points and detail the process of the proposed pattern-assisted reengineering method.

5.7. Pattern-Assisted Reengineering

In pattern-assisted reengineering, we create a *platform-independent UI model* that can be instantiated to different platforms. The main benefit of such a model is that it captures content-related information, design strategies, and context of use attributes, independent of the device. If we want to add features or somehow change the design, this will be reflected on all devices. Applying certain presentation rules and methods which are platform-dependent onto this model results in components suited for defined platforms. Once the platform components are defined, the layout can be defined according to screen size, resolution, and other device-specific constraints. To facilitate the reengineering

process, patterns can be used as a **tool** since they encapsulate design knowledge with different platform-specific implementation schemes. Applying appropriate HCI patterns in reengineering can make the process of design easier, and will result in less usability errors. In addition, since patterns are context-oriented, their use will ensure that the best solution has been applied.

The first step of reverse engineering consists of extracting patterns from the original UI. In the process of pattern extraction, we match this knowledge against known patterns and identify which patterns were used in the original interface. If we take the example of a web portal or any web application, the extracted patterns can be logically grouped into the following descriptive UI facets: (1) Navigation (2) Information or Content (3) Interaction (such as forms and other input components). This step is identical to the pattern extraction step in redesign (Figure 5-4).

During the abstraction step of reverse engineering, extracted patterns are abstracted into higher levels of design concepts and goals, generally referred to as design strategies. The aim is to create a platform-independent UI model that can then be instantiated to different platforms. Other techniques and artefacts such as domain analysis, personae, and use cases can be applied to create a more complete UI model. The platform-independent UI model has the following characteristics: (1) Includes design strategies. An example of a design strategy is query-based navigation versus conceptual model-based navigation. (2) Includes patterns that are abstracted sufficiently to become platform-independent. An example includes the *Quick Access* pattern (Table 5-1). (3) This step clearly separates content and design from presentation issues.

Moore [1996] defines the UI transformation phase as consisting of transforming an “abstract model” into a “restructured abstract model”, with human analyst input. In our approach, during the transformation step, patterns and design strategies in the model are analyzed to determine suitability to any new design requirements. Inappropriate patterns and design strategies are replaced by appropriate patterns, or removed. In addition, new patterns can be added to the model based on user requirements, and task-based changes.

It is important to differentiate between transformation at a higher level of design, which is platform and implementation-independent, and transformation at a lower level of design, which deals with presentation issues. Since our objective is to create a generic UI model, presentation issues and implementation strategies are not taken into account during the transformation phase, but rather, during forward engineering. If we consider the *Quick Access* pattern, a design decision could include the addition of this pattern during the transformation phase. However, presentation and implementation details of this pattern are abstracted away until platform constraints and capabilities are taken into account.

In the first step of forward engineering, the platform-independent UI model is instantiated to target devices based on their constraints and capabilities. Dialogue style, look-and-feel, and presentation issues are considered at this level. Depending on the device, different presentation components (or implementation strategies) may apply for each pattern and design strategy in the UI model. For example, if we go back to the *Quick Access* pattern, and want to apply it to a PDA, the combo box will be used. However, for the PC, the desktop toolbar (index browsing) is the appropriate implementation strategy.

Another example is the *progress pattern*, which can be applied if “the user wants to know whether or not the operation is still being performed as well as how much longer the user will need to wait” [Welie 2003]. The instantiation of this pattern for a desktop application can include parameters such as the estimated time left, the transfer rate, and the progress bar. A mobile phone may use the same process, but some parameters have to be left out due to platform constraints. In such a case, only the percentage of time left to complete the task may be displayed.

A final example is the *search pattern*. On the PC, this pattern can be instantiated using a complex dialogue box including different text fields, checkboxes and radio buttons. Whereas on the PDA, a simple search input box is appropriate. Simple search can be integrated in most devices since it does not need much space. Having rules that highlight such details can make the process of MUI migration easier.

Another relevant consideration for MUI migration is the capability of previewing images. For instance, when running an application on a mobile device, such as a mobile telephone, we may not have the ability to preview images due to the platform constraints of smaller screen size and lower resolution. The *Preview pattern* is appropriate for the desktop, but not for the mobile phone, and should therefore not be implemented during forward engineering.

The second step of forward engineering is pattern implementation. In this step, instantiated patterns for the target device are applied, combined and coded, resulting in the new UI.

The new UI will be better suited to new requirements, since patterns are context-oriented. Pattern-assisted reengineering simplifies the process of reengineering the UI for a new context of use. However, this method does not cover all aspects and is not complete. Currently, we are far from the point where defined patterns cover all possible situations and where a model can be created purely by combining existing patterns.

5.8. Comparing Reengineering to Redesign

In contrast to redesign, which was introduced in Section 5-3, reengineering is useful for the following reasons:

1. By creating a platform-independent UI model, reengineering facilitates forward engineering to a broad family of different platforms. The platform-independent UI model encourages design reuse, reduces the total project workload and ensures coherence between the different members of the application family. In comparison, direct redesign without such a model could be less efficient in the context of a large family of platforms, and does not include safeguards for maintaining coherence between applications.
2. Software systems inevitably change over time. Original requirements have to be modified to account for changing user needs, changes in system environment, as well

as advances in technology. To manage the process of system change, we need to maintain the system [Sommerville 2000]. Reengineering is a technique that explores the idea of reusing the original system with the goal of maintaining it and adapting it to required changes. The creation of a platform-independent UI model can facilitate future maintenance, since elements of the model can be changed, rather than the design of each specific platform.

3. In the reverse engineering phase, the process of reengineering allows the designer to re-evaluate the user's task-goals in a broader context of use. This re-evaluation can improve the usability and utility of the system. In contrast, direct redesign does not question the task goals, and therefore can result in repeating old mistakes or missing opportunities for optimization.

Although reengineering is advantageous as outlined in the above points, it has a number of disadvantages:

1. It is time-consuming and complex in today's context of rapid software development.
2. The lack of a clearly-defined pattern taxonomy can be problematic during the abstraction stages and description of the UI model.
3. The forward engineering phase requires a set of defined rules with regard to implementation strategies of each pattern, depending on the device constraints and capabilities.

5.9. Research Directions

In the current technological context, required changes to already-existing UIs are inevitable, such as migrating applications to multiple platforms. Writing code from scratch is no longer a viable solution when applying changes since it requires a large amount of resources, or is simply too risky to perform as the original knowledge may get lost. An interactive system that is up and running is often a fundamental asset to the company using it, as it carries with it a certain amount of domain knowledge and

experience. To manage the process of system change, we need to maintain the system [Sommerville 2000].

To address these issues of reuse and maintenance, this chapter discussed the idea of using HCI patterns as an approach for UI redesign and migration to different platforms. The reengineering ideas presented in this chapter are a starting point and continued research will be needed to validate them. Some work in the area of reengineering patterns has begun [Ismail and Keller 1999; Ducasse et al. 2000; Beedle 1997], however there is still much that needs to be explored, especially in user interface reengineering.

Using patterns can effectively fill the gap in existing methods for migrating UIs since they capture best design practices, and can play a role throughout the complete redesign process. The application of patterns has a number of advantages: First, they can reduce the time required for redesign since for the most common usability and UI design problems, a pattern solution already exists. Secondly, usability errors are reduced since most of the patterns have already been tested on other systems. Finally, patterns help in the comprehension of the system for future maintenance.

In this chapter, I introduced redesign through pattern mapping as a simplified version of reengineering. This approach is a significant improvement over non-structured migration methods currently in use, for the following reasons:

- The method provides a standardized table of pattern transformations, thereby reducing the redesign effort and ensuring consistency in redesign.
- The standardized transformations formalize best practices in design, thereby ensuring optimal quality of the migrated user interface.
- The method helps designers in design choices associated with the size of the source architecture and target architecture. Another relevant consideration could be the amount of information to maintain in migrating from the source platform to the target platform.

- The method is simple enough to be used easily by novice designers, as compared to reengineering which currently requires a considerable degree of expertise and abstract reasoning ability.

As a prospective outlook, I introduced pattern-assisted UI reengineering as an advancement of UI redesign. Similarly to UI redesign, HCI patterns are extracted from the user interface. However, these patterns, which are associated with various UI facets, are then abstracted into a platform-independent UI model. In order to adapt the UI to new platform-specific requirements, the platform-independent model is instantiated for various platforms, by using patterns.

For the purpose of MUI migration, UI reengineering offers the following advantages over UI redesign: (1) By creating a platform-independent UI model, reengineering facilitates forward engineering to a broad family of different platforms. (2) Reengineering reuses the original system with the goal of maintaining it and adapting it to required changes, which facilitates future maintenance. (3) Through reverse engineering the designer can re-evaluate the user's task-goals in a broader context of use. This re-evaluation can improve the usability and utility of the system and avoids repeating old mistakes.

Pattern-assisted reengineering offers the very useful ability of easily extracting multiple platform-specific designs from a single generic (platform-independent) UI model. However, the current state of the art in HCI patterns and MUI research is not yet mature enough to handle all the requirements of pattern-assisted reengineering. Before generic UI pattern-based models can be defined, more research must be addressed to define the multiple levels of abstraction of patterns and to create a clear, well-structured taxonomy of HCI patterns. Thus, within a pattern-based framework, the simplified "redesign" method proposed here is currently the most practical approach for migration of UIs between platforms.

Whether they are used in redesign or in reengineering, HCI patterns facilitate the UI migration process by encapsulating high-level design choices and rationales and allowing the designer to operate at a higher level of abstraction.

6. Conclusion and Future Investigations

Patterns aim to capture and communicate the best practices of user interface design with a focus on the user's experience and the context of use. As a result, they are an attractive UCD tool, with interesting ramifications for designing across a variety of contexts. Compared to the traditional design approach of using guidelines, patterns have emerged as a powerful tool for developing usable systems. Above all, they are a good alternative to guidelines because they are problem-oriented, yet not toolkit-specific. In addition, they are more concrete and easier to use for novice designers, context-oriented, and promote reusability. With patterns, the designer is told when, how and why the solution can be applied; and the solution is related to a specific activity.

In this thesis, I investigated the role of HCI patterns as a design and redesign tool. I wanted to see if patterns were comprehensive and mature enough to be used in practical applications of design, and in different contexts of use. More precisely, the contributions of this thesis are as follows:

1. I demonstrated how a web application can be designed using pattern-oriented design, which involves transferring knowledge gained by usability experts to software engineers through a systematic approach. This approach incorporates patterns and is facilitated by tool support.
2. I demonstrated how, by using pattern-oriented design, we could redesign a complex and user-specific Bioinformatics IBIS. By using empirical analysis, patterns were used to close the gap between user experiences and the features offered by the site. A questionnaire was used to enhance personae and model user interaction with the site. Based on these results, as well as UI expert evaluation, patterns were selected and combined to build a more usable home page and site map. In addition, the process of transforming usability problems into actual design solutions was illustrated.

3. I demonstrated how pattern-oriented design could be applied to the redesign and reengineering of Multiple User Interfaces. In this part, patterns had to be transformed in accordance to the platform constraints and capabilities of the new device. Once these new patterns were described, the existing webpage can be redesigned to fit onto a smaller device.

Our research answered some questions about the role of patterns in design and redesign; however, it also lead to some fundamental issues that need to be further investigated.

First, the validity of patterns needs to be addressed. The HCI pattern community does not have any standards for creating, documenting, and validating patterns. This is imperative since patterns should be validated before taking on such an important role as providing solutions for the design of interactive systems.

Secondly, it is time to conduct a comparative empirical study on a design with and without pattern use. We hope to be able to tackle such an endeavor in the near future with Bioinformatics tools. Such a study is important to further validate the importance and use of patterns in the design of a variety of applications. .

Thirdly, the pattern community is in desperate need of a pattern taxonomy and descriptions of different abstraction levels within patterns. For example, in UI design, there are higher-level patterns and lower-level patterns, as described in the context of MUIs. In our UPADE web language, we tackled abstraction levels to a certain extent.

Finally, patterns can be used to effectively disseminate integration of usability in software engineering. One other idea that I started to investigate is the role of patterns not just in design, but also in other parts of the user-centered lifecycle, namely requirements and testing. I call this idea PSI, or Pattern-Supported Integration, and it stemmed from a tutorial that I helped prepare and present to developers at the Daimler Chrysler Research and Technology Center in Ulm, Germany. The research center's software technology team was interested in integrating User-Centered Design (UCD) practices and techniques

into their organization. One of the key factors for cost-effective integration is the ability of an organization to capture, maintain and disseminate the UCD knowledge and best design practices. Patterns, with the complicity of organizational learning strategies, can facilitate UCD integration, as well as the training of highly skilled practitioners. Therefore, it would be interesting to see how patterns can be used effectively not just in design, but also in all aspects of software development.

References

- Alexander, C. (1964). *Notes on the Synthesis of Form*. Cambridge, Harvard University Press.
- Alexander, C. (1979). *The Timeless Way of Building*. New York, Oxford University Press.
- Alexander, C., Ishikawa, S., Silverstein, M., Jacobson, M., Fiksdahl-King, I., and Angel, S. (1977). *A pattern language - Towns, buildings, construction*. New York, Oxford University Press.
- Ambler, S. (1998). *Process Patterns*. New York, Cambridge University Press.
- Appleton, B. (1997) Patterns For Conducting Process Improvement. In *Proceedings of the 1997 Fourth Conference of Patterns Languages of Program Design*, September 3-5, 1997, Monticello, Illinois.
- Attwood T.K., Parry-Smith D.J. (1999). *Introduction to Bioinformatics*. Essex, Addison Wesley Longman Higher Education.
- Beedle, M. (1997). Pattern based reengineering. *Object Magazine*, 6(11) (January 1997), pp. 56-70.
- Borchers, J.O. (2000). A Pattern Approach to Interaction Design. In *Proceedings of the DIS 2000 International Conference on Designing Interactive Systems*, August 16–19, 2000, New York, ACM Press, pp. 369–378.
- Borchers, J.O. (2001). *A Pattern Approach to Interaction Design*. New York, John Wiley & Sons.
- Brewster, S., Leplâtre, G. and Crease, M. (1998). Using Non-Speech Sounds in Mobile Computing Devices. In *Proceedings of the First Workshop on Human Computer Interaction of Mobile Devices*, May 21-23, 1998, Glasgow, UK.
- Cockburn, A. (1997). Structuring Use Cases with Goals. *Journal of Object-Oriented Programming*, Sept.-Oct. 1997 and Nov.-Dec. 1997 issues.
- Cooper, A. (1999). *The inmates are running the asylum: Why high-tech products drive us crazy and how to restore the sanity*. Indianapolis, SAMS Publishing.

Coplien, J. O. (1995). A development process generative pattern language. In Coplien, J.O. and Schmidt, D.C., eds. *Pattern languages of program design*. Reading, MA, Addison-Wesley, pp.183-237.

Coplien, J.O. (1997). *The OrgPatterns website* [Internet]. Available from <<http://www.bell-labs.com/cgi-user/OrgPatterns/OrgPatterns>> [Accessed January, 2002].

Coram T., and Lee J. (1998). *Experiences: A Pattern Language for User Interface Design* [Internet]. Available from <<http://www.maplefish.com/todd/papers/experiences>> [Accessed January, 2002].

Ducasse, S., Debbe, R. and Richner, T. (2000). *Type-Check Elimination: Two Reengineering Patterns* [Internet]. Available from Bern University, <<http://www.iamunibe.ch/~scg>> [Accessed March, 2002].

Erickson T. (2000). Lingua Francas for Design: Sacred Places and Pattern Languages. In *Proceedings of Designing Interactive Systems*, August 17-19, 2000, New York, ACM Press.

Gaffar A., Sinnig D., Javahery H. and Seffah A. (2003). MOUDIL: A Comprehensive Framework for Disseminating and Sharing HCI Patterns. In *Patterns Forever: A Workshop at the Conference on Human Factors in Computing Systems* (ACM CHI 2003), April 2003, Florida, USA.

Gamma, E., Helm, R., Johnson, R. and Vlissides, J. (1995). *Design Patterns: Elements of Reusable Object-Oriented Software*. Reading, MA, Addison-Wesley.

Ghani, R. (2001). 3G: 2B or not 2B? The potential for 3G and whether it will be used to its full advantage. *IBM Developer Works: Wireless Articlesm*, August 2001.

Grudin, J. (1994). Groupware and Social Dynamics: Eight Challenges for Developers. *Communications of the ACM*, 37(1), pp. 92-105.

Higgins, D. and Taylor, W., eds (2000). *Bioinformatics: Sequence, Structure and databanks*. New York, Oxford University Press.

Ismail K., Keller, R. (1999). *Transformations for Pattern-based Forward Engineering* [Internet]. Available from Université de Montréal, <<http://www.iro.umontreal.ca/~labgelo/Publications/Papers/sts99.pdf>> [Accessed March, 2002].

ISO 9241, Standard on Guidance on Usability, 1991.

ISO 13407, Standard on Human-Centered Development Processes for Interactive Systems, 1998.

ISO 15527, Standard on Human-Centered Software Development Practices, 2001.

- Javahery, H., and Seffah, A. (2002). A Model for Usability Pattern-Oriented Design. In *Proceedings of TAMODIA 2002*, July 18-19 2002, Bucharest, Romania, pp. 104-110.
- Javahery, H., Seffah, A., Engelberg, D. and Sinnig, S. (2003). Chapter 12: Migrating User Interfaces Across Platforms Using HCI Patterns. In Seffah, A. and Javahery, H., eds. *Multiple User Interfaces*. Chichester, UK, John Wiley & Sons, In Press.
- Johnson, P. (1998). Usability and Mobility: Interactions on the move. In *Proceedings of the First Workshop on Human Computer Interaction With Mobile Devices*, May 21-23, 1998, Glasgow, UK.
- Karat, C.M. (1998) Guaranteeing rights for the user. *Communications of the ACM*, 41 (12) (December 1998), pp. 29-31.
- Kirakowski, J., and Corbett, M. (1993). SUMI: The Software Measurement Inventory. *British Journal of Educational Technology*, 24 (5), pp. 210-212.
- Lafrenière, D. and Granlund, A. (1999). *A Pattern-Supported Approach to User Interface Design* [Internet]. Available from UPA 99 Workshop Report, <http://www.gespro.com/lafrenid/Workshop_Report.pdf> [Accessed December, 2002].
- Li, N. (2001). *Usability Patterns-Assisted Design for Web User Interfaces*. Masters Thesis in the Department of Computer Science, Concordia University, Montreal, Canada.
- Mahemoff, M. and Johnston, L. (1999). The Planet Pattern Language for Software Internationalisation. In *Proceedings of Pattern Languages of Program Design (PLOP)*, September 15-18, 1999, Monticello, IL.
- Manns, M.L., Sharp, H., Prieto, M., and McLaughlin, P. (1998). Capturing Successful Practices in OT Education and Training. *Journal of Object Oriented Programming*, 11(1), pp. 29-34.
- McKenzie, K. (2000). *10 Usability Heuristics* [Internet]. Available from <http://www.studiowhiz.com/_publications/10heuristics.php> [Accessed March, 2003].
- Moore, M. (1996). Representation Issues for Reengineering Interactive Systems. *ACM Computing Surveys*, 28 (4) (December 1996).
- Nielsen, J. (1994). Heuristic Evaluation. In Nielsen, J. and Mack, R.L., eds. *Usability inspection methods*. New York, John Wiley & Sons.
- Nielsen, J. (2001). *How to Conduct a Heuristic Evaluation* [Internet]. Available from <http://www.useit.com/papers/heuristic/heuristic_evaluation.html> [Accessed November, 2001].

Olsen, D., Jefferies, S., Nielsen, T. et al. (2000) Cross-modal Interaction using XWeb. In *Proceedings of the 13th Annual ACM Symposium on User Interface Software and Technology* (UIST'2000), November 5-8, 2000, San Diego, USA.

Ramsay M. and Nielsen J. (2000). *WAP Usability Déjà Vu: 1994 All Over Again*. Technical Report from a Field Study in London. Fremont, USA, Nielsen Norman Group.

Seffah, A. and Javahery, H. (2002) On the Usability of Usability Patterns. In *Patterns in Practice: A Workshop for UI Designers at the Conference on Human Factors in Computing Systems* (ACM CHI 2002), April 20-25, 2002, Minneapolis, Minnesota.

Sommerville, I. (2000). *Software Engineering*, 6th Edition. Boston, MA, Addison-Wesley.

Tahir, M. F. (1997). Who's on the other side of your software: Creating user profiles through contextual inquiry. In *Proceedings of Usability Professionals Association Conference* (UPA '97), August, 1997, Monterey, USA.

Tidwell, J. (1999). *Common Ground: A Pattern Language for Human-Computer Interface Design* [Internet]. Available from <http://www.mit.edu/~jtidwell/interaction_patterns.html> [Accessed January, 2002].

Tidwell J. (2002) *UI Patterns and Techniques* [Internet]. Available from <<http://time-tripper.com/uipatterns/index.php>> [Accessed March, 2003].

Weinstein, A. (1998). *Defining Your Market: Winning Strategies for High-Tech, Industrial, and Service Firms*. New York, Haworth Press.

Welie M.V. (2003) *Interaction Design Patterns* [Internet]. Available from <<http://www.welie.com>> [Accessed March, 2003].

Welie, M.V., Van der Veer, G.C. and Eliëns, A. (2000) Patterns as Tools for User Interface Design. In *Proceedings of the International Workshop on Tools for Working with Guidelines*, October 7-8, 2000, Biarritz, France.

Appendix A: Pattern Examples from UPADE Web Language

Architectural Pattern:

Identification	Pattern Name	HIERARCHICAL
	Category	Architectural Patterns → Information Architecture Patterns
Context of Use	User	Novice or Expert
	Task	Tasks are structured hierarchically. All sub-tasks stem from one original center.
	Platform Capabilities and Constraints	<ul style="list-style-type: none"> - Information should be organized in device-independent way - Filtering mechanism is required for mobile and small devices
Usability Problem	<ul style="list-style-type: none"> - The user can easily go through from the most general overview of the Web site, such as the home page, down to the most specific or optional topics. - Much greater flexibility than the sequence structure. 	
Usability Factors	<ul style="list-style-type: none"> - Efficiency, Effectiveness, Satisfaction - Understandability, Completeness, Flexibility 	
Example	<ul style="list-style-type: none"> - Company Website, personal home page, portals 	
Design Solution	<ul style="list-style-type: none"> - All pages are organized in a hierarchical cascade model. The sub-branches expand from one generic center. There is no intersection among them. - Certain constraints should be applied to the structure's width and depth. 	
Implementation	HTML, XML, Traditional Database	
Related Patterns	Superordinate	Composite
	Subordinate	Focus Page, Utility Page, Navigation Page, Tiled Page, Stack Page
	Neighboring	Sequential, Grid
	Competitor	
Additional Reading	<ul style="list-style-type: none"> - "Hierarchy" guideline in the Yale guidelines. - "Hierarchical Set" pattern in Common Ground. 	

Navigation Support Pattern:

Identification	Pattern Name	CONVENIENT TOOLBAR
	Category	Navigation Support Patterns
Context of Use	User	Novice or Expert
	Task	Help the user reach convenient and key pages at any time from any page.
	Platform Capabilities and Constrains	<ul style="list-style-type: none"> - Suitable for Traditional Desktops, LapTop - For Mobile and Wireless applications, it should be implemented differently
Usability Problem	<ul style="list-style-type: none"> - The user can easily find useful and “safe” pages, regardless of the current state of the artefact. - The user can reach these pages in a timely fashion. 	
Usability Factors	<ul style="list-style-type: none"> - Efficiency, Safety - Consistency, Minimal Action, Minimal Memory, User Guidance, Helpfulness 	
Example	<ul style="list-style-type: none"> - Any Web site including small and mobile Web applications 	
Design Solution	<ul style="list-style-type: none"> - Group the most convenient action links, such as <i>home</i>, <i>site map</i>, and <i>help</i>. - Use meaningful pictures or terms as labels. - Position the toolbar at the same location throughout the website. 	
Implementation	HTML + Scripts, Java applets and beans, WML	
Related Usability Patterns	Superordinate	Focus Page, Tiled Page, Stack Page, Navigation Page, Utility Page
	Subordinate	None
	Neighboring	Shortcut, Path, Map, Browsing Index
	Competitor	None
Additional Reading	<ul style="list-style-type: none"> - “Convenient environment actions” and “Go back to a safe place” in Common Ground. - “Goal-oriented areas” in Experience. - “List Browser” in Amsterdam (Welie collection). - “Basic interface design” and “Links & navigation” in the Yale Web design guidelines. 	

Appendix B: NCBI Site User Evaluation



*Department of Computer Science
Concordia University*

Informed Consent to Participate in Human Subject Research

Homa Javahery, a researcher with the Human-Centered Software Engineering Group at Concordia University, and Dr. Ahmed Seffah, a professor of Computer Science at Concordia University, would appreciate your participation in a research survey designed to collect information about the user's, and in particular, the biologist's, experience with the National Center for Biotechnology Information (NCBI) website and information resource. You are being asked to complete an anonymous evaluation form that should take up no more than 30 minutes of your time.

While this information could be obtained by interviewing you, we feel that the evaluation form, sent by email, is the quickest and easiest method for obtaining this information.

We anticipate no risk to you as a result of your participation in this study other than the inconvenience of the time to complete the evaluation form. While there may be no immediate benefit to you as a result of your participation in this study, it is hoped that we may gain valuable information about your experiences with the NCBI website that can help to improve current Bioinformatics tools, applications, and information resources.

The information that you give us on the questionnaire will be recorded in anonymous form. We will not release information that could identify you. All completed surveys will be kept in a locked file cabinet in the office of Homa Javahery and will not be available to anyone not directly involved in this study. If you want to withdraw from the study at any time you may do so without penalty. The information on you up to that point will then be destroyed.

Once the study has been completed, we will be glad to give you the results. In the meantime, if you have any questions, please contact:

Homa Javahery
Human Centered Software Engineering Group
Department of Computer Science, Concordia University
Montreal, Canada
Tel. (514) 848-3024
Email. h_javahe@cs.concordia.ca
Web. <http://hci.cs.concordia.ca/www/>

I have read the above explanation and agree to participate in the study by simply filling out the evaluation form, entitled NCBI Site User Evaluation.

Name _____ Date _____

Signature _____

NCBI Site User Evaluation

Thank you for completing this evaluation form based on your experiences with the NCBI (National Center for Biotechnology Information) website. All responses will remain anonymous and results will be used solely for research purposes.

Part 1: User Information

Age: _____ Gender: Male Female

What is your first language? English French Other Please specify: _____

What is your field of study? _____

What is your highest level of education? _____

What is your current position/employment? _____

How many years have you been working in Bioinformatics or Molecular Biology?

<1 year 1-3 years >3 years Not applicable

How long have you been using the internet?

<6 mos. <1 year 1-3 years >3 years

Please list some of your hobbies and interests below:

How long have you been using the NCBI (National Center for Biotechnology Information) site?

<6 mos. <1 year 1-3 years >3 years

What are the three main tasks you perform on the NCBI site?

- 1.
- 2.
- 3.

Part 2: User Evaluation of NCBI Site

Please go through the NCBI website, found at <http://www.ncbi.nlm.nih.gov>, while answering these questions:

	Yes	No	Unsure
Do you find it easy to navigate on the NCBI website, especially when performing a new task?	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Is it visually clear what is a link or a button?	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Do you receive feedback and requested information promptly, such as when you perform a BLAST search?	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Is it easy to get lost when looking for information?	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	Yes	No	Unsure
Does the site use words, phrases, and concepts familiar to you?	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Are biological standards and terms used?	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Is the site's terminology easy to understand?	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	Yes	No	Unsure
Can you access the home page easily from any page?	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Do you find that you can cancel any operation when desired, for example when you perform a BLAST search?	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
When you choose a link or task by mistake, is it easy to go back?	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	Yes	No	Unsure
Are you confused at times because of inconsistent wording and terminology?	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Are images and fonts used consistently throughout the site?	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Are link names clear enough so that you know where they point before clicking on them?	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	Yes	No	Unsure
When filling out a form, such as the VAST search form to determine structure-structure similarity, do you know exactly which fields are <i>required</i> ?	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Does the system prompt you if you submit incorrect data? (Ex. If you submit a protein sequence instead of a nucleotide sequence)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Do you run into errors often when you use the site?	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Are error messages expressed in plain language that you easily understand?	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

	Yes	No	Unsure
If you don't know how to use something on the site, is it easy to find instructions?	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Can you recognize where you are on the site without having to remember your path from the homepage?	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Does the site use appropriate labels and descriptive links?	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Are logos, buttons, links, and colors uniform across the site?	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	Yes	No	Unsure
Does the site load quickly?	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Are there enough quick links and buttons to access pages that you use often?	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Can you locate information effectively with the NCBI search engine?	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Can you find all necessary functionality without leaving the NCBI site?	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	Yes	No	Unsure
Do you find that pages and windows contain unnecessary or excessive amounts of information?	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Are pages too lengthy?	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Is information broken up into relevant segments that are linked together logically?	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	Yes	No	Unsure
Is it easy to find help information on how to complete a desired task?	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Is help information described with concrete steps?	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Is it possible to access help information from any page?	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Part 3: General Questions

1. What do you like and dislike about the NCBI site's **home page**?
2. What do you like and dislike about the NCBI site in general?
3. How do you think the NCBI site can be improved?
4. If it were possible to view and interact with the NCBI site with a different device, such as a personal digital assistant (PDA) or a Pocket PC, would this be of use to you? Please explain.
5. Other comments:

Thank you! ☺

Appendix C: Mapping of Questions to Heuristics

1. Visibility and Navigation
Do you find it easy to navigate on the NCBI website, especially when performing a new task?
Is it visually clear what is a link or a button?
Do you receive feedback and requested information promptly, such as when you perform a BLAST search?
Is it easy to get lost when looking for information?
2. Language and Communication
Does the site use words, phrases, and concepts familiar to you?
Are biological standards and terms used?
Is the site's terminology easy to understand?
3. Control
Can you access the home page easily from any page?
Do you find that you can cancel any operation when desired, for example when you perform a BLAST search?
When you choose a link or task by mistake, is it easy to go back?
4. Consistency and Standards
Are you confused at times because of inconsistent wording and terminology?
Are images and fonts used consistently throughout the site?
Are link names clear enough so that you know where they point before clicking on them?
5. Error prevention and Recovery
When filling out a form, such as the VAST search form to determine structure-structure similarity, do you know exactly which fields are <i>required</i> ?
Does the system prompt you if you submit incorrect data? (Ex. If you submit a protein sequence instead of a nucleotide sequence)
Do you run into errors often when you use the site?

Are error messages expressed in plain language that you easily understand?

6. Recognition not recall

If you don't know how to use something on the site, is it easy to find instructions?

Can you recognize where you are on the site without having to remember your path from the homepage?

Does the site use appropriate labels and descriptive links?

Are logos, buttons, links, and colors uniform across the site?

7. Efficiency

Does the site load quickly?

Are there enough quick links and buttons to access pages that you use often?

Can you locate information effectively with the NCBI search engine?

Can you find all necessary functionality without leaving the NCBI site?

8. Minimalist Design

Do you find that pages and windows contain unnecessary or excessive amounts of information?

Are pages too lengthy?

Is information broken up into relevant segments that are linked together logically?

9. Help

Is it easy to find help information on how to complete a desired task?

Is help information described with concrete steps?

Is it possible to access help information from any page?

Appendix D: Complete Expert Results for NCBI Evaluation

Part A

Heuristic	Expert Evaluation
1. Visibility and Navigation	<ul style="list-style-type: none"> ▪ Different parts of the NCBI site look different; often the path or current position is not clear ▪ The only method for retracing a path is by using the browser's <i>Back</i> function
2. Language and Communication	<ul style="list-style-type: none"> ▪ Good for target group ▪ Home page is in more general language, and other pages are more specific to user group
3. Control	<ul style="list-style-type: none"> ▪ Undo is only supported by the browser's <i>Back</i> button ▪ The only way to exit functions in one step is to go back to the homepage, or jump to a new area ▪ Redo functionality is minimal; for instance, perform a different function on the same protein, or the same function on a different protein
4. Consistency and Standards	<ul style="list-style-type: none"> ▪ Menu structure changes a lot from page to page, so the user gets easily confused and loses orientation
5. Error prevention and Recovery	<ul style="list-style-type: none"> ▪ Most error messages are not linked to help pages or suggestion of a solution
6. Recognition not recall	<ul style="list-style-type: none"> ▪ Visual feedback is provided on the presence of most links, either through visual clues on the link or through dynamic changes as the cursor passes over the link; however the feedback method does not seem to be consistent throughout the site ▪ Menus vary almost with every new sub-site; one has to remember the path taken, and in which menu (area) one is operating
7. Efficiency	<ul style="list-style-type: none"> ▪ To speed up navigation the user can use the Site Map or Hot Spots ▪ Besides the above, there seem to be few if any short-cuts; they would be appropriate in the search functions in this context of use ▪ It would be a good idea to allow users to create their own space, customize menus, and create their own shortcuts
8. Minimalist Design	<ul style="list-style-type: none"> ▪ The left-hand menu sometimes contains text descriptions of menu items. Aside from being non-standard, this lengthens the menu unnecessarily, so that users have to scroll to find all of the elements. These text descriptions should be implemented as rollovers (pop-up text). ▪ Dialogues present unnecessary information ▪ Pages are often overloaded with links and give the user too much possibility to get lost (ex: homepage contains 45 links).
9. Help	<ul style="list-style-type: none"> ▪ Help and documentation is provided, but unfortunately not always directly linked to the problem ▪ There is an online guide and information on how to use the NCBI site, but this information is hard to find ▪ Help information is often contextual to the page displayed, which is good ▪ <i>Help</i> button or icon should be in a consistent place; this is done in some parts of the interface (e.g. BLAST overview), but in other parts of the site, the user has to search through the menu for a contextual help topic

Part B

1. Navigation structure of website

The navigation structure of the website is complicated, and there are few aids to help users understand where they are in the structure, or to quickly return to earlier steps. The only method for retracing a path is by using the browser's *Back* function. As a result, it can be tedious to return to an upper-level page from a deeply embedded page. In summary, the site is big and fairly complex, so it is easy to get lost and lose orientation.

2. Home page

There are well-identified zones of content, although overloaded with information. It is not always clear what is a link. In certain zones such as *Pub Med Central*, there seem to be many individual links but in fact, the whole zone is a single link. This is confusing and frustrating. In addition, the main (left-hand) menu contains textual descriptions of menu items. Aside from being non-standard, this lengthens the menu unnecessarily, so that users have to scroll to find all of the elements. These textual descriptions should be implemented as rollovers (pop-up text) for new users. The main menu items are not optimally legible because the font is yellow on a blue background, resulting in reduced contrast and visibility. In summary, the site is overloaded with links, low in visibility, and first time users will probably give up since there is no guidance. It may be interesting to have a different home page for different users.

3. Site Map

This page is excessively long and complicated. The site map should include only the site structure. The alphabetical index is an excellent idea, but the 3-column format makes it difficult to use. The Resource category structure uses paragraph format with several items on one line and some items straddling two lines. This makes it difficult to scan vertically to find a target, which is the main use of this type of list. The list items should be in bullet format, one item per line. For new users, the site map is way too crowded and does not provide much help. In summary, the site map is designed to give users who know what they are looking for, fast access to a certain sub-site; and for new users, additional help in locating a page or topic. The site map is complicated,

and difficult to use for either of these groups of users.

4. Search tools

The database search field is not highly visible, because it is attached to the page header rather than being in the content zone as expected. This is initially confusing. It is not clear how to add search results to the clipboard. Check-boxes seem to be provided for this purpose, but it is not clear how to use them. Overall, the search functions show a high degree of control and flexibility, which is appropriate due to the advanced nature of the task. However, this is not helpful for less experienced users. It is not clear how search results are sorted, for example whether they are ordered by decreasing relevance. In summary, the search tools are relevant for more experienced users, but more explanation and control should be given to newer users.