INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

ProQuest Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

Usability Patterns-Assisted Design for Web User Interfaces

Ning Li

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

Montréal, Québec, Canada

December 2001

ABSTRACT

Usability Patterns-Assisted Design for Web User Interfaces

Ning Li

Many Web design and usability problems have a tendency to recur in various projects. Web designers often try to reinvent solutions from scratch, because the current Web design guidelines are hard to remember, and difficult to use effectively by novice Web designers or software engineers who are not familiar with user interfaces and Web design principles. Furthermore, current Web authoring tools have long been available to assist developers in integrating many aspects of Web applications, although the majority of these environments do not provide a mechanism for explicitly ensuring the usability of the developed Web user interfaces. For our research, we will describe a pattern language supported by an innovative tool for Web applications design. Using this tool, the patterns are shown to be more than an alternative design tool to current Web design guidelines and tools. We perceive them as an ideal vehicle for gathering best design practices, and transferring by means of Web authoring tools the knowledge of usability experts to software engineers unfamiliar with usability engineering.

Keywords: Design Pattern, Usability Pattern, Patterns-Assisted Design, Web-based Interactive Software, Java, HTML, XML.

Acknowledgements

I sincerely thank my supervisor, Dr. Ahmed Seffah, who introduced me to Web application usability and design, inspired me during this work and offered me valuable guidance.

I would also thank all the people who contributed to make this thesis achievable.

Finally, I am grateful to my family for the endless love and support.

Table of Contents

List of Figures.....viii

List of Tables.....viii

Chapter 1 Introdution 1

1.1 Challenges and Problems in Web Application Design 1

1.2 Current Design Tools 2

1.2.1 Web Scripting and Programming Tools.....3

1.2.2 Low Fidelity Prototyping Tools 4

1.2.3 Design Guidelines 5

1.3 Objective of This Research 6

Chapter 2 Patterns - An Alternative Design Approach to Current Tools and Guidelines 9

2.1 Design Pattern – The Origin.....9

2.2 Patterns in Software Engineering.....10

2.2.1 Software Patterns Definitions and Categories.....11

2.2.2 Forms for Documenting Patterns 11

2.3 Patterns in Human Computer Interaction.....13

2.3.1 Collections of Patterns and Pattern Languages for HCI 13

2.3.1.1 Common Ground: An Example of Pattern Language for HCI.....14

2.3.1.2 Experience.....16

2.3.1.3 The Brighton Usability Pattern Collection.....17

2.3.1.4 The Amsterdam Collection of Patterns in User Interface Design.....17

2.3.1.5 Other Pattern Organizations in HCI.....18

2.3.2 Patterns for Web Applications Design.....21

2.4 Benefits of Usability Patterns 26
2.4.1 Benefits of Patterns and Pattern Languages for HCI 26
2.4.2 Usability Patterns vs. Guidelines 26
2.5 Tools for Patterns-Assisted Design 27
2.6 Conclusion 30

Chapter 3 UPADE Web - A Usability Pattern Language for Web Applications 31
3.1 Overview of UPADE Web Language 32
3.1.1 Patterns Collection 32
3.1.2 Patterns Definition 33
3.1.2.1 Product Oriented Patterns 35
3.1.2.1.1 Architectural Patterns 35
3.1.2.1.2 Structural Patterns 38
3.1.2.1.3 Navigation Support Patterns 42
3.2 Format for Documenting Patterns 43
3.2.1 UPADE Format 43
3.2.2 Examples 48

Chapter 4 UPADE Editor - A Design and Prototyping Tool 53
4.1 Overview of UPADE Editor 1.0 53
4.1.1 Functionality 53
4.1.2 Interface 55
4.1.3 Collected Usability Patterns 57
4.2 Scenarios of Using UPADE Editor 1.0 58
4.2.1 Browse Function 58
4.2.2 Design Function 63
4.2.3 Other Features Demonstration 68

Chapter 5 Conclusion and Future Investigations 70

5.1 Conclusion 70

5.2 Further Investigations 71

References 72

Appendix A: UPADE Web Patterns 79

A.1 Sequential Pattern 79

A.2 Hierarchical Pattern 80

A.3 Grid Pattern 81

A.4 Composite Pattern 82

A.5 Focus Page Pattern 83

A.6 Utility Page Pattern 85

A.7 Navigation Page Pattern 86

A.8 Tiled Page Pattern 88

A.9 Stack Page Pattern 89

A.10 Executive Summary Pattern 90

A.11 On Fly Description Pattern 91

A.12 Form Pattern 92

A.13 Bullet Pattern 94

A.14 Shortcut Pattern 95

A.15 Convenient Toolbar Pattern 96

A.16 Path Pattern 98

A.17 Map Pattern 99

A.18 Browsing Index Pattern 100

List of Tables

Table 2.1 Alexandrian Pattern Form 12
Table 2.2 Usability Pattern Category Classifications 19
Table 3.1 UPADE Web Language Attributes and Possible Values 48
Table 3.2 Hierarchical Pattern 49
Table 3.3 Focus Page Pattern 50
Table 3.4 Convenient Toolbar Pattern 51

List of Figures

Figure 3.1 A Schematic View of the UPADE Framework Architecture and Process 31
Figure 3.2 An Overview of UPADE Web Language 32
Figure 3.3 An Overview of IBM-UPADE Design Process 33
Figure 3.4 A Diagram of Sequential Pattern 36
Figure 3.5 A Diagram of Hierarchical Pattern 37
Figure 3.6 A Diagram of Grid Pattern 37
Figure 3.7 A Composite Pattern combining Sequential Pattern, Hierarchical Pattern and Grid pattern 37
Figure 3.8 Focus Page Pattern 38
Figure 3.9 A Diagram of Tiled Page Pattern 40
Figure 3.10 A Diagram of Stack Page Pattern 40
Figure 3.11 An Example of On Fly Description Pattern 41
Figure 4.1 Main Interface of UPADE Editor 1.0 56
Figure 4.2 "Browse" Mode is Enabled When UPADE Editor is Initiated 59
Figure 4.3 Four Design Steps of IBM-UPADE Process 59
Figure 4.4 "Web Site Architecture" Step Relates to Information Architecture Patterns. 61
Figure 4.5 "Page Structure" Step Relates to Page Managers Patterns 61
Figure 4.6 "Navigation Elements" Step Relates to Navigation Support Patterns 61
Figure 4.7 "Information Elements" Step Relates to Information Containers Patterns 61
Figure 4.8 An Example of Pattern Description Window for Home Page Pattern 62

Figure 4.9 An Example of Initiating Design Function 64
Figure 4.10 An Example of Applying Hierarchical Architecture Pattern in a Web Application 65
Figure 4.11 A Blank Surface is Created for Further Web Page Design, Along with the Refreshed Page Structure Patterns in the Product Patterns Box 66
Figure 4.12 A Web Page Combining Home Page Pattern, Information Containers Patterns and Navigation Support Patterns 67
Figure 4.13 An Example of Pattern Compatibility Check 69

Chapter 1: Introduction

"When a technology turns mature, the corresponding market changes. It is not driven by the needs of technically sophisticated consumers anymore, since the technology is commonplace enough that many competitors can produce roughly equivalent performance. Now, to improve products, companies need a design philosophy that targets the human user, not the technology (Norman, 1987)."

1.1 Challenges and Problems in Web Application Design

The first generation of Web applications has been a succession of HTML pages. The interaction was essentially a static navigation through hypertext links. Today, the convergence of the Internet, mobile telephony, and personal digital assistants (PDAs) has led to the emergence of highly interactive Web-based applications. Because of the availability of tools, the development of these Web applications is not a research challenge; however the usability issues are still an open research question [Nielsen, 1999]. The following are some of the problems and challenges in Web application usability:

- Most current Web applications adopt a layered approach to segment the different layers of Web application architecture and to isolate platform specifics from remaining issues. However, this method requires a consistent approach to applying both cognitive and social factors to user interface design and that also requires

independent developers to coordinate their activities, but unfortunately, cooperating at this level may be beyond the abilities of the industry.

- More and more Web applications will offer different platforms and devices. As computing devices exhibit drastically different capabilities, coping with such drastic variations implies much more than mere layout changes. For example, the small screens available on many PDAs only provide coarse graphic capabilities and would not be suitable for photo editing applications.

- Although many design guidelines have been issued to Web application designers, these guidelines differ from one platform or device to another. When designing a Web application with multi-devices the varying guidelines can be a source of many cross-platform inconsistencies. Even if some cross-platform guidelines have been published, they still do not take into account the particularities of a specific device, especially the device constraints and capabilities, so this can also be problematic for a user utilizing different kinds of devices to interact with the same server side service or information.

1.2 Current Design Tools

Tidwell pointed out that the latest technology and design tools can help the designer to build much more creative Web applications than in the past, but there is still room for improvement in the design process and in usability issues [Tidwell, 1999]. In particular,

we would like to integrate more efficient and cost-effective design in the development process and tools.

1.2.1 Web Scripting and Programming Tools

HTML (Hypertext Markup Language) defined a set of tags itself. HTML files can be created and processed by a wide range of tools, from simple plain text editors to sophisticated WYSIWYG (What You See Is What You Get) authoring tools. However, different browsers running on different platforms may have different styles for mapping and presenting Web page elements.

Other Web scripting languages have been introduced recently to overcome this problem. For example, XHTML (Extensible Hypertext Markup Language) provides diversity to Web pages on a range of browser platforms including desktops, mobile phones, televisions and cars. DHTML (Dynamic HTML) combines several built-in browser features that enable a Web page to be more dynamic. XML (Extensible Markup Language) is the universal format for structured documents and data on the Web. It allows developers to define their own mark-up formats, whereas HTML does not. XML is used increasingly for data. The latest version of XML allows developers to utilize the Web to exchange data among tools, applications, and repositories. It also helps to create secure and distributed applications built in a team development environment.

In addition to the trend of Web scripting tools, Java Integrated Development Environments such as Visual Age, Web Sphere and JBuilder have become more popular. IBM VisualAge for Java supports the complete cycle of Java program development.

VisualAge can build and test Java applets, servlets, and Enterprise JavaBeans. Borland JBuilder is also a visual development environment for building applications, applets, JSP/Servlets, JavaBeans, EJB and distributed J2EE applications for the Java 2 Platform. IBM WebSphere is a tool suite that helps designers to visually create a Web site.

Today, the evolution of Web scripting and programming tools enables the designer to have more power to develop heterogeneous Web applications. The latest Web user interface is a sophisticated mixture of markup tags (HTML, PHP, XML, WML), style sheets (CSS, XSL), scripts (JavaScript, VBScript) as well as embedded objects such as Java applets, ActiveX and plug-ins.

1.2.2 Low Fidelity Prototyping Tools

An alternative approach to Web scripting and programming environments is low fidelity prototyping tools such as DENIM (Design Environment for Navigation and Information Models) and SILK (Sketching Interfaces Like Krazy). DENIM and SILK are developed by GUIR (Group for User Interface Research at university of California at Berkeley). Both are low fidelity prototyping tools for early design of Web sites.

SILK is a pen-based electronic sketching tool for user interface design [UCB, 2001]. It supports the designer to roughly draw an interface using an electronic pad and stylus. Compared with traditional paper sketch, SILK has more interaction capability to facilitate the designer.

DENIM supports Web site design in the early stages, rather than creating finished Web sites [Lin et al., 2000]. DENIM is based on the original SILK project [UCB, 2001]. It allows sketching input from electronic devices. However, DENIM can support design activities at five different refinement levels (Overview, Site map, Storyboard, Sketch and Detail) and integrate these levels through zooming, while SILK can not, but DENIM does not recognize the widgets drawn by the designer, whereas SILK does.

1.2.3 Design Guidelines

Guidelines aim to capture the design knowledge into small rules, which can then be used when designing new user interfaces. Guidelines represent the middle level of design guidance in a progression from abstract principles to specific conventions [IBM, 2001a].

Yale guidelines were compiled in the book "Web Style Guide" written by Patrick J. Lynch and Sarah Horton [Lynch & Horton, 1999]. The authors described basic design principles for creating Web sites. Lynch and Horton intended to combine user interface design, graphic design, technical skills and traditional editorial principles into Web site design. Yale guidelines are classified into five categories; Interface Design, Site Design, Page Design, Web Graphics and Web Multimedia. Each category consists of several sections, which grouped the guidelines into several specific aspects. Each guideline is illustrated by a textual description, some figures, references and examples.

IBM Web Guidelines [IBM, 2001a,b] were developed by IBM Ease of Use Group for constructing ease of use Web interfaces. Although most of these guidelines aim to help

novice and intermediate level Web designers, experienced designers can still benefit from them. The majority of IBM Web guidelines are presented according to the Web design process including "Planning", "Design", "Production", and "Maintenance" phases. In addition, there is a set of guidelines particularly grouped by the E-commerce topic. IBM proposed five steps at the "Design" phase of Web design process. They are "Structure", "Navigation", "Text", "Visual Layout and Elements", and "Media" steps.

It has often been reported that guidelines have a number of problems when used [Martijn et al., 2000] in that it is difficult to select appropriate guidelines that apply to a particular design problem. Moreover, occasionally guidelines may seem to contradict each other and consequently the designer may still not solve the design problem. In addition, Tidwell pointed out that it was hard for a novice designer to even remember all the guidelines, let alone use them effectively [Tidwell, 1999]. Sometimes it is difficult to make the trade-off among these principles when they come into conflict. The designer frequently has to determine the best solution by guessing, or by resorting to other means.

1.3 Objective of This Research

In this research, we are exploring usability patterns as a design tool to capture best practices on the design of usable Web applications. We aim to define a usability pattern language for Web applications and develop a tool to support that pattern language.

1. Define a usability pattern language

This usability pattern language should feature the following characteristics:

- It should contain a set of usability patterns that gather and disseminate the design knowledge related to the user interfaces of Web applications.
- All the patterns should share the same format. We need to define a unified pattern format for documenting all the usability patterns.
- Patterns are interrelated. It is critical for a pattern language, as pattern language only exist when there is relationship between patterns.
- Patterns are related to Web design process. To help the designer know when to apply patterns during the Web application design, we should define the relation between patterns and design activities.

2. Develop an editor tool

In order to make our usability pattern language usable, especially for novice user interface designers or software developers who are not familiar with HCI design, we need to develop an editor tool to support usability pattern engineering. This tool should provide the following functionality:

- Collect all the patterns that have been defined in our usability pattern language.
- Browse the description of each pattern: The editor should provide just in-time information of each pattern for users.
- Combine existing patterns: Users of the editor tool can select appropriate patterns and combine or glue them at an abstract level.
- Establish a low fidelity prototype of Web applications: The tool should support the user in creating a prototype employing existing usability patterns at an abstract level.

- Demonstrate the relationship between patterns and the design process: The tool should illustrate the relevant patterns according to the design process and assist users in applying appropriate patterns during the design activities.

Chapter 2: Patterns - An Alternative Design Approach to Current Tools and Guidelines

"A pattern describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice (Alexander, 1977)"

2.1 Design Pattern - The Origin

Design Pattern, a popular concept in the software engineering community, was derived from the writings of the architect Christopher Alexander in the 1970s. In his book "The timeless way of building", Alexander described the term Pattern as "a three-part rule, which expresses a relation between a certain context, a problem and a solution". Alexander emphasized that each pattern is a relationship between a certain context, a certain system of forces which occurs repeatedly in that context, and a certain configuration which allows these forces to resolve themselves in that context [Alexander, 1979].

On the basis of Alexander's writing, Doug Lea suggested that a good design pattern should have six properties [Lea, 1993]:

- **Encapsulation:** Each pattern encapsulates a well-defined problem and solution.

- **Generativity:** Each pattern contains a local, self-standing process prescription describing how to construct realizations.
- **Equilibrium:** Each pattern identifies a solution space containing an invariant that minimizes conflict among forces and constraints.
- **Abstraction:** Patterns represent abstractions of empirical experience and everyday knowledge.
- **Openness:** Patterns may be extended down to arbitrarily fine levels of detail.
- **Composibility:** Patterns are hierarchically related. Most patterns are both upwardly and downwardly composible.

2.2 Patterns in Software Engineering

Patterns have been used in many different fields such as architecture, biology, chemistry and physics, recently blossoming in the software engineering community. Software patterns were first popularized by software engineering researchers in the wake of the Gamma book "Design Patterns: Elements of Reusable Object-Oriented Software" [Gamma et al., 1994]. Grady Booch claimed that patterns are very useful to object-oriented realms because they represent a higher leverage form of reuse [Rising, 1998]. Like Alexandrian patterns, the object-oriented software patterns provide concrete design solutions that not only can be put into practice right away, but good results can be obtained immediately. These patterns are sufficiently abstract as well so they can be applied to innumerable situations [Tidwell, 1999].

2.2.1 Software Patterns Definitions and Categories

The goal of software patterns is to create a body of literature to help software developers resolve recurring problems encountered throughout the entire software development lifecycle. Researchers have described software patterns and patterns language into heterogeneous definitions. Jim Coplien defined a pattern language as a structured collection of patterns and the rules to combine patterns into an architectural style [Coplien, 1996]. He stated that each pattern works in a certain context and can transform the system to produce a new system in a new context.

In the software engineering community, patterns have been largely applied to software architecture and design, and lately on software development processes and organizations. As different theoretical and philosophical ideas are applied in software patterns, patterns have been defined and categorized into varied classifications, such as

- Design Patterns [Gamma et al, 1994; Bayle et al, 1998]
- Process Patterns [Coplien 1995; Ambler, 1998]
- Activity Patterns [Bayle et al, 1998]

Other pattern categories include Organizational Patterns, Analysis Patterns, Framework Patterns and Anti-Patterns.

2.2.2 Forms for Documenting Patterns

A number of forms have been proposed in order to document patterns.

Table 2.1 describes the structure of Alexandrian form [Alexander et al., 1977].

Notation	*Description*
Title	It briefly describes the solution that the pattern offers.
Asterisks	It marks the significance of the pattern.
Picture	It depicts an archetypal example of the pattern.
Introductory paragraph	It sets the context and links to higher level or larger patterns.
¤¤¤	To mark the beginning of the problem
Headline	It summarizes the essence of the problem.
Body of the problem	It describes the empirical background of the pattern, the evidence for its validity, range of variation of manifestation.
Solution	It describes the physical and social relationships, which are required to solve the stated problem in the stated context.
Diagram	It is the illustrated diagram of the solution, along with certain labels indicating the main components.
¤¤¤	To mark the end of the main body of the pattern
Connections to smaller patterns	It is required to complete the pattern to connect to lower level or smaller patterns.

Table 2.1 Alexandrian Pattern Form

Other pattern forms include:

1) The GOF Form [Gamma et al., 1994];

2) Portland Form [Portland, 2001];

3) Coplien Form [Coplien, 1996].

2.3 Patterns in Human Computer Interaction

HCI researchers began to widely explore the utility of pattern languages for human computer interaction design after the CHI'97 workshop. They defined the Interaction Design Pattern as a pattern that describes a connection between a repeatedly encountered problem and a proven solution particularity in the HCI field [Bayle et al., 1998]. Hence, Interaction Design Patterns started to affect the reliable HCI design and appeared to be an alternative approach to current guidelines. In that workshop, HCI researchers also observed that it was difficult to distinguish among patterns, as there was a continuum of HCI patterns across various levels. The higher level patterns were more robust over time than lower level patterns, while lower level patterns changed more rapidly along with the evolution of technology.

2.3.1 Collections of Patterns and Pattern Languages for HCI

In the HCI community, there have been vigorous discussions on patterns and pattern languages worldwide since 1997, and there are many research groups devoting time to the development of pattern languages. Among these heterogeneous collections of HCI patterns and pattern languages, "Common Ground", "Experience", "Brighton" and

"Amsterdam" play a major role and have had a significant influence on the exploit of HCI patterns and pattern languages.

2.3.1.1 Common Ground: An Example of Pattern Language for HCI

Among pattern languages, Common Ground is one of the earliest, largest-scale and most influential HCI pattern collections [Tidwell, 1999]. Common Ground was defined and collected by Jennifer Tidwell. It contains a set of interrelated patterns sharing the uniform format.

Goal and objectives

Common Ground aims to assist various users in their routine work including HCI designers, GUI developers, usability engineers and test engineers. Furthermore, its intention is to benefit the whole industry and UI design practitioners.

Pattern form

The elements of the pattern form adopted by Common Ground are Pattern Name, Examples, Bad Examples (if any), Context, Problem, Forces, Solution, Resulting Context and Notes. The description of each pattern contains a context of use, a problem that needs to be solved, a set of forces in that context, and a primary rule or secondary rules (sometimes) on how those forces solve the particular problem. Both good and bad examples are given. The bad examples illustrate the misusage of the pattern or certain situations in which the pattern should have been used but was not.

Pattern categories

Patterns in Common Ground are interrelated. They are grouped into several major sets: Primary Patterns, Information Organization, Posture, Working Surface Organization, Actions and so on. Each set may contain different categories according to various characteristics. There are in total 10 categories and 61 specific patterns in Common Ground including:

- **Primary patterns for contents**: Contains Narrative, High-density Information Display and Status Display.

- **Primary patterns for actions**: Contains Form, Control Panel, WYSIWYG Editor, Composed Command and Social Space.

- **Information Organization**: Describes how to present the content or information to the user. It contains Navigable Spaces, Overview Beside Detail, Step-by-Step Instructions, Small Groups of Related Things, Series of Small Multiples, Hierarchical Set, Tabular Set, Chart or Graph, Optional Detail On Demand, Disabled Irrelevant Things, Pointer Shows Affordance and Short Description.

- **Posture**: Contains Sovereign Posture, Helper Posture and Background Posture.

- **Working Surface Organization**: Describes how to organize the content or action into working surfaces. It contains Central Working Surface, Tiled Working Surfaces, Stack of Working Surfaces and Pile of Working Surfaces.

- **Navigate Techniques**: Describes the means that the user navigates through the artifact. It contains Map of Navigable Spaces, Clear Entry Points, Color-Coded Sections, Go Back One Step and Go Back to a Safe Place.
- **Actions**: Describes various specific actions that the user may take. It contains Convenient Environment Actions, Localized Object Actions, Actions for Multiple Objects, Choice from a Small Set, Choice from a Large Set, Sliding Scale, Editable Collection, Forgiving Text Entry, Structured Text Entry and Toolbox.
- **How can the user modify the artifact?** Contains User Preferences, Personal Object Space, Scripted Action Sequence, User's Annotations and Bookmarks.
- **How can the artifact be made visually clear and attractive?** Contains Iconic Reference, Calm Grid, Repeated Framework, Few Hues and Many Values.
- **How else can the artifact actively support the user?** Contains Good Defaults, Remembered State, Interaction History, Progress Indicator, Important Message, Reality Check, Demonstration, Quick Access and Familiar Quantity.

Common Ground does not provide any clear diagram or coherent paths to depict the interrelationship among those patterns, and it does not address issues regarding implementation of such patterns.

2.3.1.2 Experience

Experience, as a pattern language for user interface design, emphasizes on the user's experience within software systems. It was developed by Todd Coram and Jim Lee [Coram & Lee, 1996]. Experience concentrates on the interactions between the user and

the interfaces of software applications. All the patterns are grouped by areas of focus including Interaction Style, Explorable Interface, Sound and Symbols. In Experience, there is a clear connection map to illustrate the relationship among the patterns; however all the patterns are described in a narrative form using a natural language.

2.3.1.3 The Brighton Usability Pattern Collection

The Brighton usability pattern collection was compiled by members of the Usability Group at the University of Brighton, UK [Brighton UK, 2001]. This collection utilizes a matrix map to present the relationship among the patterns. So far, 21 patterns have been addressed in this collection, though only 10 patterns have been fulfilled completely. Compared with Experience, the Brighton pattern collection does not disclose the particular rationale for the sequence in which patterns are discovered and written up. Moreover, the patterns in Brighton collection adopt a narrative pattern form. It is not so structured and formal as Common Ground [Martijn et al, 2000].

2.3.1.4 The Amsterdam Collection of Patterns in User Interface Design

This collection was mainly established by Martijn van Welie. It is a collection of patterns in user interface design that strictly focus on problems of the end-user when they interact with systems, rather than the problems of designers [Martijn, 2000]. The Amsterdam pattern collection, taking user's perspective, makes an explicit distinction between the user perspective and the designer perspective. All the patterns are described in a uniform pattern form, which contains Problem, Principle, Context, Forces, Solution, Rationale,

Examples, Known Uses and Related Patterns. The Amsterdam collection does not address the implementation issues of user interfaces.

2.3.1.5 Other Pattern Organizations in HCI

In addition to the above four prominent pattern collections or languages, other pattern collections exist. Although these collections are not complete, they are a good starting point for the emergence of a unified pattern language for user interface design. Table 2.2 illustrates the classifications of usability patterns suggested by several research groups.

Reference	Patterns	Comments
PSA : **A Pattern-Supported Approach to the User Interface Design Process** [Granlund & Lafrenière, 1999]	Patterns are classified into 5 categories: **1. Business Domain patterns:** To describe the type of business, goals, the actors and business processes. **2. Business Process patterns:** To describe typical processes and actors involved in the delivery of services/goods in compliance with the business goals. **3. Task patterns:** To capture the knowledge about the task, typical users and their work context from previous similar projects. **4. Conceptual Design patterns:** To describe a way to optimally arrange information and activities to support the user' s task. **5. Design patterns:** These patterns are based on Common Ground.	- Compared with other collections that focus on screen design issues, PSA suggests a wider scope for the use of patterns by looking at the overall user-oriented UI design process. - The first three categories are domain specific; the latter two are generic. - The authors proposed a pattern- supported UI design process: 1) System Definition step: Business domain and Business process patterns. 2) User Profiling and Task Analysis step: Task patterns. 3) Conceptual Design step: Conceptual patterns 4) Design step: Design patterns. - Pattern format: Name, Context, Problem, Forces, Solution, Examples.

Pattern Languages for Usability [Mahemoff & Johnston, 1998]	Patterns are collected into 4 categories: **1.Patterns of tasks:** To identify the user actions and suggest the appropriate support mechanisms. Example: "Open Existing Document". **2. Patterns of users:** To ensure the products are meeting the demands of all users. Example: "Intermediate User, Domain Expert". **3. Patterns of user-interface elements:** To be applied by designers of user interface and software architecture. Examples: "Scrollbar", "Show Status". **4. Patterns of entire systems:** To capture the issues involved in the development. Example: "Document Manipulator".	- Focus on pattern meta-collections, rather than specific collections of patterns. - Proposed that patterns for usability are relevant at two levels of abstraction: 1) Higher-level patterns can solve problems related to user- interface and task support. 2) Lower-level patterns may suggest detailed design and implementation of usable software.
Notes on a Pattern Language for Interactive Usability [Casaday, 1997]	Identified three types of usability patterns: **1.Simple Patterns:** One usability attribute dominates. **2. Intrinsic Patterns:** Multiple usability attributes conflict or support. **3.Circumstantial Patterns:** Usability attributes must be achieved, but external factors constrain the solutions.	- The pattern language is based on traditional usability attributes: Learnable, Memorable, Efficient, Reliable, Flexible, Automated, Understandable, Subjectively Satisfying.

Table 2.2 Usability Pattern Category Classifications

Although interest in patterns for User Interface Design has existed for several years, there is still no uniform pattern language or pattern form accepted by the whole HCI community so the pattern language for user interface design has not yet been established. Different usability groups collect the patterns and develop the languages according to their own understanding and criteria. Many usability patterns defined by different research groups are similar, even overlapped. For example,

- "Progress Indicator" in Common Ground, "Progress" in Amsterdam and "Time to do something else" in Brighton are exactly the same patterns. All of them aim to indicate to the user how much time remains or how far along the process is.
- "Show the format required" in Brighton and "Unambiguous format" in Amsterdam are identical in assisting the user to input the data in a correct syntax.
- "Reality Check" in Common Ground, "Shield" in Amsterdam and "Think twice" in Brighton are similar solutions for the same problem occurring in similar context. For example, in order to protect the user from some irreversible and dangerous situation, "Reality Check" Pattern tells the user the side effects and asks the user to confirm his/her action; "Shield" Pattern also suggests to add an extra protection layer and thus asks the user to confirm the action; Similarly, "Think Twice" Pattern proposes a protection of confirmation such as requiring the user to complete a simple dialogue in certain irreversible situations, so as to indicate to the user the potential problems.

In addition to the patterns mentioned, the patterns underneath have similar declarations:

- "Tiled working surfaces" in Common Ground and "Container Navigation" in Amsterdam;

- "Stack of working surfaces" in Common Ground and "Navigation spaces" in Amsterdam;
- "Short description" in Common Ground and "Hinting" in Amsterdam;
- "Overview beside detail" in Common Ground and "Preview" in Amsterdam;
- "Convenient environment actions" in Common Ground and "Goal oriented areas" in Experience.

In contrast to the patterns enumerated above, some patterns have the same or similar pattern name; however, their declarations and intrinsic contents are quite different, therefore it may confuse the user under some circumstances. For example, "Give a warning" in Brighton seems literally like "Warning" in Amsterdam. In fact, "Give a warning" suggests giving users a warning message without breaking down the workflow, whereas "Warning" intends to interrupt the task and warn the user before continuing his task.

2.3.2 Patterns for Web Applications Design

Among all the patterns illustrated in the above section, the following patterns can be particularly suitable for Web application design.

1) **Common Ground**: Narrative, High-density Information Display, Status Display, Form, Navigable Spaces, Overview Besides Detail, Small Groups of Related Things, Hierarchical Set, Tabular Set, Chart or Graph, Optional Detail On Demand, Disabled Irrelevant Things, Pointer Shows Affordance, Short Description, Helper Posture, Central Working Surface, Tiled Working Surfaces, Stack of Working Surfaces, Map of

Navigable Spaces, Clear Entry Points, Color-Coded Sections, Go Back One Step, Go Back to a Safe Place, Convenient Environment Actions, Choice from a Small Set, Choice from a Large Set, Sliding Scale, Editable Collection, Structured Text Entry, Bookmarks, Repeated Framework, Good Defaults, Important Message and Reality Check.

2) Experience: Interaction Style, Explorable Interface, Multiple Settings, Command Control Center, Garden of Windows, Goal Oriented Areas, Modeless Feedback Area, Visual Symbols.

3) Brighton: Give a Warning, Interaction Feedback, Just Looking, Show Computer is Thinking, Show the format required, Time to Do Something Else.

4) Amsterdam patterns collection: Command Area, Container Navigation, Contextual Menu, Focus, Hinting, List Browser, Navigating Space, Preview and Unambiguous Format.

Meanwhile, there are several other usability research groups that particularly emphasize the study of patterns for Web applications:

5) Usability Patterns for World Wide Web Applications (PloP '99 Conference)
This set of Web usability patterns, identified by Kimberly Perzel and David Kane, focuses on the context of Web applications [Perzel & Kane, 1999]. These patterns represent Circumstantial Patterns suggested in Casaday Usability Pattern language [Casaday, 1997] and can be categorized into User Interface Widget (UI) and System

Based (Sys.) classifications according to the criteria of Mahemoff's classifications of usability patterns [Mahemoff & Johnston, 1998]. The patterns collected in this set are:

- **Carrot and a Stick (Sys.):** Offers the users something valuable and free at first and then induces the users to provide information that they might be unwilling to share.

- **Policy Statement (Sys.):** Provides a detailed description of how the information will be used and establishes sufficient trust with users in order to acquire personal information from them.

- **Required Field Markers (UI):** Labels the essential information clearly in order to ensure the user provides correct information.

- **What They See is All They Get (UI):** Displays the most critical content of a Web page in the upper left area of the screen to ensure the user will see the most important contents of a Web page.

- **Plan B (UI):** Provides an optional solution to the disabled people who present the information without graphics.

In addition, there are other candidate patterns in this set including Client-Side Validation (Sys), Server-Side Validation (Sys), Three Clicks or You're Out (Sys.), Site Context (UI), Location, Location, Location, You Are Here (UI), Flag Planting (Sys.), Universal Navigation (UI/Sys.), Searching the Web (Sys.), Web Registration Forms (UI), Links, Jumps and Suicide Leaps (UI/Sys.), Query Forms (UI) and Data Entry Forms (UI).

6) HPR- Hypermedia Design Patterns Repository

HPR, standing for Hypermedia Design Patterns Repository, intends to provide useful design patterns for hypermedia and Web applications [HPR, 2001]. There are 28 design patterns in this repository currently. The patterns have been classified into 3 categories:

- **Interface and Layout Pattern category**: Arranges the objects on the screen physically. It contains Active Reference, Behavioral Grouping, Behavior Anticipation, Here I am, Index Navigation, Info-Interaction Decoupling, Information Factoring, Information on Demand, Information-Interaction Coupling, Process Feed-back, Selectable Keywords, Selectable Search Space and Simple Search Interface pattern.

- **Structure/Navigation Pattern category:** Provides the schema to assist the user navigation through the Web application. It contains Collection Center, Complex Entity, Guided Tour, Hybrid Collection, Index Navigation, Navigation Strategy, Navigational Context, News, Node as a Navigational View, Opportunistic Linking, Selectable Search Engine, Set-Based Navigation, and Structured Answer pattern.

- **Content Oriented Pattern category**: Organizes the information of the Web application in general. It contains Advising, Analyse Organize Synthesize and Complex Entity pattern.

HPR patterns don't adopt the Alexdarian pattern form. All the patterns are not described into unified format either. The elements of the pattern template used are: Pattern Name, Keywords (sometimes), Problem Statement, Motivation, Forces (only 5 patterns have),

Solution, Consequences (occasionally), Known Uses (sometimes), Discussion (sometimes), Examples, Implementation and Related Patterns.

7) Three Hypermedia Design Patterns

This set of patterns was developed by D.M. Germán and D.D. Cowan, who proposed three design patterns for hypermedia design. These design patterns intended to give the designer the foundation to cope with problems, rather than provide a specific process to solve a particular problem [Germán & Cowan, 1999].

This collection documents the patterns using the pattern form of Gamma [Gamma et al, 1994], which contains Name, Intent, Motivation, Applicability, Structure, Participants, Collaborations, Consequences and Known Uses. There are 3 design patterns proposed in this collection:

- **Hyper-book**: Transfers a traditional textual document into a hypertext version and organizes the content in the most effective way.
- **Hyper-map**: Organizes certain kind of information, which is related to two-dimensional data.
- **Virtual product**: Provides and organizes effective information of virtual products in order to present the product in the electronic catalog attractively to the user.

8) Hierarchical Structure Through Navigation Side Bars

One design pattern -"Side Bar Navigation" was suggested for hypertext in this paper [Østebye, 1999]. This pattern intends to organize the layout of the navigation

information by applying the hierarchical structure to the hierarchical navigation links and the direct linking to cross-references links.

2.4 Benefits of Usability Patterns

2.4.1 Benefits of Patterns and Pattern Languages for HCI

Usability is a critical factor for successful Web applications. Web application development often involves the IT professionals with widely various backgrounds. A usability patterns language, which provides a common language for communicating usability concerns among diverse participants, has the capability to enhance the communication and facilitate the UI design process. The advantages and benefits of communication provided by the usability patterns are much more valuable than in conventional software applications, as HCI is an interdisciplinary field. Thus, usability patterns can obviously assist the user-centered design process [Mahemoff & Johnston, 1998] and particularly benefit the Web application design [Perzel & Kane, 1999].

2.4.2 Usability Patterns vs. Guidelines

Guidelines have been shown to have problems concerning selection, validity and applicability [Martijn et al, 2000; Tidwell, 1999]. Most guidelines suggest a general absolute validity, but in fact, they can only be applied in a specific context. Guidelines have no intrinsic way of stating the context for which they apply and also it is often difficult to see what the problem is and why the guideline is set up in a particular way.

Patterns have appeared as a possible solution to solve certain problems which guidelines have suffered. Compared with guidelines, patterns have more advantages [Martijn et al., 2000; Brighton UK, 2001]:

- Guidelines are usually described in two forms: "do this" or "do not do this", whereas patterns only emphasize the "do this" aspect and the descriptions of patterns are much more constructive.

- Patterns clearly depict the context and tell the designer when, how and why the solution can be applied. The context and problem are explicit, as well as the solution based on a rationale. Furthermore, solutions in patterns are very concrete and thus it will not raise new problems.

- Patterns represent proven design solutions in a much richer context than guidelines, and patterns contain a great deal of explicit information, that guidelines do not. This makes patterns altogether a richer resource for the designer than guidelines. Patterns contain more complex design knowledge than guidelines, whereas several guidelines are integrated into one pattern.

- Patterns are problem oriented and are potentially more usable for designers than guidelines.

2.5 Tools for Patterns-Assisted Design

Developers have often found it difficult to translate the narrative pattern descriptions into a particular implementation. Even if some people have no trouble translating the pattern

into code, they still find it inconvenient when they have to do it repeatedly [Budinsky et al, 1996].

Furthermore, with the booming of patterns and pattern languages in the software engineering community, more recently in the HCI field, patterns assisted tools are being more frequently demanded by the system developers and usability engineers, regardless of their experience or inexperience. A particular pattern language tool can enhance the user's comprehension, decrease the complexity of the language, and eliminate the ambiguity of all the terminology, while putting the language into practice in a real environment, which is a critical issue for the pattern language exploit.

However, there is a lack of tool support for usability patterns engineering in the human computer interaction community. Only few tools for software patterns exist in the software engineering realm, such as:

1. Tool Support for Object-Oriented Patterns [Florijn et al., 1997]

This is a prototype tool that aims to support developing object-oriented programs using design patterns in three ways:

- To generate program elements for a new instance of a pattern that is taken from an extensible collection of "template" patterns.
- To integrate pattern occurrences with the rest of the program by binding program elements to a role in a pattern.
- To check whether occurrences of patterns still meet the invariants governing the patterns and repairing the program in case of problems.

This tool provides three level views in a program: the pattern level view, the design level view and the code level view. However, some research groups found this approach difficult for large designs and also found those design views cumbersome [Yacoub & Ammar, 1999].

2. IBM- Automatic Code Generation from Design Patterns [Budinsky et al., 1996]

This is a tool for generating design pattern code automatically from application-specific information supplied by the user. It also integrates on-line references to design patterns. The architecture of the tool has three components:

- The Presenter: Implements the user interface specified by Presentation Descriptions.
- The Code Generator: Interprets Code Generation Descriptions and then generates the code for a given pattern.
- The Mapper: Interprets Mapping Descriptions that specify how the other two components cooperate.

This tool is limited to system design and implementation; it does not support domain analysis, requirement specification, documentation and debugging.

3. POAD- Pattern-Oriented Analysis and Design [Yacoub, 1999; Yacoub & Ammar, 1999]

POAD is a structure composition approach to build constructional design patterns. The tool supports three design models for composing patterns. The three model views are:

- Pattern-Level view: Models the application as a visual composition of patterns at a high design level.

- Pattern Interfaces view: A refinement of the Pattern-Level view using pattern interfaces and explicitly defining relationships between interfaces.
- Detailed Pattern-Level view: Shows the details of the pattern, and reveals the internals of the pattern and the connection between interfaces and internals.

In addition to the above tools, there are other tools to support patterns in the software engineering community:

- **FACE**: Framework Adaptive Composition Environment [Meijler et al, 1997];
- **PSiGene CASE tool** [Schuetze et al., 1997];
- **The Pattern-Lint** [Sefika et al, 1996].

2.6 Conclusion

In this chapter, we traced the history of patterns, software engineering patterns and patterns for HCI. On the basis of our observations, we argued that patterns are a more cost-effective tool than Web guidelines. We observed that although there have been a number of patterns and pattern languages available for user interface design, most of the existing pattern languages don't address implementation issues; instead they emphasize the user's perspective, therefore, they cannot be used by novice developers. Moreover, most of the patterns are described using natural languages, which makes them difficult to use effectively and efficiently. Our investigations also showed lack of tools for supporting pattern-oriented design, in particular HCI design. Furthermore, we found that there were no particular usability pattern languages for Web applications.

Chapter 3: UPADE Web - A Usability Pattern Language for Web Applications

UPADE - **U**sability **P**atterns-**A**ssisted **D**esign **E**nvironment is an ongoing research project. For this research, we are investigating the original solution of employing usability patterns as a framework for integrating usability in CASE tools while increasing the usability of systems.

Figure 3.1 depicts the whole structure of UPADE framework.

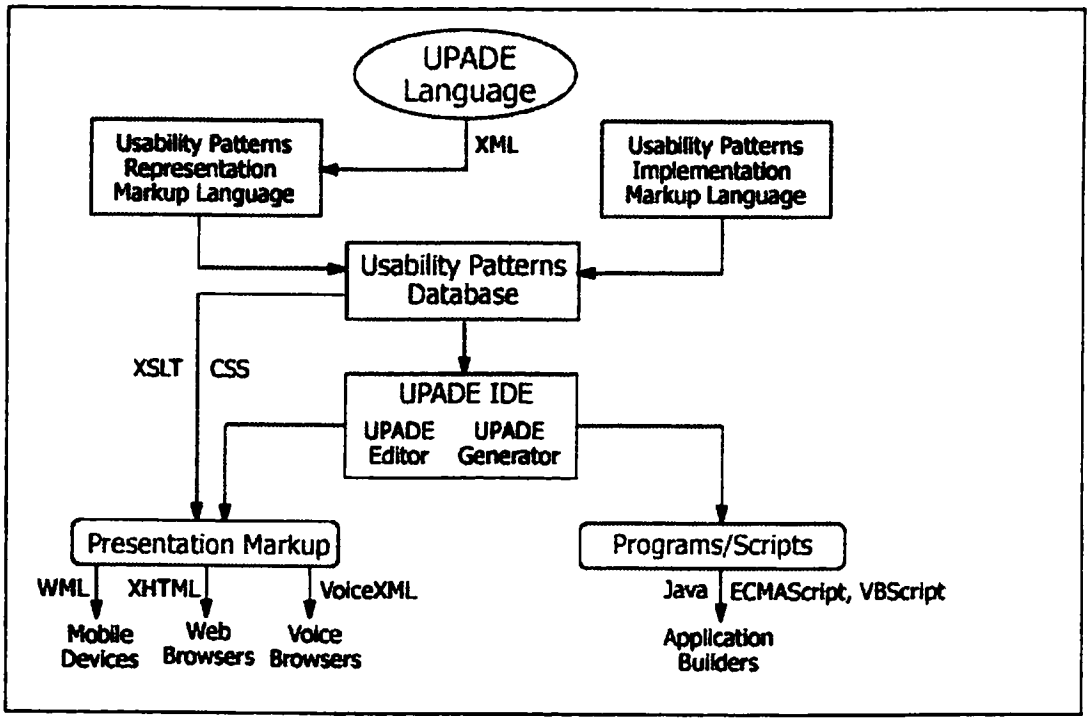


Figure 3.1: A Schematic View of the UPADE Framework Architecture and Process

Within the UPADE framework, at the present stage, we are exploring how to establish an UPADE language for Web applications (**UPADE Web language**) and developing an editor tool (**UPADE Editor**) for browsing, editing and combining UPADE Web patterns.

3.1 Overview of UPADE Web Language

UPADE Web language is a usability pattern language. It contains a set of interrelated patterns whereby each of the usability patterns provides a proven solution for a common usability problem that occurs in a specific context of use for Web applications.

3.1.1 Patterns Collection

UPADE Web language defines three categories of product-oriented patterns.

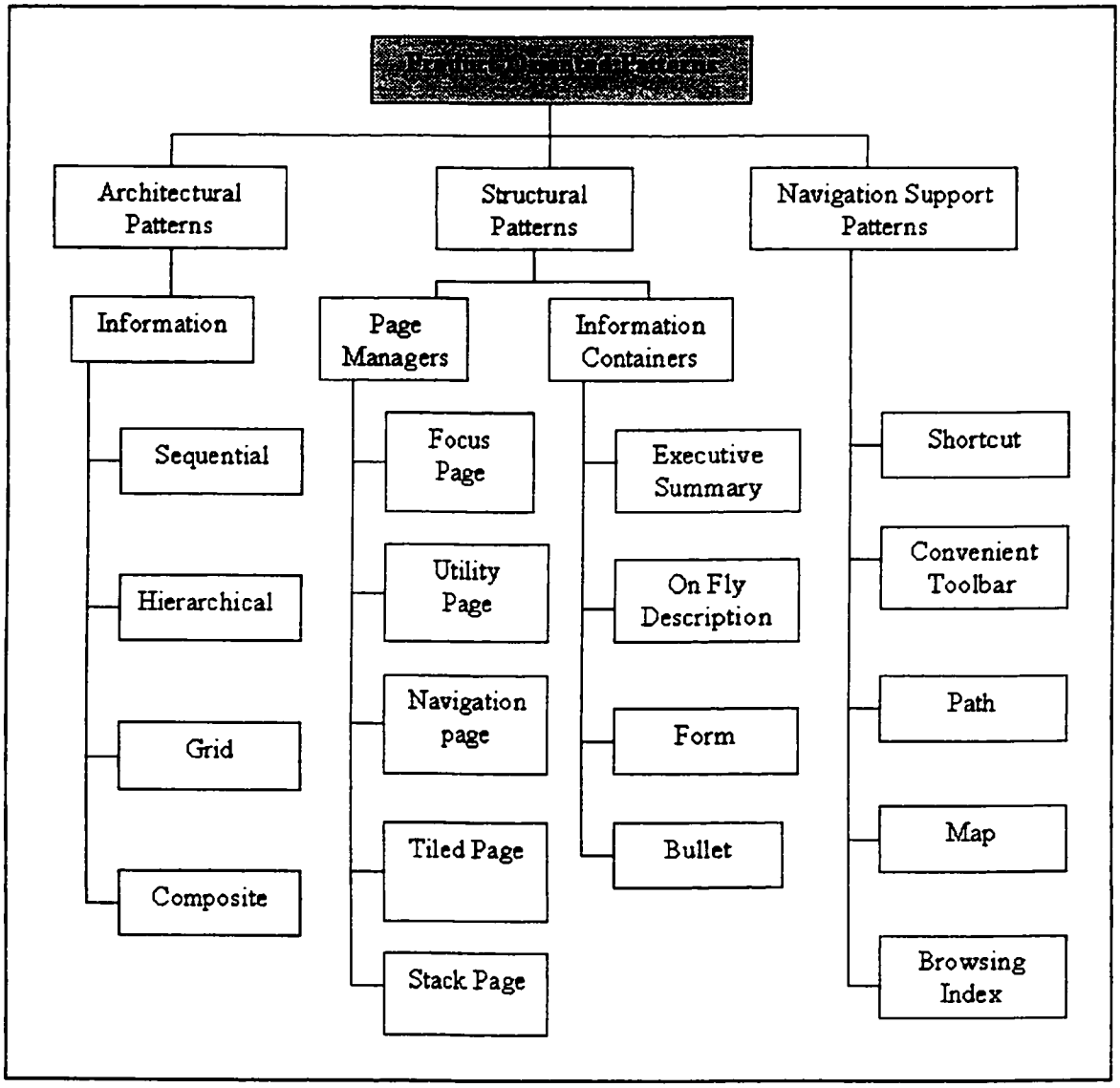


Figure 3.2: An Overview of UPADE Web Language

So far, we have collected and identified 18 product-oriented patterns. The above figure 3.2 depicts the hierarchical structure of product-oriented patterns in UPADE Web language. Product-oriented patterns (Product patterns) are a set of specific and functional patterns that include the following categories:

1) **Architectural Patterns**: Describe different schemes for organizing the content of a Web application.
2) **Structural Patterns**: Define the physical and logical layout of commonly used Web pages and also suggest how to group information in a cognitively respectable structure and to display them.
3) **Navigation Support Patterns**: Provide navigation between a set of pages and chunk of information.

3.1.2 Patterns Definition

We exploited IBM Web design guidelines [IBM, 2001a] to show how our patterns are used. Figure 3.3 illustrates the design steps and the corresponding activities in IBM-UPADE design process.

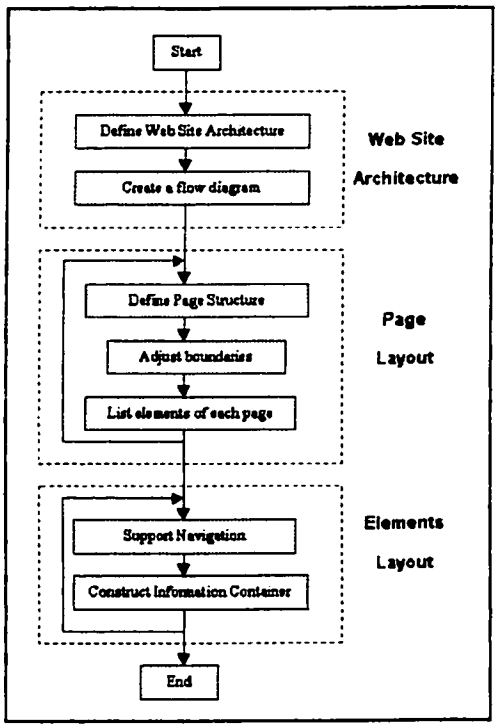


Figure 3.3: An Overview of IBM-UPADE Design Process

1.Web Site Architecture

The first step of IBM-UPADE design process intends to define the general information architecture of the Web site. It includes two major activities:

1) Define Web Site Architecture: On the basis of the collected information from the user and task analysis, the designer can pick up an appropriate pattern from the Architectural Pattern category for the Web site architecture.

2) Create a Flow Diagram: A flow diagram can identify all Web pages within the Web site and show the pathways linking each page.

2. Page Layout

This design step aims to define the physical and logical structure of Web pages. It includes the following three major activities:

1) Define Page Structure: According to the usage and function of a page, the designer can apply a suitable Page Managers pattern of Structural Pattern category into the Web page structure.

2) Adjust Boundaries: Due to diversified screen resolution settings and monitor sizes, the designer should define a dimension to satisfy the "safe area" requirement.

3) List Elements of Each Page: Make an itemized list of the contents of each page then organize the items into categories that distinguish them that will appear on every page, on certain pages or on individual pages only.

3. Elements Layout

During this design step, the designer can arrange specific product patterns to support navigation or information grouping. It includes two major activities:

1) Support Navigation: Select navigation element patterns from Navigation Support Pattern category and locate them on the appropriate place to support navigation between pages.

2) Construct Information Container: To support the organization of chunk of information, the designer can pick up relevant product patterns from the Information Containers Pattern category.

3.1.2.1 Product Oriented Patterns

3.1.2.1.1 Architectural Patterns

A logical site organization allows users to make successful predictions about where to find things. Consistent methods of grouping, ordering, labeling, and graphically arranging information allow users to extend their knowledge from pages they have visited to pages they are unfamiliar with. If the designer misleads users with a structure that is not logical (or has no comprehensible structure at all), users will be constantly frustrated by the difficulties of finding their way around.

This category describes different schemes for organizing the content of a Web application across pages, directories, etc. The goal of most organizational schemes is to keep the number of local variables the reader must keep in short-term memory to a minimum,

using a combination of graphic design and layout conventions along with editorial division of information into discrete units.

Information Architecture Patterns

1) Sequential: The pages are organized in a sequence, where the contents of the pages are presented in a linear narrative (Figure 3.4). Information that naturally flows as a narrative, time line or in logical order is ideal for this pattern.

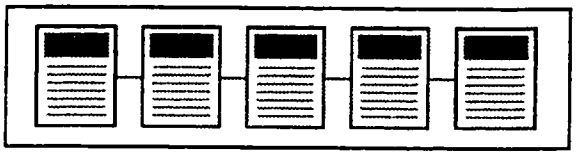


Figure 3.4: A Diagram of Sequential Pattern

We observed this pattern covers the "Sequence" guideline of Yale guidelines [Lynch & Horton, 1999]. It is similar to "Step by Step Instructions" pattern in Common Ground [Tidwell, 1999].

2) Hierarchical: All the pages are organized in a hierarchical model. The user can easily go through from the most general overview of the Web site, such as home page, down to the most specific or optional topics. The pattern contains a "Hierarchy" guideline stemming from Yale guidelines and is similar to "Hierarchical Set" pattern in Common Ground. Figure 3.5 illustrate the diagram of Hierarchical Pattern.

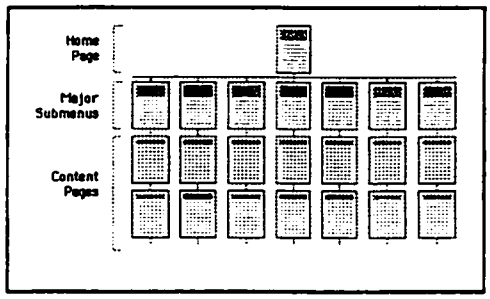


Figure 3.5: A Diagram of Hierarchical Pattern

3) Grid: This pattern organizes the pages into two dimensions (Figure 3.6). The contents of the pages must share a highly uniform structure of topics and sub-topics. This pattern covers the "Grid" guideline suggested in Yale guidelines.

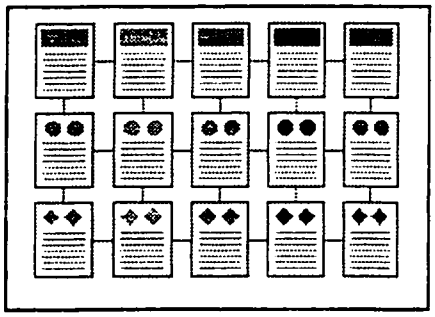


Figure 3.6: A Diagram of Grid Pattern

4) Composite: A complex and large Web application is generally organized using a combination of several architectural patterns (Figure 3.7).

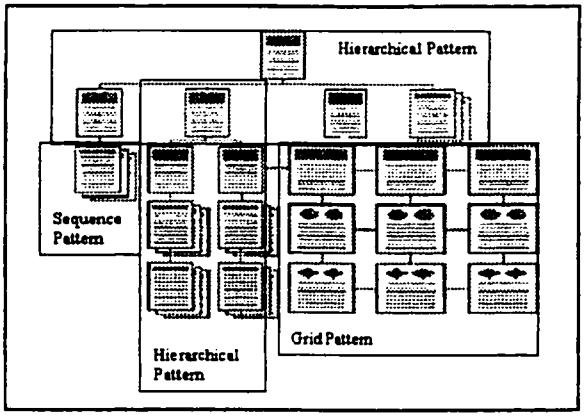


Figure 3.7: A Composite Pattern combining Sequential Pattern, Hierarchical Pattern and Grid pattern

3.1.2.1.2 Structural Patterns

This set of patterns establishes a consistent physical and logical screen layout of pages. It allows a designer to quickly "plug in" chunk of information for pages without having to stop and rethink the basic design approach for each new page. Patterns of this category also suggest different ways for presenting, grouping and organizing information. This category includes two kinds of patterns: Page Managers patterns and Information Containers patterns.

1. Page Managers Patterns

1) Focus Page: This pattern aims to help the designer build a Web page that is the fountainhead and center of the Web site. Such a page pattern should balance aesthetics and practicality to attract users from their first glance. Figure 3.8 depicts the diagram of this pattern.

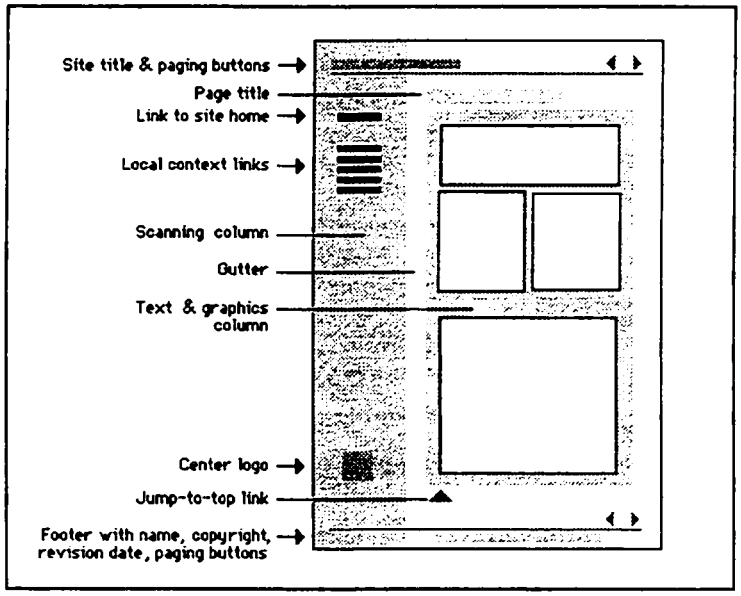


Figure 3.8: Focus Page Pattern

This pattern is similar to "Navigable Space" and "Central Working Surface" patterns collected in Common Ground. Moreover, this pattern can combine the "News" pattern in Hypermedia Design Patterns [HPR, 2001] and "What They See is All They Get" usability pattern defined by Kimberly Perzel and David Kane [Perzel & Kane, 1999]. In addition, this pattern can contain several guidelines suggested in Yale guidelines: "Balanced Pages", "Design Grids for Pages", "Graphics Safe Areas", "Consistency" and "Page Length".

2) Utility Page: This pattern helps the designer organize extra information and provide additional explanation or assistance to the user. Good applications can be bookmarks and help information. The pattern can embrace "Helper Posture" and "Bookmarks" pattern defined in Common Ground.

3) Navigation Page: This pattern groups the information and directs the users to the appropriate information they are looking for and then leads them to the proper page or area. The pattern is similar to "Navigable Space" described in Common Ground and can also contain the guideline "Image Maps" of Yale guidelines.

4) Tiled Page: This pattern provides a paralleled structure of working surfaces and the contents of each surface are related. Thus, it can show the contents to the user from general to specific at the same time. As shown in figure 3.9: Pane "A" for viewing the menu list or catalogues, pane "B" for viewing the brief introduction for that selected catalogue, and pane "C" for viewing detailed information of "B" part.

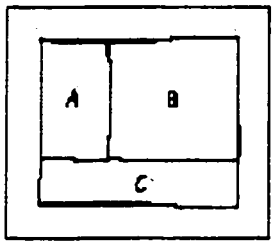


Figure 3.9: A Diagram of Tiled Page Pattern

This pattern stems from "Tiled Working Surfaces" pattern in Common Ground. It is similar to "Overview Beside Detail" pattern in Common Ground, "Garden of Windows" pattern in Experience [Coram & Lee, 1996] and "Container Navigation" pattern in Amsterdam collection [Martijn, 2000].

5) Stack Page: This pattern provides an overlapped structure of working surfaces. The information elements are grouped in separate surfaces and it allows the user to select only one surface at a time. Each surface should be labeled with the name of the category and navigation areas should be placed at the top or left of the surfaces.

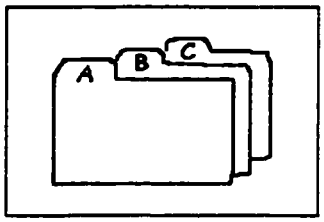


Figure 3.10: A Diagram of Stack Page Pattern

This pattern originates from "Stack of Working Surfaces" pattern in Common Ground. It is also similar to "Navigating Spaces" pattern in Amsterdam Collection and "Garden of Windows" pattern in Experience.

2. Information Containers

1) Executive Summary: This pattern gives users a preview of underlying information before time is spent in downloading and reading large amounts of information. It is an especially critical design issue for small devices. The pattern is similar to "Preview" pattern defined in Amsterdam collection.

2) On Fly Description: This pattern shows a short (one sentence or shorter) description of a target object. When the mouse focuses on one object, the pattern presents the additional information to the users, in the form of clarifying data or explanations of possible actions (Figure 3.11). The pattern is similar to "Short Description" pattern suggested in Common Ground and it is also similar to "Hinting" pattern in Amsterdam Patterns Collection and "Behavior Anticipation" pattern in HPR [HPR, 2001].

Figure 3.11: An Example of On Fly Description Pattern

3) Form: This patterns intention is to collect complete information from the user and it is widely applied in on-line registration and surveys. This pattern is similar to "Form" pattern in Common Ground and it can also contain or implement the following patterns: "Forgiving Text Entry", "Structured Text Entry", "Good Defaults", "Remembered State" in Common ground; "Show the Format Required" pattern in Brighton; "Focus!" pattern

in Amsterdam; "Required Field Marker" pattern and "What They See is All They Get" pattern defined by Kimberly Perzel and David Kane [Perzel & Kane, 1999].

4) Bullet: Bullet pattern collects small amounts or simple information from users. It presents the information into groups and labels the content with bullets. This pattern is similar to "Small Groups of Related Things" pattern and "Choice From A Small Set" pattern defined in Common Ground.

3.1.2.1.3 Navigation Support Patterns

Patterns of this category implement techniques for navigating among a set of pages or a chunk of information.

1) Shortcut: This pattern is normally located on the homepage, which lists all the frequently visited pages in a list box. Thus, the experienced users can directly find their favorite pages by using this pattern. This pattern can contain the other patterns described in Common Ground: "Choice From A Small Set" and "Choice From A Large Set".

2) Convenient Toolbar: This pattern aims to assist the user to reach the most useful and frequently visited pages promptly. The pattern groups the frequently used actions together, labels these actions with meaningful icons or notations and locates them on a consistent place throughout the Web site. The pattern is similar to "Convenient Environment Actions" pattern defined in Common Ground, as well as being similar to "Goal Oriented Areas" pattern in Experience, "List Browser" pattern in Amsterdam and

"Guided Tour" pattern in HPR [HPR, 2001]. Moreover, this pattern contains the guidelines of "Basic Interface Design" and "Links & Navigation" in Yale guidelines.

3) Path: The path pattern indicates the whole path since the user accesses into the Web site. It reduces the user's memory load.

4) Map: This pattern depicts a map of all navigable pages in a Web site. It can support the user to reach any page directly. This pattern is fairly similar to "Map of Navigable Spaces" pattern in Common Ground and it contains patterns such as "High-Density Information Display" pattern and "Tabular Set" pattern defined in Common ground; "Hyper-Map" pattern suggested by D.M. Germán and D.D. Cowan [Germán & Cowan, 1999].

5) Browsing Index: This pattern allows the user to navigate directly from one item to another by using a natural ordering. The ordering is based on a ranking and the ordering criterion is visible to the user. This pattern is similar to "Index Navigation" pattern suggested in HPR [HPR, 2001].

3.2 Format for Documenting Patterns

3.2.1 UPADE Format

In general, HCI patterns are documented using software engineering pattern forms. Most of the HCI pattern forms suggest no concerns on the relationship between patterns

and design process. However, the UPADE Web language emphasizes the designer perspective and the patterns are presented according to the design process. Therefore, the existing pattern forms are not suitable for UPADE Web language. In UPADE Web language, we describe the patterns using the following form:

Usability Pattern Name = **Name**

Alias = **Alias**

Intent = **Intent** /*Applicability and Goals

{

If the user find himself

As for Examples = **Examples**

With Usability Problem = **Problems**

In the Context of Use = **Context**

Entailing Forces = **Forces**

Then Based on the Rationale = **Rationale** /*Justifications and Design Principles

Apply Solution = **Solution**

Use Implementation = **Implementation**

Leading to

Consequence = **Consequence** /* Measurable Usability Attributes

Where Other Usability Patterns Can Apply = **Patterns**

For further information and understanding

Reference = **List_Reference**

}

The following table 3.1 summarizes the language attributes and possible values:

Characteristic	Definition	Attributes and possible values
Name and Alias	Each pattern should have a meaningful name that represents the problem it is addressing. We entitle the pattern according to Alexander's name principles.	
Intent	This section is an abstract, which describes the goals and the applicability of the usability patterns.	
Context	It defines the context in which the problem occurs including user characteristics, tasks and technical, physical and organizational environment.	- **User** = {Novice, Intermediate, Expert, Occasional} - **Task** = { Duration, Frequency, Flexibility } - **Workplace** = {Hardware, Software, User Posture, Location, Group Working}
Problem	This section describes the usability problems that the pattern attempts to solve within the given context and constraints of the problem.	
Forces	The notion of force generalizes the kinds of criteria that we use to justify the usability of a product. Patterns deal with the larger, harder-to-measure, and conflicted sets of goals and constraints encountered in the development of every artifact you ever create.	
	This section gives instance(s) of situations where the pattern is used. Examples help usability	

Examples	engineers understand the scope and domain of applicability of the pattern. This also enforces the fact that the pattern describes a proven solution. Examples can be provided in several ways: prose, diagrams, pictures (hand sketches or photographs) that illustrate the use of the pattern.	
Rationale	This section describes, in a solution-independent manner, the reasoning (design rationale) behind and suitability of the pattern as a justified choice towards solving the usability problem in different context of use. The rationale assists a usability engineer in making an appropriate choice by describing how and why the pattern works, with an insight into the internal structure and key mechanisms of the system.	
Solution	This section describes the actual solution provided by the pattern to solve the usability problem. It describes the solution approach briefly and the solution elements, which identify the pattern's structure, presentation, logic and behavior.	
Implementation	This section includes: -**Structure**: A description of a pattern at a high level abstraction using a graphical notation. It is supplemented by a detailed explanation of the participants and collaborations. - **Strategies**: Describe different ways a pattern can be implemented.	

Consequences	This section describes usability-related impact and trade-off from the application of the pattern. Consequences should refer to the relevant factors and criteria that we used to justify the usability of a design solution.	Identified measurable aspects of Usability: {Learnablity, Task Completion, Error Analysis, Performance, Satisfaction} Other usability factors and criteria: **Factors** = {Efficiency, Effectiveness, Satisfaction, Productivity, Safety, Accessibility, Universality} **Criteria** = {Understandability, Operability, Aesthetics, Compliance, Consistency, Flexibility, Minimal Action, Minimal Memory load, Guidance, Accuracy, Completeness, Required Resources, Helpfulness, Controllability}
Related Patterns	This section gives all the related patterns, which are either super-ordinate, sub-ordinate, competitor, or neighboring patterns. **Super-ordinate pattern**: It is the superior of the described pattern, which can contain the target pattern and other patterns. **Sub-ordinate pattern**: It can be embedded into the described pattern. **Competitor pattern**: It can provide the same or competitive function as the described pattern and cannot be used together with the described pattern. **Neighboring pattern**: It belongs to the same pattern category as the described pattern.	

References	This section provides references (with an annotated bibliography, if necessary) for further understanding about the design principles underlying the pattern as well the pattern itself.	

Table 3.1: UPADE Web Language Attributes and Possible Values

3.2.2 Examples

In this thesis, we haven't described all the sections of the pattern form for each UPADE pattern. We utilize a simple version of UPADE format for documenting patterns. Here, we extract three specific patterns from each category. The following table 3.2 describes Hierarchical Pattern.

Identification	**Name**	**HIERARCHICAL**
	Category	Product-Oriented Patterns→ Architectural Patterns→ Information Architecture Patterns
Context	User	Novice and expert
	Task	Tasks are structured into a hierarchy. All the sub-tasks stem from one original center.
	Workplace	Web applications
Problems	- The user can easily go through from the most general overview of the Web site, such as home page, down to the most specific or optional topics. - Have more flexibility than sequence structure.	
Forces	- Efficiency - Effectiveness - Satisfaction - Understandability - Completeness - Flexibility	

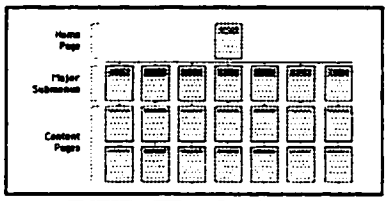
Examples		Most of the current Web sites are utilizing the hierarchical structure.
Rationale	- Causality - Population stereotypes	
Solution	- All the pages are organized in a hierarchical cascade model. The sub-branches expend from one generic center. There is no intersection among sub-branches. - Certain constraints should be applied on the width, depth of the structure.	
Patterns	Super-ordinate	Composite
	Sub-ordinate	Focus Page, Utility Page, Navigation Page, Tiled Page, Stack Page
	Neighboring	Sequential, Grid
	Competitor	Sequential, Grid
References	- "Hierarchy" guideline in Yale guidelines. - "Hierarchical Set" pattern in Common Ground.	

Table 3.2: Hierarchical Pattern

Table 3.3 describes Focus Page Pattern:

Identification	**Name**	**FOCUS PAGE**
	Category	Product-Oriented Patterns→ Structural Patterns→ Page Managers Patterns
Context	User	Novice and expert
	Task	The Web page is the fountainhead and center of a Web site. It must balance aesthetics and practicality to attract the user, especially the novice, at first glance.
	Workplace	Web page
Problems	- The novice shows interests on the Web site and may be willing to continue exploring. - The expert user can find the useful information easily and reach the target topics promptly.	

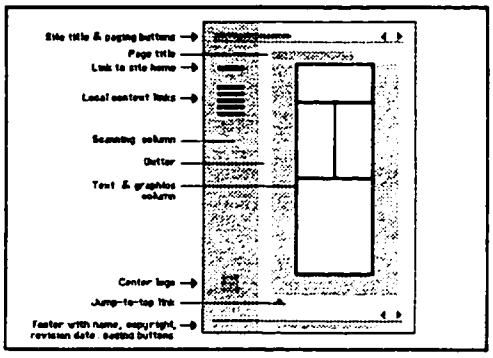
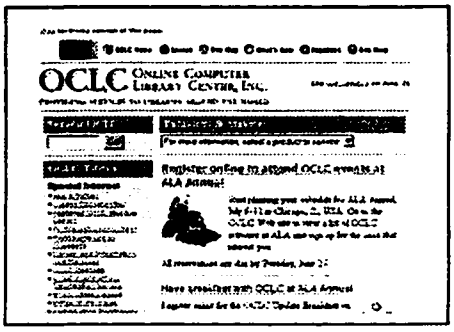
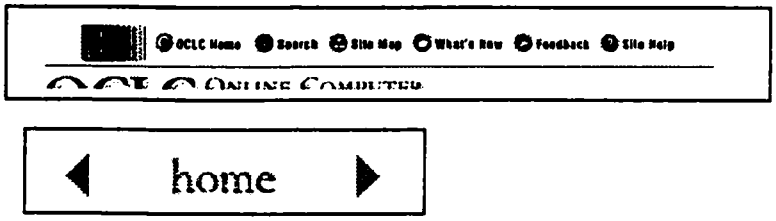
<table>
<tr><td>Forces</td><td colspan="2">- Efficiency
- Effectiveness
- Understandability
- Attractiveness</td></tr>
<tr><td>Examples</td><td colspan="2">   </td></tr>
<tr><td>Rationale</td><td colspan="2">- Affordance
- Causality
- Transfer effects
- Population Stereotype</td></tr>
<tr><td>Solution</td><td colspan="2">- Deploy a utility toolbar, an index of major topics, a shortcut menu, push information sections and maintenance support information section on the Web page.
- Apply dimension constraints on the width and length of a page.
- Avoid frames, big size images, irrelevant applets, blinking texts and banners.</td></tr>
<tr><td rowspan="4">Patterns</td><td>Super-ordinate</td><td>Sequential, Hierarchical, Grid, Composite</td></tr>
<tr><td>Sub-ordinate</td><td>Executive Summary, On Fly Description, Shortcut, Convenient Toolbar, Browsing Index</td></tr>
<tr><td>Neighboring</td><td>Tiled Page, Stack Page</td></tr>
<tr><td>Competitor</td><td>Tiled Page, Stack Page</td></tr>
<tr><td>References</td><td colspan="2">- “Navigable space” and “Central Working Surface” patterns in Common Ground.
- “News” in HPR (HDPR, 2001)
- “What they see is all they get” defined by Kimberly Perzel and David Kane (Kimberly et al., 1999)
- “Balanced pages”, “Design grids for pages”, “Graphics safe areas”, “Consistency” and “Page length” in Yale Guidelines.</td></tr>
</table>

Table 3.3: Focus Page Pattern

Table 3.4 illustrates Convenient Toolbar Pattern.

Identification	**Name**	**CONVENIENT TOOLBAR**
	Category	Product-Oriented Patterns→ Navigation Support Patterns
Context	User	Expert
	Task	Assist the user to reach the most useful and frequently visited pages at any time throughout the Web site.
	Workplace	Web site
Problems	- The user can easily find the most commonly used pages regardless of the current state of the artifact. - The user can reach these convenient pages promptly.	
Forces	- Efficiency - Safety - Consistency - Minimal Action - Minimal Memory - User Guidance - Helpfulness	
Examples		
Rationale	- Affordance - Mapping - Causality	
Solution	- Group the most convenient action links, such as home, site map, help and etc. - Utilize meaningful metaphors and accurate phases as labels. - Locate it at the consistent place throughout the Web site.	
Patterns	Super-ordinate	Focus Page, Tiled Page, Stack Page, Navigation Page, Utility Page
	Sub-ordinate	
	Neighboring	Shortcut, Path, Map, Browsing Index
	Competitor	Shortcut

References	- "Convenient environment actions", "Go back to a safe place" in Common Ground. - "Goal oriented areas" in Experience. - "List Browser" in Amsterdam - "Guided tour" in HPR. - "Basic interface design" and "Links & navigation" in Yale guidelines.

Table 3.4: Convenient Toolbar Pattern

Chapter 4: UPADE Editor - A design and Prototyping Tool

In order to make UPADE Web patterns language usable, especially by developers who are not familiar with HCI design as well as novice designers, we embedded the patterns into a tool (UPADE editor). The UPADE editor provides a pattern box containing product-oriented patterns. The designer can not only browse the detailed description of each pattern in UPADE editor, but can also pick up a product pattern from the pattern box and combine patterns to develop low fidelity prototypes.

4.1 Overview of UPADE Editor 1.0

As part of our thesis, we have built the first version of UPADE editor. It is implemented in J.D.K 1.2.

4.1.1 Functionality

UPADE editor 1.0 can fulfill Browse and Design functions as following:

1. Browse:

UPADE editor can provide software developers and pattern engineers with just-in-time details and information on product-oriented patterns. The information is depicted in unified format consisting of pattern name, a description of the pattern, some illustrated diagrams and several practical examples.

The designer can acquire the pattern information at any circumstance whether in "Browse" or "Design" mode of UPADE editor. When UPADE editor is in "Browse"

mode, once the designer clicks on any specific pattern in the Product Pattern box, a pop-up window appears and related just-in-time information is given inside the window. Also, when UPADE editor is in "Design" mode, the designer can still get the information support only if he utilizes the right button of the mouse on the target pattern.

Furthermore, UPADE editor can demonstrate the interrelationship between product patterns and the design process. When the designer selects any step of the design process, the relevant product patterns will be enumerated immediately in the Product Pattern box.

2. Design:

First of all, UPADE editor supports design process starting from a more general to a more specific level. Three different levels are provided to the designer:

- Web Application level: The designer can establish a prototype of a Web application, which characterizes certain architecture containing Web pages and specific elements on each page.

- Web Page level: In this level, the designer can select a suitable Page Managers pattern for the Web page structure, then embeds required page elements into the page to establish a prototype of a Web page.

- Elements level: The designer can pick up certain page element patterns including Navigation Support patterns and Information Containers patterns from the Product Pattern box and combine them to establish a prototype.

Secondly, UPADE editor can illustrate appropriate product patterns in the Product Patterns box according to circumstances. For example, when the designer triggers the section titled "Navigation Area" in a Page Managers pattern, all the Navigation Support patterns will be presented immediately in the Product Patterns box.

Thirdly, UPADE editor supports combination and organization of existing patterns. The designer can embed Page Managers patterns into Information Architecture patterns. (S)he can also insert Navigation Support patterns and Information Containers patterns into Page Managers patterns. Moreover, the designer has the capability to organize Navigation Support patterns and Information Containers patterns within Page Managers patterns. The designer can add, move and even delete the patterns of the former two categories inside the latter patterns. The purpose of these activities is to explore the way that organize and combine the existing patterns to customize and generate the new patterns.

Finally, UPADE editor provides a mechanism to check the usage of patterns. It can automatically examine the compatibility of certain patterns and then give the related instruction to the designer.

4.1.2 Interface

The main interface of UPADE editor is separated into five areas (Figure 4.1)

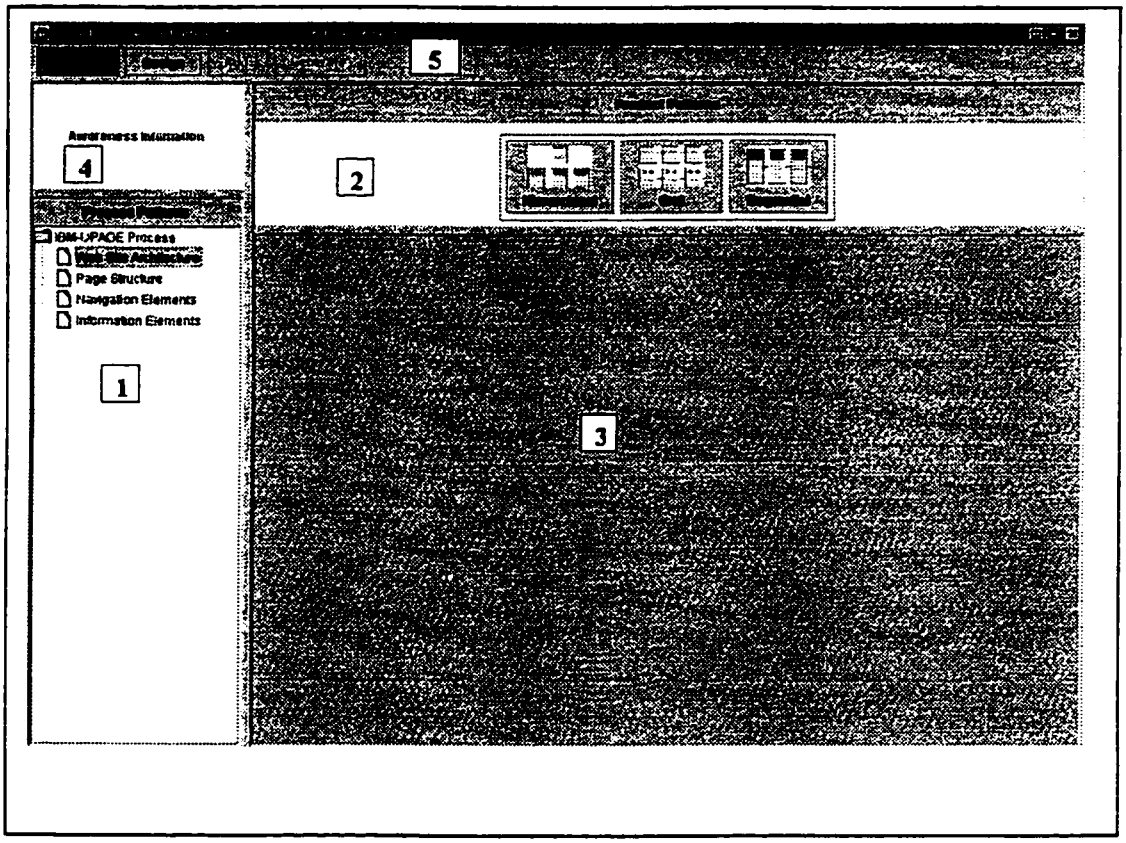


Figure 4.1: Main Interface of UPADE Editor 1.0

1. Process Patterns Box: Illustrates all the suggested Web design processes. The corresponding design steps of each design process are organized into a hierarchical structure.

2. Product Patterns Box: All the product patterns are presented in this area and grouped in accordance with their categories.

3. Working Area: The major operation surface for designers. The designer can fulfil the "Browse" and "Design" activity in this area.

4. Awareness Information Area: This area aims to display constructive suggestions to the designer, in order to indicate to the designer the correct use of UPADE Web patterns. Please note that this function hasn't been implemented completely in the current version of UPADE editor. It will be realized in the future.

5. Button Bar: Composed of four buttons, titled "Browse", "Design", "Add" and "Generate", which represents the functionality of UPADE editor. The two latter buttons are dimmed, because "Add" and "Generate" functions are not intended to be implemented in UPADE editor 1.0

4.1.3 Collected Usability Patterns

In the Process Patterns box of UPADE editor 1.0, we have implemented four major design steps of the IBM-UPADE process. They are Web Site Architecture, Page Structure, Navigation Elements and Information Elements. Each step is tightly related to a certain category of Product patterns.

For the Product Patterns box, we have collected the most representative patterns from each category of product-oriented patterns and presented them inside the box. For example,

- Information Architecture Patterns category: Sequential Pattern, Hierarchical Pattern, and Grid Pattern.

- Page Managers Patterns category: Home Page (Focus Page) Pattern, Tiled Page Pattern, and Stack Page Pattern.

- Information Containers Patterns category: Quick Summary (Executive Summary) Pattern and On Fly Description Pattern.

- Navigation Support Patterns category: Shortcut Pattern, Index (Browsing Index) Pattern, Convenient Actions toolbar (Convenient Toolbar) Pattern, Map Pattern and Path Pattern.

4.2 Scenarios of Using UPADE Editor 1.0

In this section, we offer detailed examples to depict the functionality of UPADE editor.

4.2.1 Browse Function

"Browse" is the default mode when UPADE editor is initiated. As shown in Figure 4.2, the "Browse" button is automatically enabled on the button bar, while the Process Patterns box illustrates all the available design processes such as IBM-UPADE process. Each design process is symbolized by a folder and all the relevant design steps are packed into the folder correspondingly.

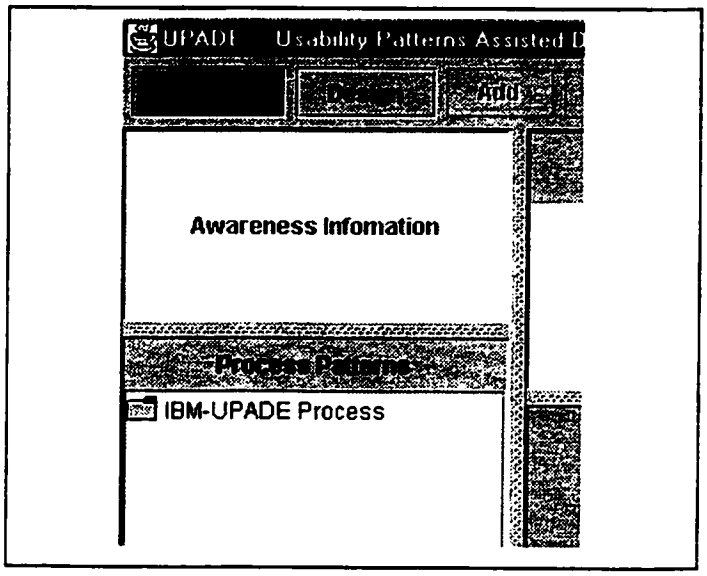


Figure 4.2: "Browse" Mode is Enabled When UPADE Editor is Initiated

Step 1: Select appropriate package of design process

Suppose we decide to choose "IBM-UPADE" design process. A double click on the folder will spread the contents organized into a tree structure. As illustrated in Figure 4.3, four steps of design process are embraced inside this folder.

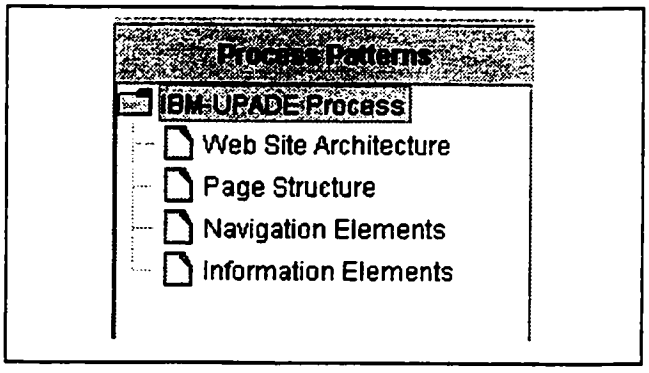


Figure 4.3: Four Design Steps of IBM-UPADE Process

Step 2: Acquire the interrelationship between product patterns and design process steps

When the designer selects any specific step of design process in the Process Patterns box, the corresponding product patterns are enumerated in the Product Patterns box. Each product pattern is represented by a distinct icon and an accurate short notation.

The following four diagrams (Figure 4.4-Figure 4.7) exemplify how the product patterns accord with each step in the Process Patterns box.

This mechanism automatically reveals the interrelationships between steps of design process and product patterns. It reduces the complexity of UPADE WEB language.

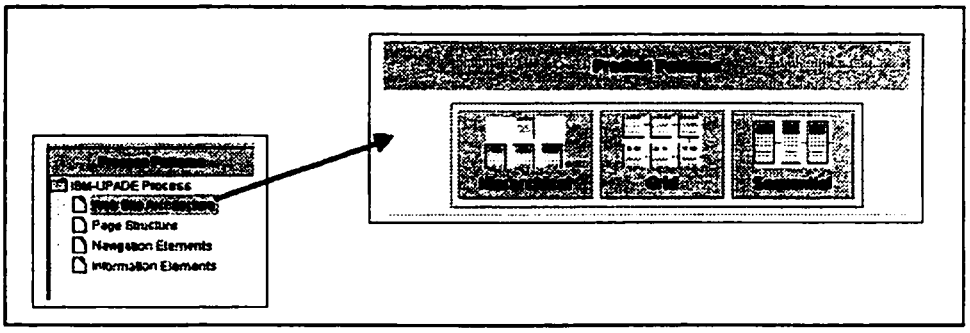


Figure 4.4: "Web Site Architecture" Step Relates to Information Architecture Patterns.

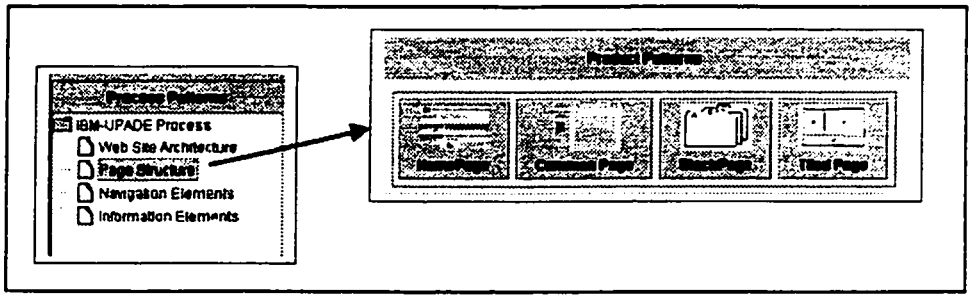


Figure 4.5: "Page Structure" Step Relates to Page Managers Patterns.

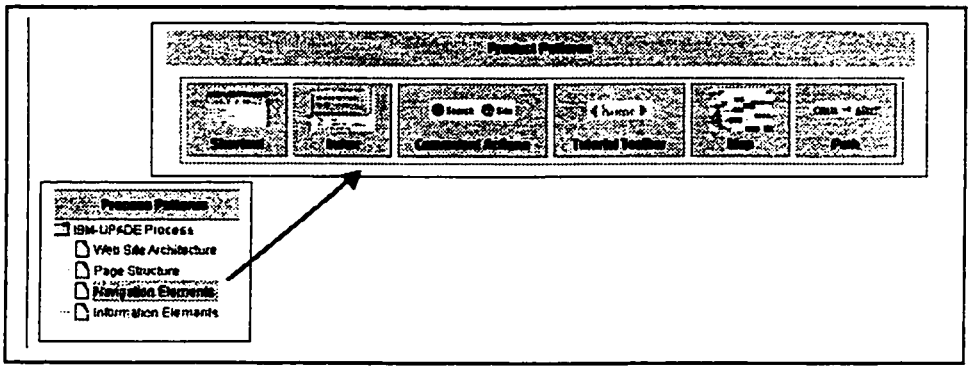


Figure 4.6: "Navigation Elements" Step Relates to Navigation Support Patterns.

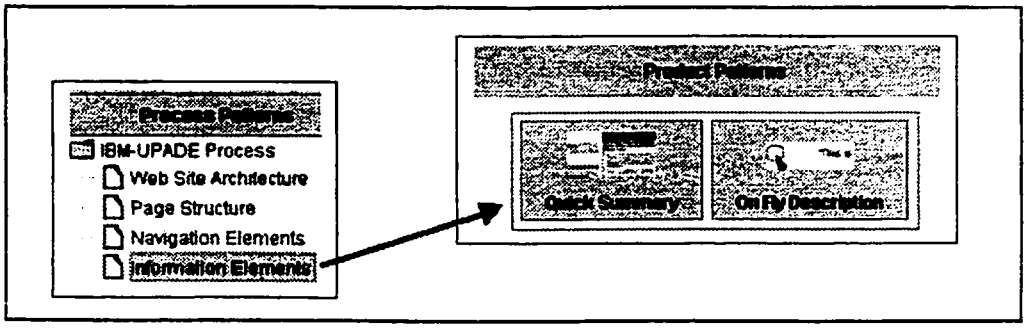


Figure 4.7: "Information Elements" Step Relates to Information Containers Patterns.

Step 3: Browse the pattern information

When the designer (especially the novice designer) wants to acquire the interpretation of certain patterns and to understand how to utilize each pattern properly, (s)he can browse the detailed pattern information from a Pattern Description window. The description window helps the designer reduce the memory load and enhance the understandability of UPADE WEB language. Figure 4.8 shows when the designer clicks on the icon of "Home Page" pattern, a pop-up Pattern Description window appears on the working surface, that contains all the information concerning Home Page pattern, such as the pattern name, the definition of the pattern and one practical example.

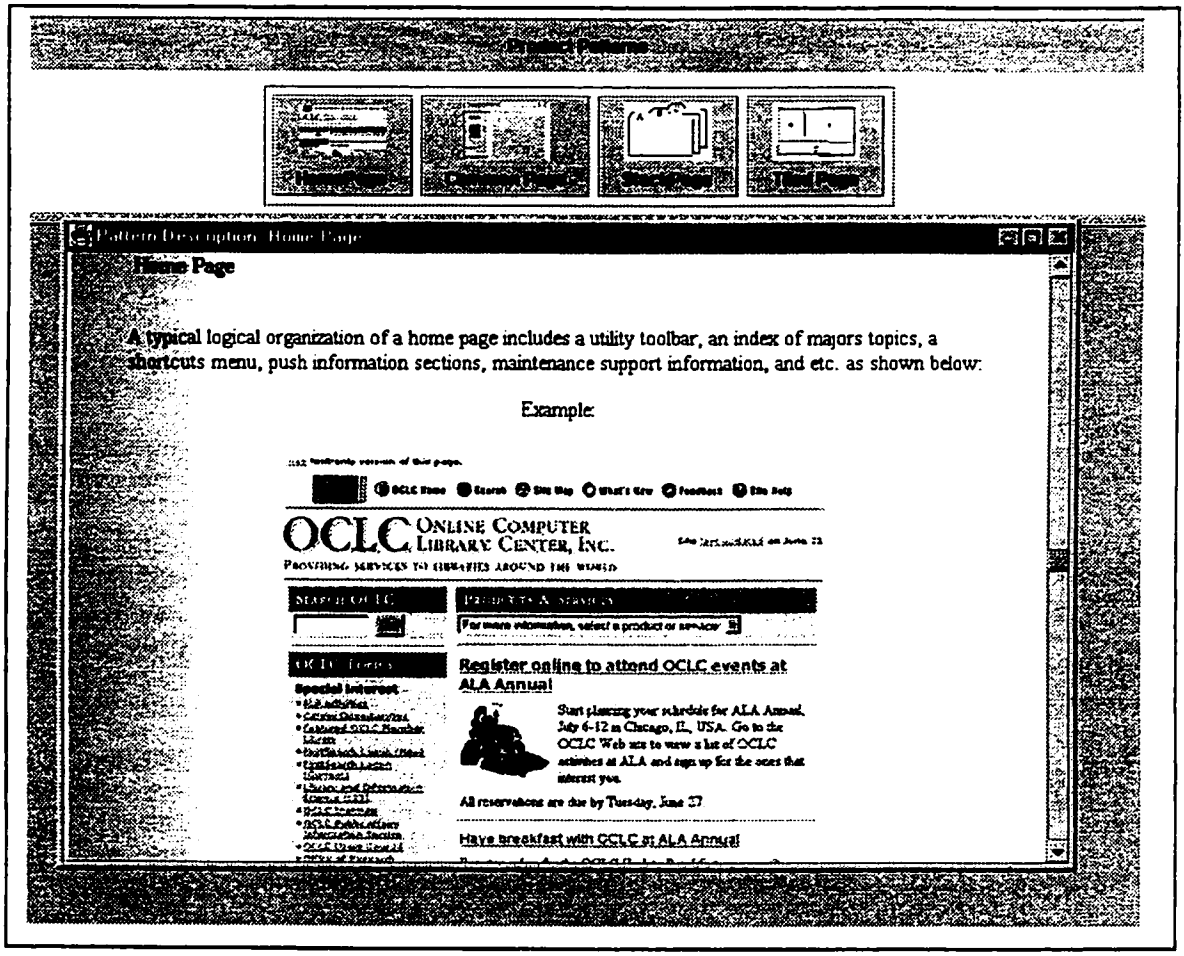


Figure 4.8: An Example of Pattern Description Window for Home Page Pattern.

4.2.2 Design Function

Step 1: Choose a proper design level

Before starting the design process, the designer should above all determine an appropriate design level for his task. For instance, to establish a prototype of a Web application, (s)he can begin the design process from "Web Site Architecture" level; to establish a prototype of a Web page, (s)he needs only to start the process from "Page structure" level. Thus, the designer should select a specific design step from the Process Patterns box to initiate his or her design activity.

As illustrated in Figure 4.9 below, no.2 notation shows that the designer begins his design process from the top level-"Web Site Architecture".

Step 2: Initiate design function

To enable the design function of UPADE editor, the designer should select the "Design" button on the button bar, and then the editor will switch the default "Browse" mode to "Design" mode.

In the following Figure 4.9, no.1 notation shows that the designer clicks on the "Design" button to initiate the design function. No.3 depicts the subsequent image: A blank operation surface appears in the Working Area, while the Product Patterns box enumerates all the Information Architecture patterns.

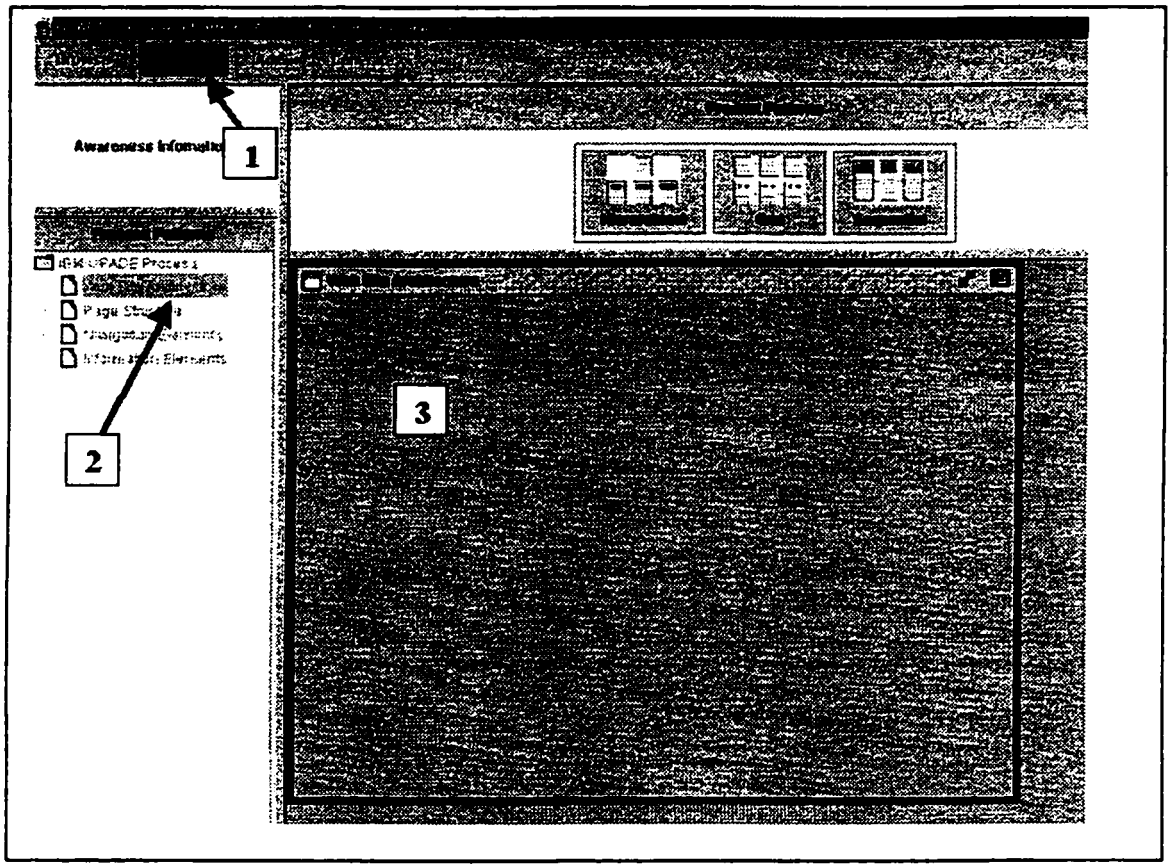


Figure 4.9: An Example of Initiating Design Function

Step 3: Apply an appropriate Information Architecture Pattern

The design process starts with the description of the information architecture. We defined three basic information architecture patterns for organizing the content of a Web application and all these patterns are illustrated in the Product Patterns box. The designer should choose one pattern to suit the Web application. Here we use Hierarchical Architecture pattern as an example, which is the most popular scheme in Web sites because Web sites should always be organized as offshoots of a single home page.

As shown in Figure 4.10, once the designer picks up the Hierarchical pattern from the Product Patterns box, a tree-based hierarchical diagram is portrayed on the operation surface immediately.

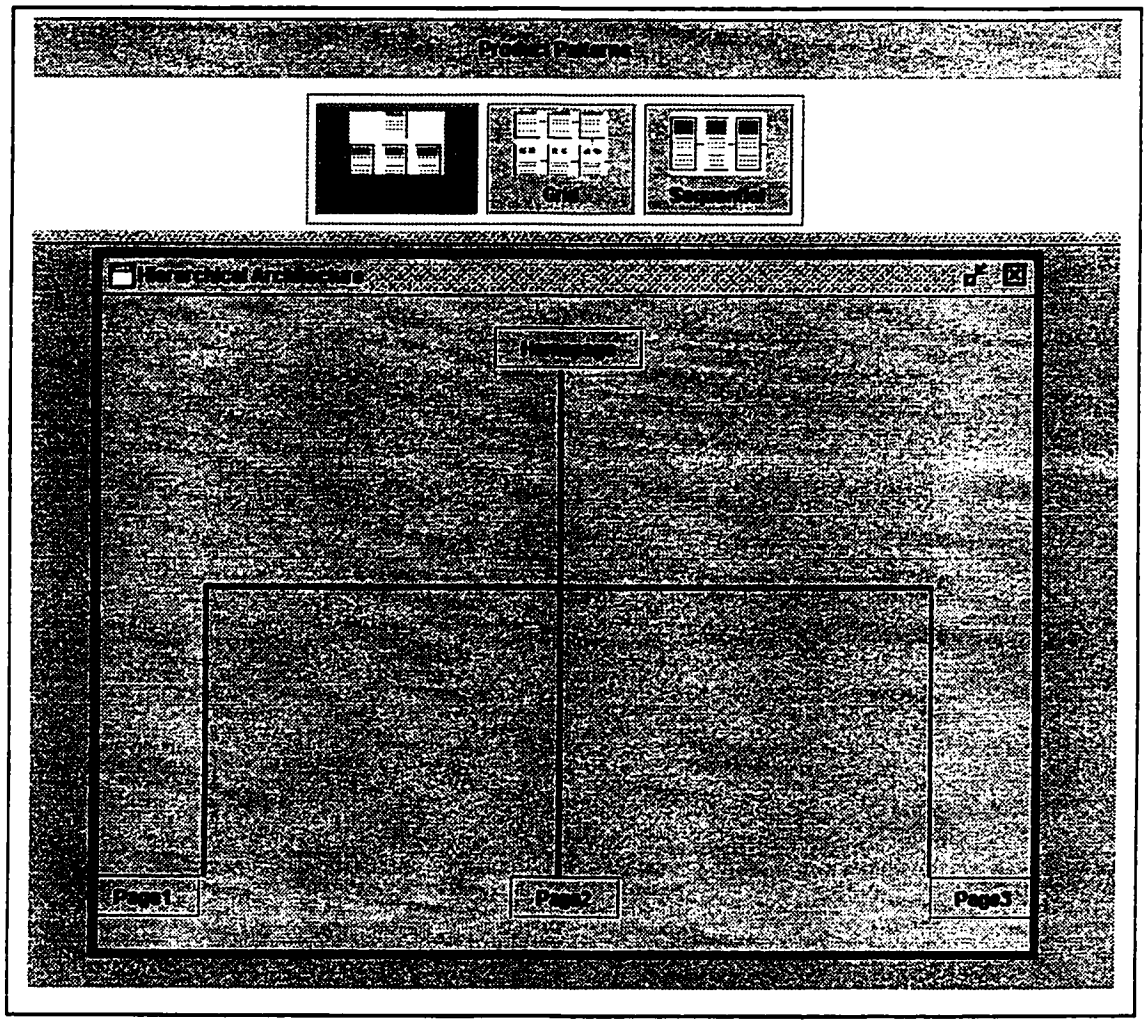


Figure 4.10: An Example of Applying Hierarchical Architecture Pattern in a Web Application

Step 4: Apply Page Managers patterns

The next step of design process is to apply the page structure in establishing a consistent physical and logical screen layout. In the previous picture (Figure 4.10), once the

designer triggers any node of the hierarchy tree in the operation surface, a blank surface will be created for further development on the Web page. Meanwhile, the Information Architecture patterns in the Product Patterns box will be replaced by Page Managers patterns accordingly, which contains Home Page pattern, Common Page pattern, Stack Page pattern and Tilted Page pattern (Figure 4.11).

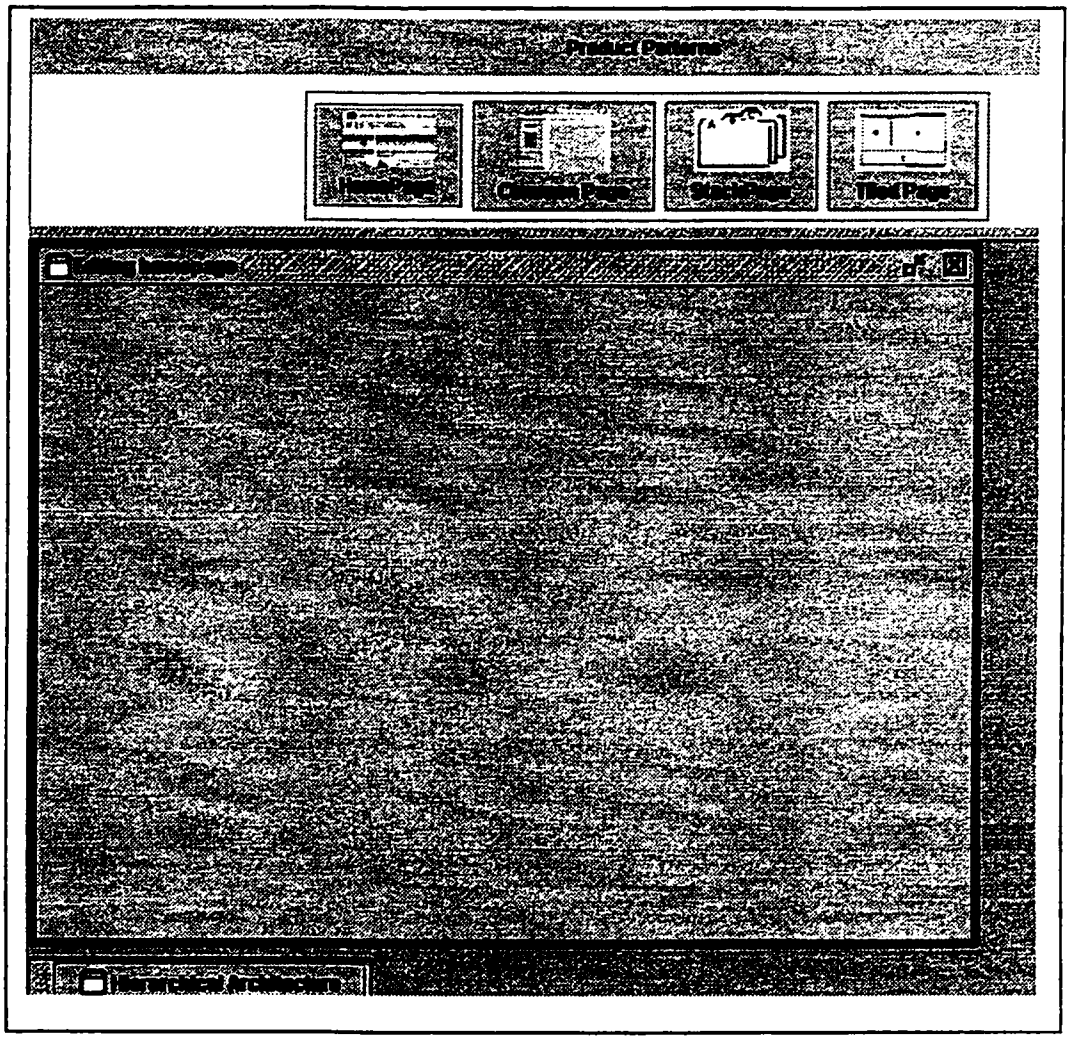


Figure 4.11: A Blank Surface is Created for Further Web Page Design, Along with the Refreshed Page Managers Patterns in the Product Patterns Box.

The designer should then afterwards choose a specific Page Managers pattern for the Web page from the Product Patterns box. Suppose the designer selects the "Home Page" pattern for the current Web page. Figure 4.12 illustrates the consequences of applying a Home Page pattern: The newly page operation surface is composed of Navigation area and Information area.

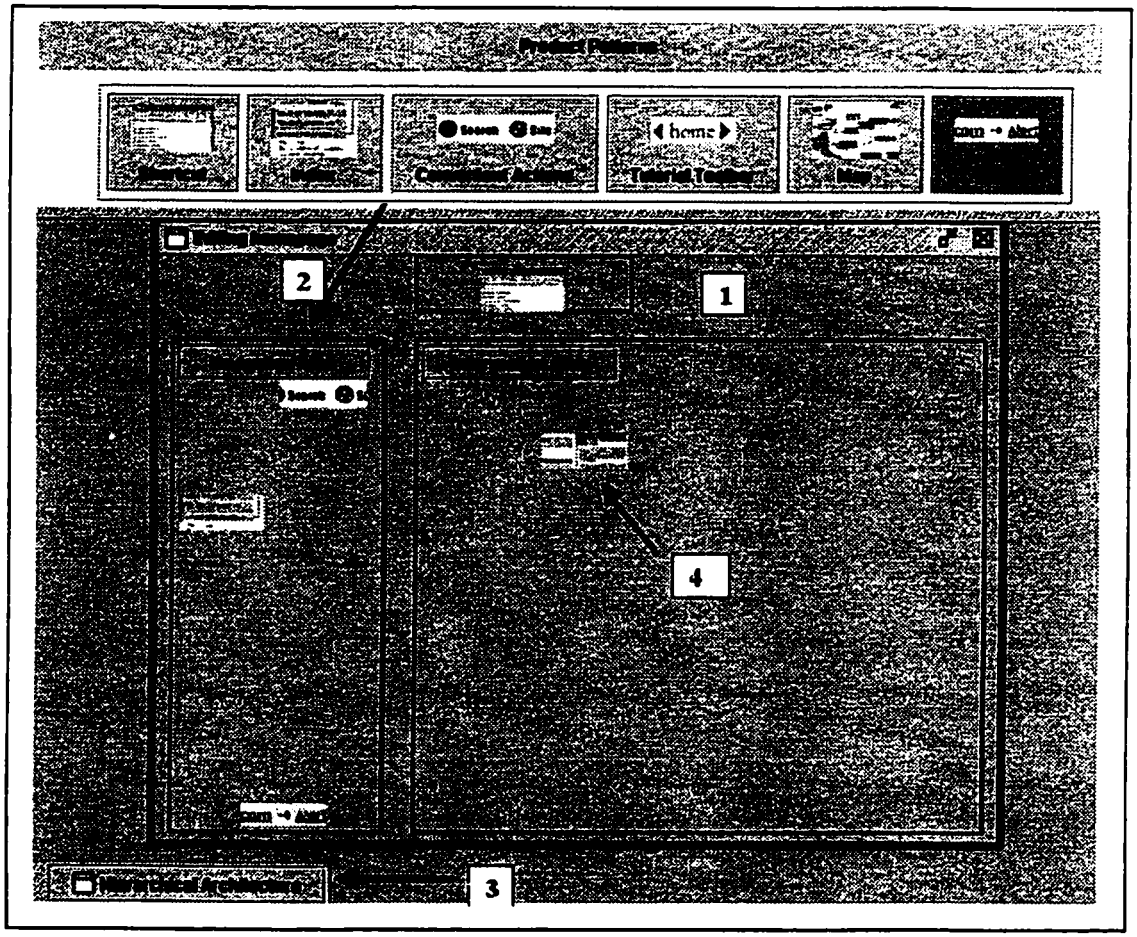


Figure 4.12: A Web Page Combining Home Page Pattern, Information Containers Patterns and Navigation Support Patterns.

Step 5: Embed Information Containers and Navigation Support patterns

The last step of the design process is to embed Information Containers and Navigational Support patterns into the Web page. When the designer triggers the Navigation Area or Information Area on the page operation surface, the corresponding Navigation Support patterns or Information Containers patterns will refresh the Product Patterns box. The designer can then select a specific pattern from the Product Patterns box, combining with other patterns, to form a Web page (Figure 4.12). In addition to add Navigation Support patterns and Information Containers patterns to the Web page, the designer has the capability to move any of these patterns or even delete useless ones.

4.2.3 Other Features Demonstration

UPADE editor supports the designer in looking through a Web application at different levels in that the designer can have an overview of the Web application, while (s)he can also acquire detailed information of each Web page. As pointed by no.3 notation in the above Figure 4.12, by enlarging the minimized architecture surface, the designer can get the whole structure of the Web application.

Moreover, UPADE editor provides a mechanism to check the usage of certain patterns. It can automatically verify the compatibility of the patterns to the circumstances according to their contexts of use. For example, if a novice designer is not familiar with the attributes of Shortcut pattern, (s)he might decide to add a Shortcut pattern into a common page instead of the home page. However, with our checking mechanism, UPADE editor will kindly inform the designer that it is not appropriate for the task (Figure 4.13).

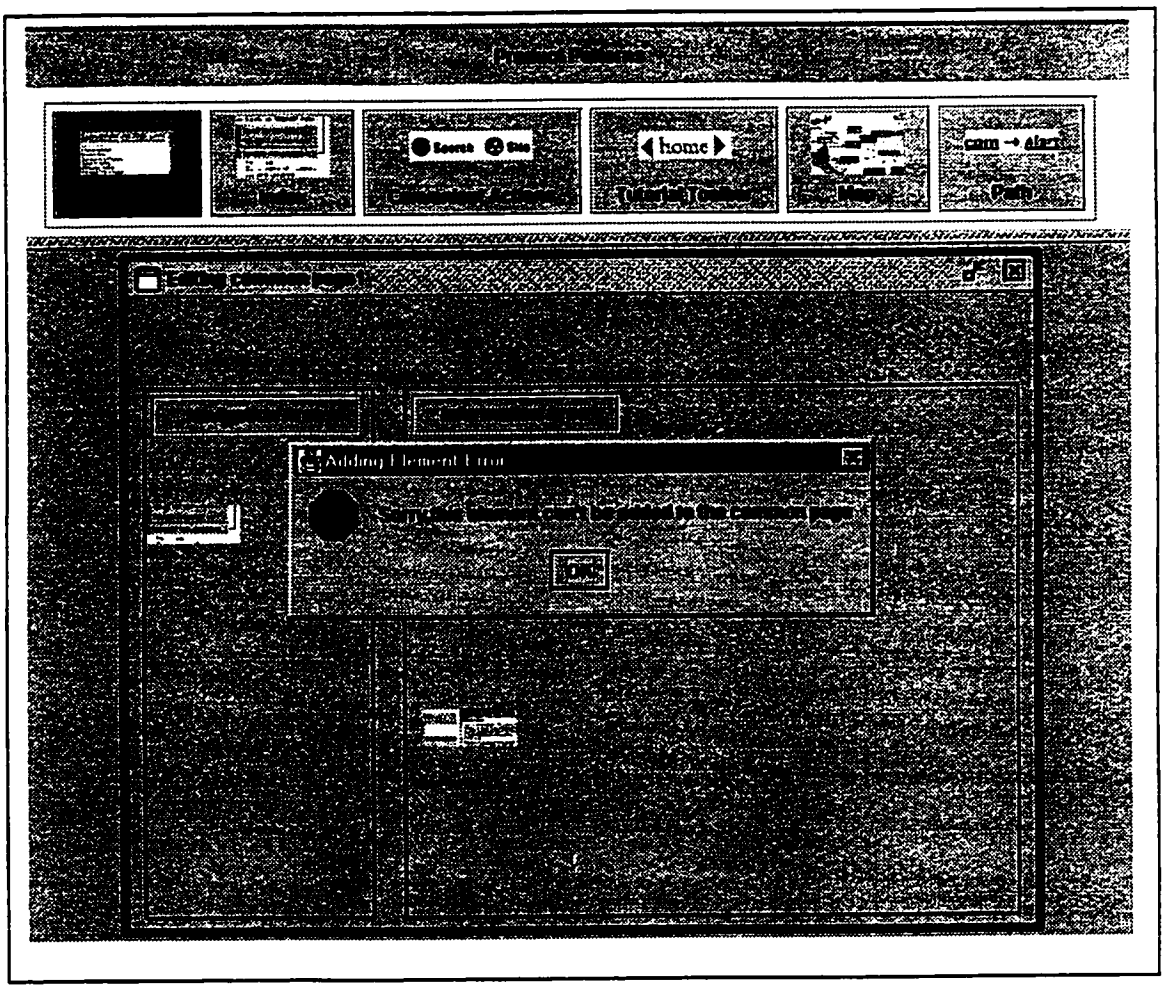


Figure 4.13: An Example of Pattern Compatibility Check

Chapter 5: Conclusion and Future Investigations

5.1 Conclusion

In this thesis, we explored a usability patterns-assisted design approach for Web user interfaces. We developed a usability pattern language for Web applications (UPADE Web language) and an editor tool (UPADE editor) to support that language.

Our investigations revealed that Web design and prototyping tools, such as Web scripting, programming tools and low fidelity prototyping tools, are not concerned with usability issues. We also found that usability patterns are an alternative design approach to current Web guidelines and Web authoring tools. However, we discovered that the current pattern languages or collections for user interface design such as Common Ground, Experience, Brighton Usability Pattern Collection and Amsterdam Collection, unfortunately, are described in natural languages. It is hard for developers especially the novice designers to use these patterns in real design activities effectively and efficiently. Furthermore, none of the existing usability pattern languages are dedicated to Web application design.

This limitation motivated us to define and develop a usability pattern language for Web applications (UPADE Web language) and a tool to support pattern-oriented design (UPADE editor). We defined 18 usability patterns and these patterns are categorized into Architectural Patterns, Structural Patterns and Navigation Support Patterns. In addition to

browsing and searching, UPADE editor allows a Web designer to combine existing patterns and to create a new user interface.

5.2 Further Investigations

The UPADE Web language and the editor presented here are part of a long-term research project. The expectation is that UPADE will first enable clear communication between usability experts, user interface pattern writers (and ultimately pattern users), and second will enable CASE tool support for design patterns, permitting the novice designer (pattern user) to operate at a higher level of abstraction when building multi-platform and device-independent design.

As a next step, we suggest the implementation of usability patterns using an XML-based markup language. XML facilitates the generation of a complete and operational user interface. Meanwhile, to make the outcome of our research beneficial to the industrial world, Web designers, software developers and pattern researchers have to test and evaluate the UPADE language for Web applications and UPADE IDE, including the UPADE editor.

References

[Alexander, 1964] Alexander, Christopher, **"Notes on the Synthesis of Form"**, Harvard University Press, 1964.

[Alexander, 1975] Alexander, Christopher, **"The Oregon Experiment"**, Oxford University Press, 1975.

[Alexander et al., 1977] Alexander, Christopher, Ishikawa, S., Silverstein, M., Jacobson, M., Fiksdahl-King, I., & Angel, S. A. **"A Pattern Language"**. New York: Oxford University Press. 1977.

[Alexander, 1979] Alexander, Christopher, **"The timeless way of building"**. New York: Oxford University Press, 1979.

[Ambler, 1998] Scott W. Ambler, **"An Introduction To Process Patterns"**, 1998. http://www.ambysoft.com/processPatternsPaper.html

[Apple Inc., 1992] Apple Computer, Inc. (1992) **"Macintosh Human Interface Guidelines"** Addison-Wesley Publishing Company.

[Bayle et al., 1998] Bayle, E., Bellamy, R., Casaday, G., Erickson, T., Fincher, S., Grinter, B., Gross, B., Lehder, D., Marmolin, H., Potts, C., Skousen, G. & Thomas, J. **"Putting It All Together: Towards a Pattern Language for Interaction Design. Summary Report of the CHI '97 Workshop"** SIGCHI Bulletin, ACM, January, 1998. http://www.pliant.org/personal/Tom_Erickson/Patterns.WrkShpRep.html

[Brighton UK, 2001] Brighton UK, **"The Brighton Usability Pattern Collection"**, The Usability Group at the University, 2001. http://www.it.bton.ac.uk/cil/usability/patterns/

[Budinsky et al., 1996] F. J. Budinsky, M. A. Finnie, J. M. Vlissides, and P. S. Yu **"Automatic code generation from design patterns"**, IBM Vol. 35, No. 2, 1996 - Object technology.
http://www.research.ibm.com/journal/sj/budin/budinsky.html

[Carlow Inc., 1992] Carlow International Incorporated (1992) **"NASA Human-Computer Interface Guidelines"**.
http://groucho.gsfc.nasa.gov:80/Code_520/Code_522/Documents/HCI_Guidelines/

[Casaday, 1997] George Casaday, **"Notes on a Pattern Language for Interactive Usability"**, Marcam Corporation, CHI 97.
http://www.acm.org/sigchi/chi97/proceedings/short-talk/gca.htm

[Coplien, 1995] James O. Coplien, **"A Generative Development-Process Pattern Language"**, PLoPDI, 1995

[Coplien, 1996] James O. Coplien, **"Software Patterns"**, Bell Laboratories, Naperville, Illinois, SIGS Books ISBN 1-884842-50-X, 1996.
http://hillside.net/patterns/definition.html
http://www1.bell-labs.com/user/cope/Patterns/WhitePaper/SoftwarePatterns.pdf

[Coram & Lee, 1996] Todd Coram and Jim Lee, **"Experiences -- A Pattern Language for User Interface Design"**, PloP conference, 1996.
http://www.maplefish.com/todd/papers/experiences/Experiences.html

[Florijn et al., 1997] Gert Florijn, Marco Meijers, Pieter van Winsen **"Tool support for object-oriented patterns"** Utrecht University ECOOP'97, pp.472.
http://www.serc.nl/people/florijn/work/patterns.html

[Gamma et al, 1994] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides, **"Design Patterns Elements of Reusable Object-Oriented Software"**, Published

October 1994, ISBN 0-201-63361-2.

[Germán & Cowan, 1999] D.M. Germán, D.D. Cowan, DCS, Waterloo, Canada, **"Three Hypermedia Design Patterns"**, Proceedings of the HT99 Workshop on Hypermedia Development.
http://www.eng.uts.edu.au/~dbl/HypDev/ht99w/submissions/GermanHT99Workshop.pdf

[Granlund & Lafrenière, 1999] Åsa Granlund, Daniel Lafrenière, **"PSA-A Pattern-Supported Approach to the User Interface Design Process"**, June 1999.
http://www.gespro.com/lafrenid/PSA.pdf

[HPR, 2001] **Hypermedia Design Patterns**, an initiative of ACM-SIGWEB in collaboration with the University of Italian Switzerland, 2001.
http://www.designpattern.lu.unisi.ch/PatternsRepository.htm

[IBM, 2001a] **IBM-ease of use-web design guidelines-Design**, 2001 July.
http://www-3.ibm.com/ibm/easy/eou_ext.nsf/Publish/574

[IBM, 2001b] **IBM Web Design Guidelines**, 2001 July.
http://www-3.ibm.com/ibm/easy/eou_ext.nsf/Publish/572

[Kant et al., 1998] Maarten van de Kant, Stephanie Wilson, Mathilde Bekker, Hilary Johnson and Peter Johnson. **"PatchWork: A Software Tool for Early Design"**. In Proceedings of Human Factors in Computing Systems: CHI'98. Los Angeles, CA. pp.221-222, April 18-23 1998.

[Lea, 1993] Lea, D. **Christopher Alexander: An Introduction for Object-Oriented Designers**. 1993.
http://g.oswego.edu/dl/ca/ca/ca.html

[Lin et al., 2000] James Lin, Mark W. Newman, Jason I. Hong, and James A. Landay. **"DENIM: Finding a tighter fit between tools and practice for web site design"**. CHI Letters: Human Factors in Computing Systems, CHI 2000, 2000. 2(1): pp. 510-517.
http://guir.berkeley.edu/pubs/denim/denim-chi-2000.pdf

[Lynch & Horton, 1999] Patrick J. Lynch, Sarah Horton, " **Web Style Guide"**, Yale University Press; ISBN: 0300076754, March 1999.
http://info.med.yale.edu/caim/manual/index.html

[Mahemoff & Johnston, 1998] Mahemoff, M. J. and Johnston, L. J. **"Pattern Languages for Usability: An Investigation of Alternative Approaches"** In Tanaka, J. (Ed.), Asia-Pacific Conference on Human Computer Interaction (APCHI) 98 Proceedings, 25-31. Los Alamitos, CA: IEEE Computer Society. In Shonan Village, Japan, July 15-17, 1998.
http://www.cs.mu.oz.au/~moke/papers/candidate/

[Martijn, 2000] Martijn van Welie , **"The Amsterdam Collection of Patterns in User Interface Design"**, April 2000.
http://www.cs.vu.nl/~martijn/patterns/index.html

[Martijn et al., 2000] Martijn van Welie, G.C. van der Veer, A. Eliëns, **"Patterns as Tools for User Interface Design"**: In: International Workshop on Tools for Working with Guidelines, pp. 313-324, 7-8 October 2000, Biarritz, France.
http://www.cs.vu.nl/~martijn/gta/docs/TWG2000.pdf

[Martijn & Hallyard, 2000] Martijn van Welie, Hallyard Traetteberg, **"Interaction Patterns in User Interfaces"**, PloPD 2000.
http://www.cs.vu.nl/~martijn/patterns/PLoP2k-Welie.pdf

[Martijn et al.,1999] Martijn van Welie, van der Veer, G.C., and Eliëns, A. **"Breaking down Usability"**. Human-Computer-Interaction Interact'99 Conference (Edinburgh, 30th August - 3rd September 1999). IOS Press, pp.613–620.

[Meijler et al., 1997] Meijler, T. D., S. Demeyer, and R. Engel, **"Making Design Patterns Explicit in FACE, A Framework Adaptive Composition Environment"**, in Software Engineering Notes, ESEC/FSE, Vol. 22, No 6, Nov 1997, pp.94-110.

[Microsoft, 1987] Microsoft Corporation (1987) **"The Windows Interface: An application design guide"**. Microsoft Corporation.

[Nielsen, 1999] Nielsen J. **"Designing Web Usability: The Practice of Simplicity"**. New Riders, 1999.

[NIST, 2001] NIST, **"Web Metrics Testbed"**, 2001. http://zing.ncsl.nist.gov/WebTools/

[Østebye, 1999] Kasper Østebye, Norwegian Computing Center, Norway, **"Hierarchical structure through navigation side bars"**, Proceedings of the HT99 Workshop on Hypermedia Development. http://www.eng.uts.edu.au/~dbl/HypDev/ht99w/submissions/OesterbyeHT99Workshop.pdf

[Perzel & Kane, 1999] Kimberly Perzel, David Kane, SRA International, **"Usability Patterns for Applications on the World Wide Web"**, PloP '99 Conference. http://jerry.cs.uiuc.edu/~plop/plop99/proceedings/Kane/perzel_kane.pdf

[Portland, 2001] **Portland Form**, 2001 http://c2.com/cgi/wiki?PortlandForm

[Rising, 1998] **"The patterns handbook"**, collected and introduced by Linda Rising, Cambridge University, 1998, pp.237.

[Schuetze et al., 1997] Schuetze, M., J. P. Riegel, and G. Zimmermann, **"A Pattern-Based Application Generator for Building Simulation"**, in Software Engineering Notes, ESEC/FSE, Vol. 22, No. 6 Nov 1997, pp.468-482.

[Sefika et al., 1996] Sefika, M., A. Sane, R. Campbell, **"Monitoring Compliance of a Software System with its high-Level Design Models"**, Proc. of ICSE'96, 1996.

[Tidwell, 1999] Tidwell, Jenifer, **" Common Ground: A Pattern Language for Human-Computer Interface Design "**, 1999.
http://www.mit.edu/~jtidwell/common_ground.html

[TO U., 1990] The Open University in association with the Department of Trade and Industry (1990) **"Usability Now! A Guide to Usability"**, The Open University, ISBN 0749243449.

[UCB, 2001] **"DENIM and SILK"**, Group for User Interface Research, University of California at Berkeley, 2001 July.
http://guir.berkeley.edu/projects/denim/

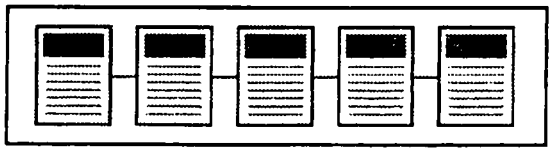
[Yacoub et al., 2000] Sherif Yacoub, Hengyi Xue, and Hany Ammar , **"Automating the Development of Pattern-Oriented Designs"**, in Proc. of Application Specific Software Engineering Technology ASSET'2000, IEEE Computer Society, Dallas, Texas, March 2000
http://www.csee.wvu.edu/~ammar/

[Yacoub, 1999] Sherif M. Yacoub, West Virginia University , **"Pattern-Oriented Analysis and Design (POAD)"**, 1999.
http://www.csee.wvu.edu/~yacoub/phd.htm

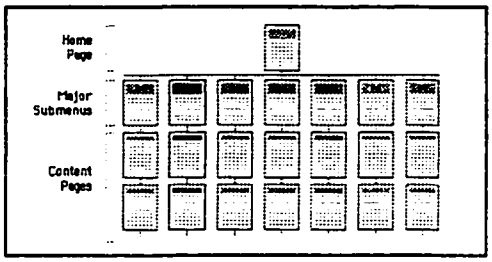
[Yacoub & Ammar, 1999] Sherif M. Yacoub and Hany H. Ammar ,West Virginia University, **"Tool Support for Developing Pattern-Oriented Architectures"**, In Proceedings of the 1st Symposium on Reusable Architectures and Components for Developing Distributed Information Systems (RACDIS'99), Orlando, Florida, August 2-3, 1999, Vol I, pp. 665-670.
http://www.csee.wvu.edu/~ammar/

Appendix A: UPADE Web Patterns

A.1 Sequential Pattern

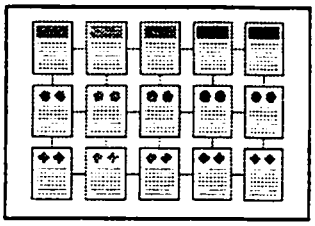
<table>
<tr><td rowspan="2">Identification</td><td>Name</td><td colspan="2">SEQUENTIAL</td></tr>
<tr><td>Category</td><td colspan="2">Product-Oriented Patterns→ Architectural Patterns → Information Architecture Patterns</td></tr>
<tr><td rowspan="3">Context</td><td>User</td><td colspan="2">Novice and expert</td></tr>
<tr><td>Task</td><td colspan="2">Task is composed of consecutive steps. There is a narrative time line or logical order among the contents.</td></tr>
<tr><td>Workplace</td><td colspan="2">A small Web site or sub-branch of a comprehensive Web application.</td></tr>
<tr><td>Problems</td><td colspan="3">- The user can easily follow a narrative, time line or other logical order to find all the relevant information.
- It is the simplest method to organize basic content and training materials.</td></tr>
<tr><td>Forces</td><td colspan="3">- Effectiveness
- Efficiency
- Understandability
- Consistency
- User Guidance (legibility)</td></tr>
<tr><td>Examples</td><td colspan="3"> 
Broadly applied into the organization of Index, tutorial, encyclopedias and glossaries.</td></tr>
<tr><td>Rationale</td><td colspan="3">- Causality
- Population stereotypes</td></tr>
<tr><td>Solution</td><td colspan="3">Organize the pages in a sequence, where the contents of the pages are presented in a linear narrative.</td></tr>
<tr><td rowspan="4">Patterns</td><td>Super-ordinate</td><td colspan="2">Composite</td></tr>
<tr><td>Sub-ordinate</td><td colspan="2">Focus Page, Utility Page, Navigation Page, Tiled Page, Stack Page</td></tr>
<tr><td>Neighboring</td><td colspan="2">Hierarchical, Grid</td></tr>
<tr><td>Competitor</td><td colspan="2">Hierarchical, Grid</td></tr>
<tr><td>References</td><td colspan="3">- "Sequence" guideline of Yale guidelines.
- "Step by step instructions" pattern in Common Ground.</td></tr>
</table>

A.2 Hierarchical Pattern

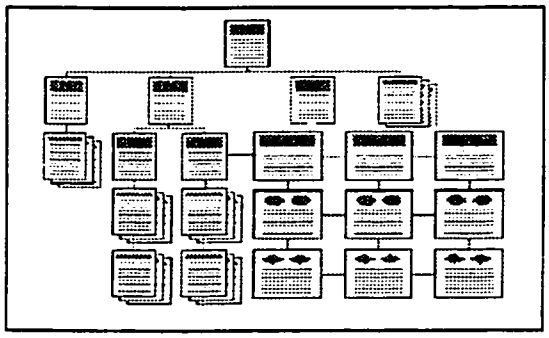
<table>
<tr><td rowspan="2">Identification</td><td>Name</td><td colspan="2">HIERARCHICAL</td></tr>
<tr><td>Category</td><td colspan="2">Product-Oriented Patterns→ Architectural Patterns→ Information Architecture Patterns</td></tr>
<tr><td rowspan="3">Context</td><td>User</td><td colspan="2">Novice and expert</td></tr>
<tr><td>Task</td><td colspan="2">Tasks are structured into a hierarchy. All the sub-tasks stem from one original center.</td></tr>
<tr><td>Workplace</td><td colspan="2">Web applications</td></tr>
<tr><td>Problems</td><td colspan="3">- The user can easily go through from the most general overview of the Web site, such as home page, down to the most specific or optional topics.
- Have more flexibility than sequence structure.</td></tr>
<tr><td>Forces</td><td colspan="3">- Efficiency
- Effectiveness
- Satisfaction
- Understandability
- Completeness
- Flexibility</td></tr>
<tr><td>Examples</td><td colspan="3"> 

Most of the current Web sites are utilizing the hierarchical structure.</td></tr>
<tr><td>Rationale</td><td colspan="3">- Causality
- Population stereotypes</td></tr>
<tr><td>Solution</td><td colspan="3">- All the pages are organized in a hierarchical cascade model. The sub-branches expend from one generic center. There is no intersection among sub-branches.
- Certain constraints should be applied on the width, depth of the structure.</td></tr>
<tr><td rowspan="4">Patterns</td><td>Super-ordinate</td><td colspan="2">Composite</td></tr>
<tr><td>Sub-ordinate</td><td colspan="2">Focus Page, Utility Page, Navigation Page, Tiled Page, Stack Page</td></tr>
<tr><td>Neighboring</td><td colspan="2">Sequential, Grid</td></tr>
<tr><td>Competitor</td><td colspan="2">Sequential, Grid</td></tr>
<tr><td>References</td><td colspan="3">- "Hierarchy" guideline in Yale guidelines.
- "Hierarchical Set" pattern in Common Ground.</td></tr>
</table>

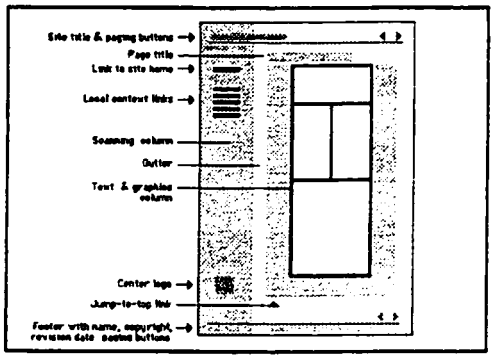
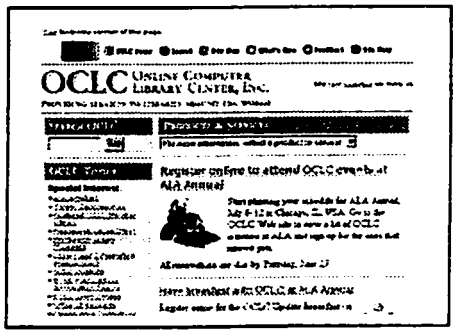
A.3 Grid Pattern

<table>
<tr><td rowspan="2">Identification</td><td>Name</td><td colspan="2">GRID</td></tr>
<tr><td>Category</td><td colspan="2">Product-Oriented Patterns→ Architectural Patterns→ Information Architecture Patterns</td></tr>
<tr><td rowspan="3">Context</td><td>User</td><td colspan="2">Expert</td></tr>
<tr><td>Task</td><td colspan="2">Topics and contents are fairly correlated with each other. Moreover, there is no particular hierarchy of importance.</td></tr>
<tr><td>Workplace</td><td colspan="2">Web applications</td></tr>
<tr><td>Problems</td><td colspan="3">- The user can find all the related information organized into multiple dimensions.
- It can benefit the experienced user who already has some basic knowledge on the topic and interrelationship.
- More complicate than sequence structure.</td></tr>
<tr><td>Forces</td><td colspan="3">- Productivity
- Effectiveness
- Completeness
- Accuracy</td></tr>
<tr><td>Examples</td><td colspan="3"> 
Applications: Procedural manuals, lists of university courses, medical case descriptions</td></tr>
<tr><td>Rationale</td><td colspan="3">- Causality</td></tr>
<tr><td>Solution</td><td colspan="3">- Establish a coordinate system to organize the contents, such as a time line versus several categories of the technology evolution.
- Make each unit to share a highly uniform structure of topics and sub-topics.</td></tr>
<tr><td rowspan="4">Patterns</td><td>Super-ordinate</td><td colspan="2">Composite</td></tr>
<tr><td>Sub-ordinate</td><td colspan="2">Focus Page, Utility Page, Navigation Page, Tiled Page, Stack Page</td></tr>
<tr><td>Neighboring</td><td colspan="2">Sequential, Hierarchical</td></tr>
<tr><td>Competitor</td><td colspan="2">Sequential, Hierarchical</td></tr>
<tr><td>References</td><td colspan="3">- "Grid" guideline in Yale guidelines.</td></tr>
</table>

A.4 Composite Pattern

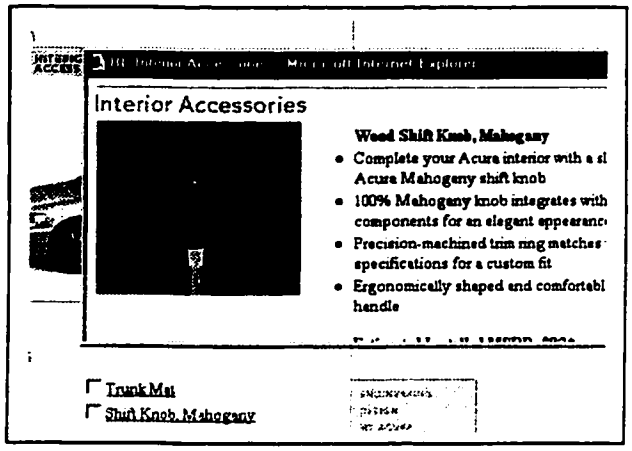
<table>
<tr><td rowspan="2">Identification</td><td>Name</td><td colspan="2">COMPOSITE</td></tr>
<tr><td>Category</td><td colspan="2">Product-Oriented Patterns→ Architectural Patterns→ Information Architecture Patterns</td></tr>
<tr><td rowspan="3">Context</td><td>User</td><td colspan="2">Novice and expert</td></tr>
<tr><td>Task</td><td colspan="2">Tasks and topics may have very different structures and usage.</td></tr>
<tr><td>Workplace</td><td colspan="2">Web applications</td></tr>
<tr><td>Problems</td><td colspan="3">- Heterogeneous information can be well and completely organized.
- It is easy and understandable for user to surf at a complex Web site where topics and contents may have various structures.</td></tr>
<tr><td>Forces</td><td colspan="3">- Effectiveness
- Efficiency
- Flexibility
- Understandability
- Completeness</td></tr>
<tr><td>Examples</td><td colspan="3"> 
Most comprehensive Web sites apply such a structure.</td></tr>
<tr><td>Rational</td><td colspan="3">- Causality
- Affordance</td></tr>
<tr><td>Solution</td><td colspan="3">Apply appropriate structure model according to the interrelationship of the contents. It can combine sequential, hierarchical and grid architectural patterns.</td></tr>
<tr><td rowspan="4">Patterns</td><td>Super-ordinate</td><td></td></tr>
<tr><td>Sub-ordinate</td><td>Sequential, Hierarchical, Grid</td></tr>
<tr><td>Neighboring</td><td>Sequential, Hierarchical, Grid</td></tr>
<tr><td>Competitor</td><td>Sequential, Hierarchical, Grid</td></tr>
<tr><td>References</td><td colspan="3"></td></tr>
</table>

A.5 Focus Page Pattern

Identification	**Name**	**FOCUS PAGE**
	Category	Product-Oriented Patterns→ Structural Patterns→ Page Managers Patterns
Context	User	Novice and expert
	Task	The Web page is the fountainhead and center of a Web site. It must balance aesthetics and practicality to attract the user, especially the novice, at first glance.
	Workplace	Web page
Problems	- The novice shows interests on the Web site and may be willing to continue exploring. - The expert user can find the useful information easily and reach the target topics promptly.	
Forces	- Efficiency - Effectiveness - Understandability - Attractiveness	
Examples	 	
Rational	- Affordance - Causality - Transfer effects - Population Stereotype	
Solution	- Deploy a utility toolbar, an index of major topics, a shortcut menu, push information sections and maintenance support information section on the Web page. - Apply dimension constraints on the width and length of a page. - Avoid frames, big size images, irrelevant applets, blinking texts and banners.	

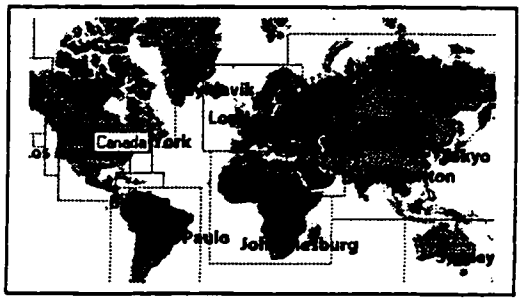
Patterns	Super-ordinate	Sequential, Hierarchical, Grid, Composite
	Sub-ordinate	Executive Summary, On Fly Description, Shortcut, Convenient Toolbar, Browsing Index
	Neighboring	Tiled Page, Stack Page
	Competitor	Tiled Page, Stack Page
References	- "Navigable space" and "Central Working Surface" patterns in Common Ground. - "News" in HPR [HPR, 2001] - "What they see is all they get" defined by Kimberly Perzel and David Kane [Perzel & Kane, 1999] - "Balanced pages", "Design grids for pages", "Graphics safe areas", "Consistency" and "Page length" in Yale Guidelines.	

A.6 Utility Page Pattern

<table>
<tr><td rowspan="2">Identification</td><td>Name</td><td colspan="2">UITILITY PAGE</td></tr>
<tr><td>Category</td><td colspan="2">Product-Oriented Patterns→ Structural Patterns→ Page Managers Patterns</td></tr>
<tr><td rowspan="3">Context</td><td>User</td><td colspan="2">Novice and Expert</td></tr>
<tr><td>Task</td><td colspan="2">Provide extra information or assistance to the user without interrupting the workflow.</td></tr>
<tr><td>Workplace</td><td colspan="2">Web applications</td></tr>
<tr><td>Problems</td><td colspan="3">- The user can organize his favorite Web links or items in the utility page.
- The user can acquire all the commonly helpful information in this page.
- It can enhance the understandability of the user on the target object through an extra explanation window.</td></tr>
<tr><td>Forces</td><td colspan="3">- Efficiency
- Productivity
- User Guidance
- Helpfulness
- Required Resources
- Consistency</td></tr>
<tr><td>Examples</td><td colspan="3"> 

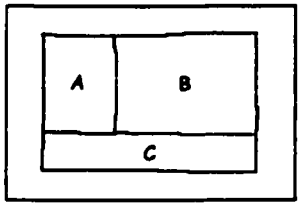
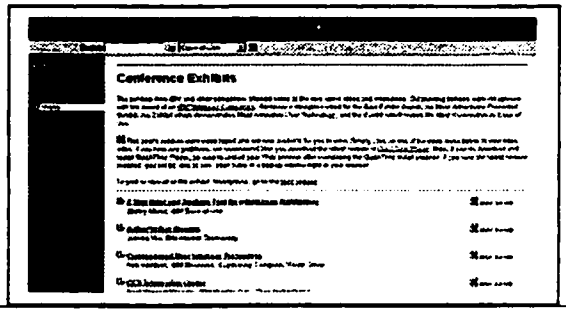
Applications:
- Bookmarks;
- Pop-up info windows;
- FAQ pages.</td></tr>
<tr><td>Rationale</td><td colspan="3">- Affordance
- Mapping</td></tr>
<tr><td>Solution</td><td colspan="3">- Group all the frequently asked questions in one page; make an index at the top of the page.
- Make the additional explanation consistent to the selected items.</td></tr>
<tr><td rowspan="4">Patterns</td><td>Super-ordinate</td><td colspan="2">Sequential, Hierarchical, Grid, Composite</td></tr>
<tr><td>Sub-ordinate</td><td colspan="2"></td></tr>
<tr><td>Neighboring</td><td colspan="2">Focus Page, Navigation Page, Tiled Page, Stack Page</td></tr>
<tr><td>Competitor</td><td colspan="2"></td></tr>
<tr><td>References</td><td colspan="3">- "Helper Posture" and "Bookmarks" patterns in Common Ground.</td></tr>
</table>

A.7 Navigation Page Pattern

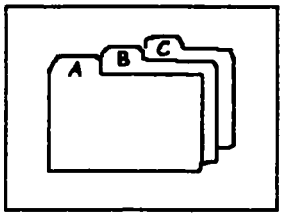
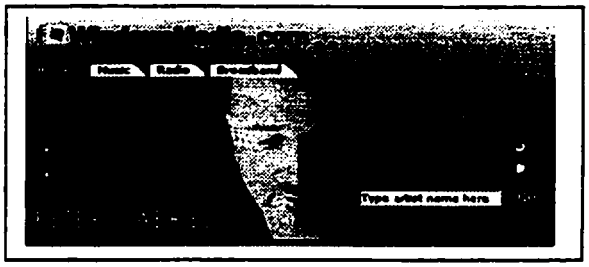
<table>
<tr><td rowspan="2">Identification</td><td>Name</td><td>NAVIGATION PAGE</td></tr>
<tr><td>Category</td><td>Product-Oriented Patterns→ Structural Patterns→ Page Managers Patterns</td></tr>
<tr><td rowspan="3">Context</td><td>User</td><td>Novice and expert</td></tr>
<tr><td>Task</td><td>- All the information is grouped and condensed into a page.
- Each item in this page directs the user to reach the appropriate topic and acquire further detailed information.</td></tr>
<tr><td>Workplace</td><td>Web page</td></tr>
<tr><td>Problems</td><td colspan="2">- The user can have an overview on the whole allocation of the information.
- The user can find and reach the target topic promptly.</td></tr>
<tr><td>Forces</td><td colspan="2">- Efficiency
- Satisfaction
- Understandability
- Consistency
- Minimal Action
- User Guidance</td></tr>
<tr><td>Examples</td><td colspan="2"> 
Applications:
- World weather forecast;
- Geographical Maps.</td></tr>
<tr><td>Rationale</td><td colspan="2">- Affordance
- Mapping
- Transfer effects
- Popular stereotype</td></tr>
<tr><td>Solution</td><td colspan="2">- Group the relevant topics.
- Utilize meaningful pictures or metaphors to hint the user the proper area
- Provide labels, on fly description and other extra information on the target object to allow the user to know what it is.</td></tr>
<tr><td rowspan="4">Patterns</td><td>Super-ordinate</td><td>Sequential, Hierarchical, Grid, Composite</td></tr>
<tr><td>Sub-ordinate</td><td>Map, Form, On Fly Description, Executive Summary</td></tr>
<tr><td>Neighboring</td><td>Focus Page, Navigation Page, Tiled Page, Stack Page</td></tr>
<tr><td>Competitor</td><td></td></tr>
</table>

References	- "Navigable space" described in Common Ground. - "Image Maps" of Yale guidelines.

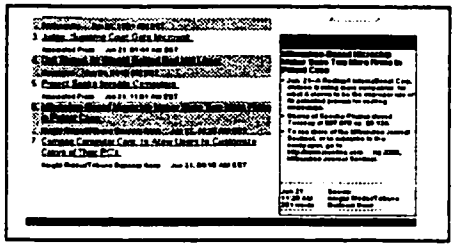
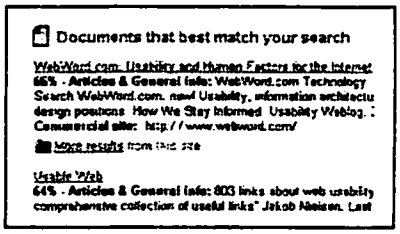
A.8 Tiled Page Pattern

Identification	**Name**	**TILED PAGE**
	Category	Product-Oriented Patterns→ Structural Patterns→ Page Managers Patterns
Context	User	Expert and Novice
	Task	The contents are well structured and presented to the user from general to specific level.
	Workplace	Web page
Problems	The user can easily look through the introductions of the topics and find the relevant detailed information on the same page.	
Forces	- Efficiency - Productivity - Understandability - Completeness	
Examples	 	
Rationale	- Mapping - Causality	
Solution	- Divide the page into several surfaces. - Pane "A" is the menu list or catalogues, pane "B" is the brief introduction for the selected catalogue and pane "C" is detailed information of "B" part.	
Patterns	Super-ordinate	Sequential, Hierarchical, Grid, Composite
	Sub-ordinate	Executive Summary, On Fly Description, Shortcut, Convenient Toolbar, Browsing Index, Path, Tutorial Toolbar
	Neighboring	Focus Page, Stack Page
	Competitor	Focus Page, Stack Page
References	- "Tiled Working Surfaces", "Overview beside detail" in Common Ground - "Garden of windows" in Experience - "Container navigation" in Amsterdam collection	

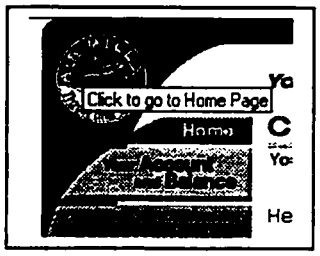
A.9 Stack Page Pattern

<table>
<tr><td rowspan="2">Identification</td><td>Name</td><td colspan="2">STACK PAGE</td></tr>
<tr><td>Category</td><td colspan="2">Product-Oriented Patterns→ Structural Patterns→ Page Managers Patterns</td></tr>
<tr><td rowspan="3">Context</td><td>User</td><td colspan="2">Novice and Expert</td></tr>
<tr><td>Task</td><td colspan="2">- Contents are grouped into categories.
- No obvious hierarchical structure exists amongst those categories or topics.</td></tr>
<tr><td>Workplace</td><td colspan="2">Web page</td></tr>
<tr><td>Problems</td><td colspan="3">- All the information is clearly grouped.
- The user can easily navigate the information through switching the tabs of the working spaces.
- The user can only access one category each time.</td></tr>
<tr><td>Forces</td><td colspan="3">- Effectiveness
- Understandability</td></tr>
<tr><td>Examples</td><td colspan="3"> 
  </td></tr>
<tr><td>Rationale</td><td colspan="3">- Constraints
- Affordance
- Mapping</td></tr>
<tr><td>Solution</td><td colspan="3">- Utilize several surfaces stacked together to group information and label them by the name of the categories.
- Locate the navigation area at the top if the number of surfaces is less than 8; otherwise, put the navigation area on the left side.</td></tr>
<tr><td rowspan="4">Patterns</td><td>Super-ordinate</td><td colspan="2">Sequential, Hierarchical, Grid, Composite</td></tr>
<tr><td>Sub-ordinate</td><td colspan="2">Executive Summary, On Fly Description, Browsing Index</td></tr>
<tr><td>Neighboring</td><td colspan="2">Focus Page, Tiled Page</td></tr>
<tr><td>Competitor</td><td colspan="2">Focus Page, Tiled Page</td></tr>
<tr><td>References</td><td colspan="3">- "Stack of Working surfaces" in Common Ground.
- "Navigating spaces" in Amsterdam Collection
- "Garden of Windows" in Experience.</td></tr>
</table>

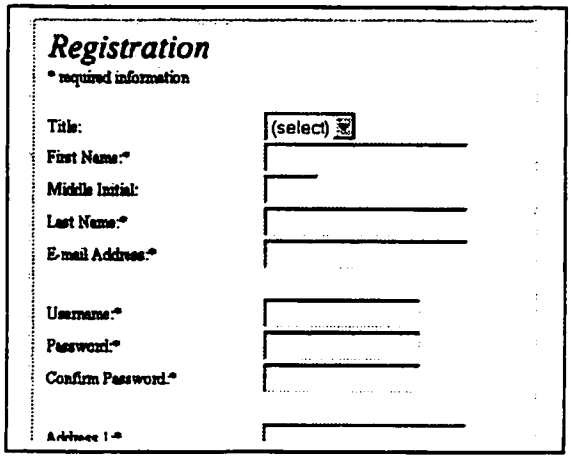
A.10 Executive Summary Pattern

Identification	**Name**	**EXECUTIVE SUMMARY**
	Category	Product-Oriented Patterns→ Structural Patterns→ Information Containers Patterns
Context	User	Novice
	Task	Provide preview of the underlying information for the user.
	Workplace	Web page
Problems	- Help the user determine if the topic has the relevant information. - Avoid the user spending time downloading and reading large amounts of information on irrelevant topics.	
Forces	- Efficiency - Effectiveness - Safety - User Guidance - Helpfulness	
Examples	 	
Rationale	- Mapping - Causality	
Solution	- Summarize the contents by using concise sentences beneath or beside the underlying topics.	
Patterns	Super-ordinate	Focus Page, Tiled Page, Stack Page, Utility Page, Navigation Page
	Sub-ordinate	Browsing Index
	Neighboring	On Fly Description, Form, Bullet
	Competitor	On Fly Description
References	- "Preview" in Amsterdam collection	

A.11 On Fly Description Pattern

Identification	**Name**	**ON FLY DESCRIPTION**
	Category	Product-Oriented Patterns→ Structural Patterns→ Information Containers Patterns
Context	User	Novice
	Task	Provide the user a short description of the objects that the mouse is focused on.
	Workplace	Web applications
Problems	- The user can get additional information on the target object. - The user may feel confident to acquire extra confirmation before next possible actions.	
Forces	- Satisfaction - Safety - Effectiveness - Helpfulness - User Guidance - Accuracy	
Examples	 	
Rationale	- Mapping - Causality - Transfer effects	
Solution	- Give an accurate and short phrase or sentence in close spatial or temporary proximity to the target object.	
Patterns	Super-ordinate	Focus Page, Tiled Page, Stack Page, Utility Page, Navigation Page
	Sub-ordinate	
	Neighboring	Executive Summary, Form, Bullet
	Competitor	Executive Summary
References	- "Short description" in Common Ground - "Hinting" in Amsterdam - "Behavior Anticipation" in HPR.	

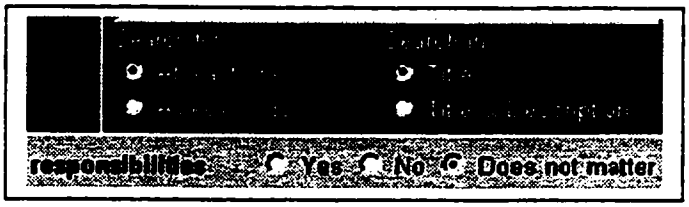
A.12 Form Pattern

<table>
<tr><td rowspan="2">Identification</td><td>Name</td><td colspan="2">FORM</td></tr>
<tr><td>Category</td><td colspan="2">Product-Oriented Patterns→ Structural Patterns→ Information Containers Patterns</td></tr>
<tr><td rowspan="3">Context</td><td>User</td><td colspan="2">Novice and Expert</td></tr>
<tr><td>Task</td><td colspan="2">Collect information from the user</td></tr>
<tr><td>Workplace</td><td colspan="2">Web applications</td></tr>
<tr><td>Problems</td><td colspan="3">- Allow the user to input complete information in the format of characters or the Arabic numerals.
- The user knows clearly how to use the form.
- The user can check all the input information before submission.</td></tr>
<tr><td>Forces</td><td colspan="3">- Effectiveness
- Understandability</td></tr>
<tr><td>Examples</td><td colspan="3"> 

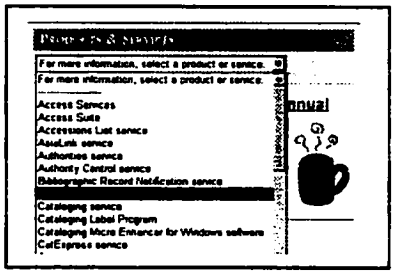
Applications:
- Survey;
- On line registration.</td></tr>
<tr><td>Rationale</td><td colspan="3">- Transfer effects
- Affordance
- Visual Constraints</td></tr>
<tr><td>Solution</td><td colspan="3">- Provide some list-boxes to reduce the user's work.
- Apply constraints on the fields or provide certain format checking mechanisms for the information correction.
- Allow the user to clear or cancel his action</td></tr>
<tr><td rowspan="4">Patterns</td><td>Super-ordinate</td><td colspan="2">Utility Page, Tiled Page, Stack Page</td></tr>
<tr><td>Sub-ordinate</td><td colspan="2">Bullet</td></tr>
<tr><td>Neighboring</td><td colspan="2">Executive Summary, On Fly Description, Bullet</td></tr>
<tr><td>Competitor</td><td colspan="2"></td></tr>
</table>

References	- "Form", "Forgiving text entry", "Structured text entry", "Good defaults", "Remembered state" in Common ground; - "Show the format required " in Brighton; - "Focus!" in Amsterdam; - "Required field marker" and "What they see is all they get" defined by Kimberly Perzel and David Kane.

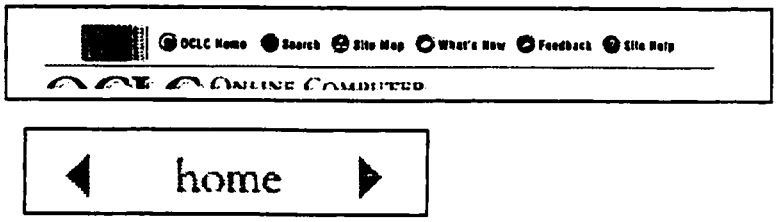
A.13 Bullet Pattern

<table>
<tr><td rowspan="2">Identification</td><td>Name</td><td colspan="2">BULLET</td></tr>
<tr><td>Category</td><td colspan="2">Product-Oriented Patterns→ Structural Patterns→ Information Containers Patterns</td></tr>
<tr><td rowspan="3">Context</td><td>User</td><td colspan="2">Novice, Expert</td></tr>
<tr><td>Task</td><td colspan="2">Collect small amount or simple information from the user</td></tr>
<tr><td>Workplace</td><td colspan="2">Web applications</td></tr>
<tr><td>Problems</td><td colspan="3">- The user may feel easily to do the action.
- The user only needs to click on the mouse to select the items.
- It is clear and easy for the user to utilize the bullets.</td></tr>
<tr><td>Forces</td><td colspan="3">- Efficiency
- Minimal Action
- Understandability</td></tr>
<tr><td>Examples</td><td colspan="3"> 
Applications: Survey, Searching engine, on-line shopping</td></tr>
<tr><td>Rationale</td><td colspan="3">- Mapping
- Visual Constraints
- Accuracy</td></tr>
<tr><td>Solution</td><td colspan="3">- Group the information according to the contents or categories
- Label the contents by accuracy phrases or sentence</td></tr>
<tr><td rowspan="4">Patterns</td><td>Super-ordinate</td><td colspan="2">Focus Page, Utility Page, Navigation Page, Tiled Page, Stack Page</td></tr>
<tr><td>Sub-ordinate</td><td colspan="2"></td></tr>
<tr><td>Neighboring</td><td colspan="2">Executive Summary, On Fly Description, Form</td></tr>
<tr><td>Competitor</td><td colspan="2">Form</td></tr>
<tr><td>References</td><td colspan="3">- "Small groups of related things" and "choice from a small set" in Common Ground.</td></tr>
</table>

A.14 Shortcut Pattern

Identification	**Name**	**SHORTCUT**
	Category	Product-Oriented Patterns→ Navigation Support Patterns
Context	User	Expert
	Task	Help the user reach his favorite or frequently visited pages
	Workplace	Web page, mainly used on Homepage
Problems	- The user can reach the target pages directly - The user can avoid extra time on looking for the relevant pages.	
Forces	- Efficiency - Productivity - Minimal Action - Minimal Memory Load - User Guidance	
Examples		
Rationale	- Causality - Constraints - Transfer effects	
Solution	- Locate on the homepage - Provide the collection of all the frequently visited pages links in a list box	
Patterns	Super-ordinate	Focus Page, Tiled Page, Stack Page, Navigation Page, Utility Page
	Sub-ordinate	
	Neighboring	Convenient Toolbar, Path, Map, Browsing Index
	Competitor	Convenient Toolbar
References	- "Choice from a small set", "Choice from a large set" in Common Ground.	

A.15 Convenient Toolbar Pattern

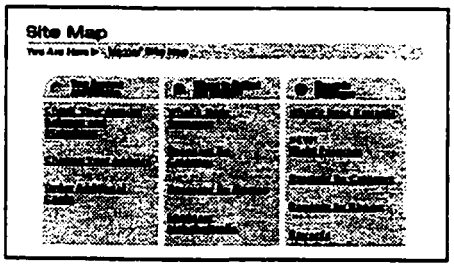
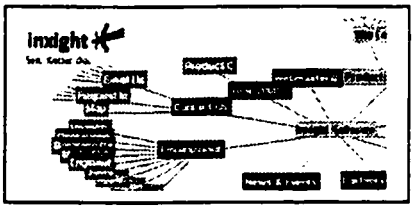
Identification	**Name**	**CONVENIENT TOOLBAR**
	Category	Product-Oriented Patterns→ Navigation Support Patterns
Context	User	Expert
	Task	Assist the user to reach the most useful and frequently visited pages at any time throughout the Web site.
	Workplace	Web applications
Problems	- The user can easily find the most commonly used pages regardless of the current state of the artifact. - The user can reach these convenient pages promptly.	
Forces	- Efficiency - Safety - Consistency - Minimal Action - Minimal Memory - User Guidance - Helpfulness	
Examples		
Rationale	- Affordance - Mapping - Causality	
Solution	- Group the most convenient action links, such as home, site map, help and etc. - Utilize meaningful metaphors and accurate phases as labels. - Locate it at the consistent place throughout the Web site.	
Patterns	Super-ordinate	Focus Page, Tiled Page, Stack Page, Navigation Page, Utility Page
	Sub-ordinate	
	Neighboring	Shortcut, Path, Map, Browsing Index
	Competitor	Shortcut

References	- "Convenient environment actions", "Go back to a safe place" in Common Ground. - "Goal oriented areas" in Experience. - "List Browser" in Amsterdam - "Guided tour" in HPR. - "Basic interface design" and "Links & navigation" in Yale guidelines.

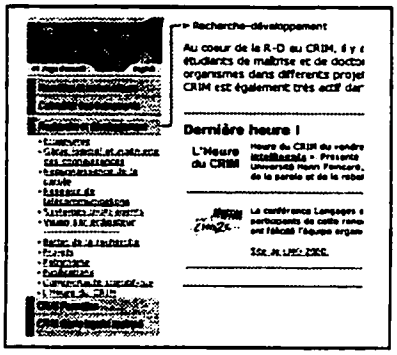
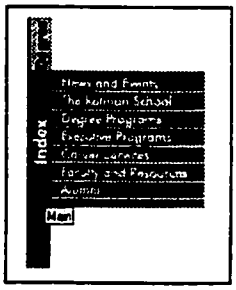
A.16 Path Pattern

Identification	**Name**	**PATH**
	Category	Product-Oriented Patterns→ Navigation Support Patterns
Context	User	Novice and Expert
	Task	Indicate the user his current location
	Workplace	Web applications
Problems	- The user is pleased to acquire the whole path explicitly, which makes the user feel more confident and controllable on the Web site. - The user can conveniently navigate between pages by using the path. - The user does not need extra memory for the pages' allocation.	
Forces	- Effectiveness - Safety - Satisfaction - Consistency - Minimal Memory load - User Guidance - Controllability	
Examples	useit.com → Alertbox → June 2000 Customers as Designers	
Rationale	- Mapping - Causality	
Solution	- Utilize accurate words as labels and meaningful icons as directions to indicate each stage of the process since the user accessed the Web site. - Locate it consistently at the top of the Web pages	
Patterns	Super-ordinate	Focus Page, Tiled Page, Stack Page, Navigation Page, Utility Page
	Sub-ordinate	
	Neighboring	Shortcut, Convenient Toolbar, Map, Browsing Index
	Competitor	
References		

A.17 Map Pattern

Identification	**Name**	**MAP**
	Category	Product-Oriented Patterns→ Navigation Support Patterns
Context	User	Novice and Expert
	Task	Collect complete links of all the Web pages.
	Workplace	Web applications
Problems	- The user can easily acquire the skeleton of the whole Web site and the collection of the complete page links. - The user may reach any page directly. - The user can search the target page manually.	
Forces	- Efficiency - Flexibility - Completeness - User Guidance	
Examples	 	
Rationale	- Mapping - Causality - Population stereotype - Transfer Effects	
Solution	- Group the information according to the alphabetical sequence or categories. - Illustrate the index or topic list using a stable form or a dynamic tree format	
Patterns	Super-ordinate	Tiled Page, Stack Page, Navigation Page
	Sub-ordinate	
	Neighboring	Shortcut, Convenient Toolbar, Path, Browsing Index
	Competitor	
References	- "Map of navigable spaces", "High-density information display" and "Tabular set" in Common ground; - "Hyper-Map" pattern defined by D.M. Germán and D.D. Cowan	

A.18 Browsing Index Pattern

Identification	**Name**	**BROWSING INDEX**
	Category	Product-Oriented Patterns→ Navigation Support Patterns
Context	User	Novice
	Task	Present visible and structured items and sub-items to the user and allow the user to switch directly from one item to another.
	Workplace	Web applications
Problems	- The user can easily and promptly navigate amongst the items from the menu. - The user is able to know the interrelationship of items.	
Forces	- Efficiency - Effectiveness - User Guidance	
Examples	 	
Rationale	- Affordance - Mapping - Causality	
Solution	- Provide dynamic list of the items due to constraints of the space. - The order of the index might be based on a ranking and the ordering criterion must be visible.	
Patterns	Super-ordinate	Focus Page, Tiled Page, Stack Page, Utility Page, Navigation Page
	Sub-ordinate	Executive Summary, On Fly Description
	Neighboring	Shortcut, Convenient Toolbar, Path, Map
	Competitor	
References	- "Index navigation" pattern in HPR	