

Copyright is owned by the Author of the thesis. Permission is given for a copy to be downloaded by an individual for the purpose of research and private study only. The thesis may not be reproduced elsewhere without the permission of the Author.

Managing User Interface Pattern Collections

A thesis presented in partial fulfilment of the requirements
for the degree of
Master of Science in Computer Science
at Massey University, Palmerston North, New Zealand.

Junhua Deng

2006

Abstract

The research presented in this thesis describes the development of a comprehensive UI pattern management tool, MUIP, to support researchers and UI designers manipulate and explore a repository of UI pattern collections.

The concept of patterns originated from Alexander's pattern language for the architecture domain. Later, the software development and HCI communities adopted the pattern concept. Many disparate UI pattern collections have been developed and published using various media, such as books, internet, etc. Various pattern formats were used in these collections. In 2003, to cope with this problem, a group of HCI researchers developed a standardised pattern form, called PLML. Researchers have authored patterns, investigated the characteristics of pattern collections and also identified many of the functions required to manage pattern collections.

A framework for MUIP has been developed in the light of the analysis of the relevant literature and a survey of existing pattern tools. The framework supports the following features: pattern authoring, manipulating forces, browsing patterns, searching patterns, versioning and customising patterns, relating patterns, manipulating collections and importing or exporting patterns. Patterns are described using the standard pattern form (PLML). An enhanced version of PLML, called PLML v1.2, has been developed so that pattern contents can be organised more effectively.

Based on this framework, a specification of a comprehensive pattern management system for manipulating pattern collections was developed and a prototype implemented accordingly. A formal evaluation confirmed the usefulness of the prototype.

Acknowledgements

First of all, I want to thank my supervisor, Associate Professor Elizabeth Kemp. She introduced me to the research area of UI patterns and tools. I am especially thankful to her conscientious and patience in the research and for her helpful insights and guidance.

Thanks also go to my second supervisor, Ms Lis Todd for her co-supervision, and for her helpful suggestions in regard to the research.

In addition, I would like to thank Mr. Robert Thompson for proofreading and suggesting corrections to the thesis.

Finally, I would give many thanks to my parents, my wife Yu Liu, my sisters and my brother for all their support and love during these years.

Related Publication

Deng, J., Kemp, E., & Todd, E. G. (2005). Managing UI pattern collections. In *Proceedings of the 6th ACM SIGCHI New Zealand Chapter's International Conference on Computer-Human Interaction: Making CHI Natural* (Auckland, New Zealand, July 07 - 08, 2005). CHINZ '05, vol. 94. ACM Press, New York, NY, 31-38.

List of Contents

CHAPTER 1	INTRODUCTION	1
1.1	BACKGROUND	1
1.2	MOTIVATION AND OBJECTIVES	5
1.3	RESEARCH APPROACH	7
1.4	OUTLINE OF THESIS	8
CHAPTER 2	LITERATURE REVIEW	9
2.1	ALEXANDER'S PATTERN CONCEPT	9
2.2	PATTERNS IN SOFTWARE ENGINEERING	11
2.3	PATTERNS IN HUMAN COMPUTER INTERACTION (HCI)	13
2.3.1	<i>Patterns and Guidelines</i>	14
2.3.2	<i>Pattern Collections</i>	17
2.3.3	<i>Types of Relationships</i>	17
2.4	PATTERN FORMS	20
2.5	FORMAL PATTERN FORMS FOR HCI PATTERNS	21
2.5.1	<i>XML</i>	22
2.5.2	<i>Interface Markup Languages</i>	22
2.5.3	<i>Pattern Language Markup Language (PLML)</i>	23
2.6	FORCES IN THE PATTERN FORM	26
2.7	CATEGORISATION OF PATTERNS	27
2.8	SUMMARY	34
CHAPTER 3	EXISTING UI PATTERN MANAGEMENT SPECIFICATIONS AND TOOLS	35
3.1	INTRODUCTION	35
3.2	EXISTING UI PATTERN MANAGEMENT SPECIFICATIONS AND TOOLS	35
3.3	EXISTING HCI PATTERN TOOLS	38
3.3.1	<i>Types of Pattern Tools</i>	39
3.3.2	<i>Cataloging Tools</i>	40
3.3.3	<i>Management Tools</i>	43
3.3.4	<i>UI Design Assistance Tools</i>	49
3.3.5	<i>Summary of Existing Tools</i>	52
3.4	CONCEPTUALISATION OF A PATTERN MANAGEMENT TOOL	53
3.4.1	<i>The Support of a Common Pattern Form - PLML</i>	53
3.4.2	<i>Needs of Users</i>	54
3.4.3	<i>Proposed Key Features of the Tool with Analysis of Existing Tools</i>	54
3.4.4	<i>Conceptualisation of a Pattern Management Tool</i>	60
3.5	SUMMARY	62
CHAPTER 4	PROTOTYPE FOR MUIP	63
4.1	FEATURE SPECIFICATION	63

4.1.1	<i>Analysis</i>	64
4.1.2	<i>PLML v1.2</i>	67
4.2	DESIGN ISSUES	69
4.2.1	<i>System Architecture</i>	69
4.2.2	<i>Database Design</i>	71
4.2.3	<i>Development Tools</i>	74
4.2.4	<i>Prototyping</i>	76
4.3	REVIEW OF MUIP	87
4.4	SUMMARY	88
CHAPTER 5 EVALUATION		90
5.1	BACKGROUND	90
5.1.1	<i>Techniques</i>	92
5.1.2	<i>Triangulation</i>	97
5.2	METHODOLOGY	97
5.2.1	<i>Determining Goals</i>	98
5.2.2	<i>Exploring Questions for the Goals</i>	98
5.2.3	<i>Choosing Evaluation Techniques</i>	98
5.2.4	<i>Identifying the Practical Issues</i>	99
5.2.5	<i>Deciding Ethical Issues</i>	103
5.3	PILOT STUDIES AND EVALUATION	104
5.3.1	<i>Changes to Documents</i>	104
5.3.2	<i>Evaluate, Interpret, and Present the Data</i>	106
5.3.3	<i>Evaluation Summary</i>	113
5.3.4	<i>Changes to MUIP Caused by the Evaluation</i>	114
5.4	SUMMARY	117
CHAPTER 6 CONCLUSIONS AND FURTHER WORK		118
6.1	REVIEW OF THE RESEARCH	118
6.2	CONTRIBUTIONS	120
6.3	FURTHER WORK	122
APPENDIX A: PLML V1.1		124
APPENDIX B: PLML V1.2		125
APPENDIX C: USE CASE SPECIFICATIONS AND MAIN SCENARIOS		126
APPENDIX D: DATABASE SCHEMA IN SQL STATEMENT		146
APPENDIX E: HANDY HINTS FOR EVALUATING THE MUIP		149
APPENDIX F: INFORMATION SHEET FOR EVALUATING THE MUIP		150
APPENDIX G: INFORMED CONSENT FORM FOR MUIP EVALUATION		151
APPENDIX H: OBSERVATION FORM (OBSERVATIONAL PROTOCOL) FOR MUIP EVALUATION		152
APPENDIX I: TASK SCENARIOS FOR EVALUATING THE MUIP		155
APPENDIX J: QUESTIONNAIRE FOR EVALUATING THE MUIP		157
REFERENCES		159

List of Figures

1.1	Mode Cursor pattern from Welie's pattern collection.....	2
1.2	Intriguing Branches pattern from Tidwell's pattern collection.....	4
1.3	Steps in the research approach.....	7
3.1	Damask's proposed user interface.....	50
3.2	IdealXML.....	51
4.1	Use case diagram for MUIP.....	66
4.2	A model of Client-Server system.....	70
4.3	Different tiers of the system.....	71
4.4	ER model of the database.....	73
4.5	DBDL for setting up the OurUser relation.....	74
4.6	SQL for creating the table OurUsers.....	74
4.7	Proposed Class Diagram of MUIP.....	78
4.8	Proposed main interface of MUIP.....	79
4.9	Pattern Editor.....	80
4.10	Forces View.....	81
4.11	Forces field of Pattern Editor.....	81
4.12	Pattern Content View.....	82
4.13	Pattern Finder.....	82
4.14	Versioning patterns.....	83
4.15	Annotations.....	84
4.16	Adding relationships and Type view.....	84
4.17	Collection View.....	85
4.18	Importing pattern in RTF.....	86
5.1	The interface of MUIP with the changes made.....	115

List of Tables

2.1	Alexandrian pattern form.....	10
2.2	GOF pattern form.....	12
2.3	Comparison between guidelines & patterns.....	16
2.4	Summary of the relationship types of OO design patterns.....	18
2.5	Salingarios' relationship types.....	18
2.6	Mullet and Welie's relationship types.....	19
2.7	Shuemmer's relationship types.....	19
2.8	Relationship types from CHI'03.....	20
2.9	Comparing the major pattern forms from different domains.....	21
3.1	Summary of features of four specifications.....	38
3.2	Summary of tools.....	40
3.3	Features for a pattern management tool.....	61
4.1	Mappings between the main features and the use cases.....	65
4.2	Comparison between PLML v1.1 & PLML v1.2.....	69
4.3	Features implemented.....	87
5.1	Techniques.....	93
5.2	Summary of techniques.....	94
5.3	Summary of participants.....	100
5.4	Result table of database.....	103
5.5	Sub-aspects of evaluation question with main source of data.....	103
5.6	Final questionnaire for the evaluation.....	105
5.7	Results of observation protocol.....	107
5.8	Analysis of data of scenario 3.....	108
5.9	Observation notes for scenario 3.....	109
5.10	Results of closed-ended questions.....	111
5.11	Results of open-ended questions.....	112
5.12	Results of database logging.....	113

Chapter 1 Introduction

1.1 Background

From ordinary life, people or organizations have recognised and defined various rules, guidelines and standards to support achieving a good solution for a real design problem. Following these principles, designers can facilitate the development process and control the quality of the design to achieve the proposed design goal easily. For example, a group of builders can construct a house of good quality based on predefined templates in a reasonably short period (Alexander *et al.*, 1977).

Templates emerge from recognised principles of good design. They form, as it were, a pattern for others to follow. The concept of a “pattern” was first proposed by the architect, Christopher Alexander, in his PhD thesis which was subsequently published as the book *Notes on the Synthesis of Form* (1964). In a later book, *A Pattern Language* (Alexander *et al.*, 1977), the authors provide a definition of a pattern and introduce a detailed collection of patterns as a language in the urban architectural domain. A pattern is not invented, but it is identified from the fundamental principles of a good design. A pattern is a narrative description, which represents a design problem and a solution for the design problem within a specific context. A pattern not only points out an invariant solution to a problem with various goals and constraints, but also explains the relationship between them and how the pattern balances the constraints. The idea of architectural patterns is to capture knowledge of good architectural design and document this knowledge in a way that can be used as common vocabularies by the community. The aim of Alexander’s patterns is to help people build a town or a building which has “quality without a name” (Alexander *et al.*, 1979). More details about Alexander’s patterns will be introduced in Section 2.1.

In 1987, the software engineering community adopted the pattern concept from the architectural domain. Beck & Cunningham (1987) outlined a pattern language in object-oriented design and summarised five patterns for user interface design at the OOPSLA-87 workshop in Orlando. The authors described an experiment where novice designers of Smalltalk succeeded in designing their own Smalltalk user interface after mastering some basic Smalltalk user interface concepts from those five patterns. The initial success in

interface design motivated the authors to write a pattern language for object-oriented design. The OOPSLA workshops and software engineering community continued to discuss and exchange software patterns after 1987. More details about software patterns will be described in Section 2.2.

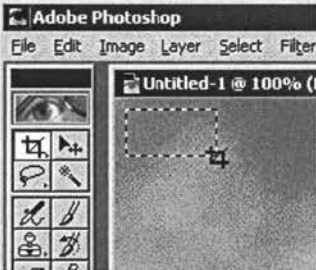
Mode Cursor	
Author	Martijn van Welie
Problem	The user is creating or modifying an object and needs to know which edit function is selected.
Principle	Immediate Feedback (Feedback)
Context	In many direct manipulation applications the users first selects a tool/function, thus entering a special mode/state, and then works on an object. Since such applications usually offer many functions to create or modify objects.
Forces	<ul style="list-style-type: none">• Not every function may have an icon or shape.• Completing a function may cause several intermediate states which may also need to be shown.• The user needs immediate feedback on which function was selected, i.e. which mode/state the system is in.
Solution	Show the interface state in the cursor. The interface state changes many times during interaction, for instance when a function is selected or when an action such as dragging is performed. Therefore, show the current state to the user by changing the cursor. The cursor can be changed to an icon or some other shape that gives feedback about the current interface state. Change the cursor back to a neutral cursor if the function is completed or deactivated.
Rationale	The cursor gives extra feedback about the active function. The user watches the cursor when performing a function so it is the most appropriate place on the screen to give feedback i.e. the user does not need to look at another portion of the screen. The solution increases satisfaction and may decrease errors.
Examples	

Figure 1.1 Mode Cursor pattern from Welie’s pattern collection (2004a).

In 1986, the earliest related reference in the human computer interface (HCI) literature to the Alexandrian pattern concept appears in Norman and Draper’s book, *User Centered System Design* (1986). However, the HCI community did not pay much attention to patterns until the workshop “Putting it All Together: Pattern Languages for Interaction Design” at CHI’97, much later than the software engineering community. Since then, different researchers have developed many UI patterns and pattern collections. Some pattern collections are disseminated in related workshops (Welie *et al.*, 2002) whilst others are published on the web (Tidwell, 1999; Welie, 2004b; Tidwell, 2002; Graham, 2003b; Laakso, 2003) or in

books (Borchers, 2001; Duyne *et al.*, 2003; Graham, 2003a). Although the pattern concept originates in Alexander's work (1964), pattern authors prefer to use their own forms when writing UI patterns. As a result, there exist many variant pattern forms among these collections. Figure 1.1 and Figure 1.2 illustrate two different patterns from Welie's and Tidwell's pattern collections. These two patterns are written in distinct pattern forms. In Chapter 2 various major pattern forms will be analysed.

A collection of patterns might be seen as a pattern language. Todd *et al.* (2004) investigated the factors which determine whether a collection could be considered a pattern language. Some elaborate collections cannot as yet be considered a pattern language as they do not meet the criteria. For this reason, the generic word "collection" is used to refer to a set of patterns in this thesis, even when the set of patterns could be classed as a pattern language.

Borchers (2001) has claimed that there is a high level of agreement on the need to construct a pattern language. The author believes that there will be a common form for HCI patterns, which derives from current various HCI pattern forms and unifies them. Welie *et al.* (2000) first used XML (Extensible Markup Language) to create a standard and consistent pattern format for publishing HCI patterns in their pattern catalog website. Later, some researchers advised the use of XML to document patterns for pattern tools (Borchers, 2001; Greene *et al.*, 2002). The CHI 2003 Pattern Workshop produced a significant outcome for the structure of HCI patterns in the specification of the Pattern Language Markup Language (PLML) (Fincher, 2003). PLML uses XML to document pattern data content in a consistent form based on the standardised XML DTD or Schemas, which can be found in Appendix A. PLML can be applied to unify the format of existing HCI pattern collections. The details of PLML are described in Chapter 2.

The HCI community applies patterns in solving user interface design tasks. This raises the issue of the creation, management and use of pattern collections. Computer systems can facilitate the activities of pattern composition and consumption. The potential users of such a computer system fall into four categories according to the main tasks they may wish to perform in relation to pattern collections: end users, UI researchers, UI designers, and software developers. The intuitive examples of pattern collections can enhance the knowledge of the end users and aid them to express their requirements to the designers. The UI pattern collections act as the common language to facilitate the communication between end users and designers. For UI researchers, or UI designers, who wish to build their own

patterns or collections based on the work of others, it is necessary to have a suitable tool to access and manipulate the available pattern collections. The knowledge of UI pattern collections enables the software developer to review whether the UI patterns have been applied in the software project properly. This research will focus on the needs of the UI researchers and designers.


Intriguing Branches

A political earthquake in the land of earthquakes (News)

By [aphrael](#)
Fri Jul 25th, 2003 at 09:08:32 PM EST

While the rest of the world focuses on the deaths of the [Brothers Hussein](#), the rumblings of a [political earthquake](#) are threatening to bring [California](#) government to its knees. On Thursday, Lieutenant Governor [Cruz Bustamante](#), prompted by a petition signed by more than [1,600,000](#) people, called a snap election to recall the state's unpopular Democratic Governor, [Gray Davis](#). It is the first recall of a Governor in the United States since [1921](#).

[Full Story](#) (165 comments, 2611 words in story)



From <http://kuro5hin.org>

Use when: The user will be moving along a linear path -- a text narrative, a well-defined task, a slideshow, a Flash movie, etc. But you want to present additional content that's not the main focus of attention. It might be information tangential to a story, as in the example above. It might be supporting text -- examples, explanations of concepts, definitions of terms -- or full-fledged help text. Or it could be hidden functionality, like an "Easter egg."

In any case, you want a graceful way of presenting the content so that it's ignorable by users trying to get something done quickly, but still available to users for whom it's appropriate.

Why: People are curious. If they see something that looks interesting, and they have the time and initiative to check it out, they will. Web surfing would never have become a '90s hobby without this natural curiosity and willingness to follow links into the unknown!

A tradition of creating Intriguing Branches as inline links is already well-established on the Web. But a more interesting use of it might be in functional applications. It's well-known that users tend to ignore stuff labeled as "Help," for instance. But what if you put help-like content behind links (or buttons, or icons) that were labeled in some other way, like "More..."? You can exploit users' natural curiosity to get them into a place where they can learn what they need to learn.

Skillful and playful use of Intriguing Branches can make your interface more fun. It's often a good thing.

How: Starting with a deep understanding of your users, create "doors" into the supplemental content that would appeal to them. These doors might be underlined links (even in desktop applications), headlines, buttons, menu items, icons, or clickable image regions -- it's up to you to figure out how to label them in such a way as to inspire curiosity. There's an art to it. When in doubt, usability-test it with a good sample of your user base.

With particularly obscure affordances, like icons or images, you might want to add tooltips or some other kind of Short Description to inform the user where they might be going when they click on it! (With an Easter egg, though, its very non-obviousness is part of the fun.)

Also, provide an obvious way for the user to get back to their original workflow. The idea is to get them reading the branch content, then going back to what they came for, not to get them stranded in a backwater! Popup windows should provide "Close" buttons; new pages in a browser-like UI should provide "Back" links or buttons.

Figure 1.2 Intriguing Branches pattern from Tidwell's pattern collection (2002).

The CHI 2003 Pattern Workshop (Perspectives on HCI Patterns: Concepts and Tools) is a milestone for HCI pattern research. The workshop had two main goals. The first goal related

to conceptual issues - to identify and share a variety of perspectives on patterns and pattern languages for design activities such as interaction design and HCI design. It is necessary to consider why and how patterns are created, and how and when they can be reused during the design process. For this goal, the workshop aimed (Fincher *et al.*, 2003, p.1045):

- *To identify what is important in the area, what is fundamental to patterns as applied to HCI;*
- *To enhance our understanding of the concept of patterns and pattern languages in relation to other theoretical models used in design;*
- *To work towards the creation of a “map” of the conceptual territory of the pattern endeavour.*

The second goal related to computer tools - to understand how computer systems can efficiently support a variety of activities during building and reusing the patterns and related software components. For this goal, the workshop aimed (Fincher *et al.*, 2003, p.1045):

- *To create a shared understanding of the needs and activities of the diverse groups of people involved in pattern creation and pattern use;*
- *To identify ways in which the capabilities of the computer can enhance and support these activities.*

The research described in this thesis is concerned with identifying the need and activities of UI researchers and designers involved in pattern authoring and pattern consumption, and then conceptualising a pattern management tool to support these needs and activities.

1.2 Motivation and Objectives

The motivation for this research comes from three perspectives. First, for researchers and UI designers who wish to build their patterns or collections based on the work of others, it is necessary to have a suitable pattern tool to access and manipulate the available pattern collections. Ideally, such a tool will be based on a structured pattern structure, such as that specified by PLML. Second, none of the existing UI pattern tools have totally supported PLML to provide a standardised format for pattern authoring and consumption. Last, because the specification of PLML made in CHI 2003 Pattern Workshop is still in the early stages, developing a UI pattern tool, which totally supports PLML, will help evaluate the specification.

Computer systems can facilitate the activities of pattern composition and consumption, and therefore, have been proposed to support the creation and use of patterns or collections. A pattern tool could support the role-specific activities to manage patterns and collections, and the different tasks required by various user groups such as HCI design practitioners, software engineers, and domain experts (Borchers, 2001; Greene *et al.*, 2002; Gaffar *et al.*, 2003; Schuemmer, 2003b).

Although most of the existing pattern tools outline the importance of using a standardised format for patterns, none have totally implemented PLML as yet. Also, since the PLML specification is still in its initial version, there is a need to develop a prototype in order to evaluate the PLML specification. Development of an effective pattern management tool with PLML could greatly contribute to the UI pattern domain.

Based on the above motivations, we need to consider the research question: how can a tool be developed to support the management and use of the collection of UI patterns based on PLML?

The objectives of this research are to investigate the functions required for both the creation and manipulation of pattern collections within a pattern management tool based on PLML and how these can be best supported.

The research aims:

- To identify from the literature the needs and activities of those developing pattern collections;
- To analyse current tools to determine to what extent they meet the needs of those using pattern collections;
- To conceptualise a pattern management tool to manipulate pattern collections;
- To design, implement and evaluate a prototype to support the conceptualisation of a tool;
- To evaluate the specification of PLML.

Related knowledge from human computer interaction and database management is used to conceptualise and implement a pattern management system to manipulate and use UI pattern collections.

1.3 Research Approach

The research approach employed to achieve the objectives described above is illustrated in Figure 1.3 and summarised as follows:

- The research approach focused on the research problem: how to develop a tool to support the management and use of collections of UI patterns based on PLML?
- Based on the research problem and the motivation described in Section 1.2, the literature review, investigates the needs and activities for such a tool. To identify the requirements of the tool, existing specifications were investigated. A number of tools supporting the management of patterns were also evaluated.
- Based on the literature review and analysis of existing tools, a specification for a tool to manage pattern collections has been defined.
- A prototype has been implemented based on the requirements.
- The evaluation phase validates the features of the prototype. Overall, this is an iterative process of specification, prototyping and evaluation.
- A discussion considers to what extent the implementation of the prototype meets the objectives of this research.

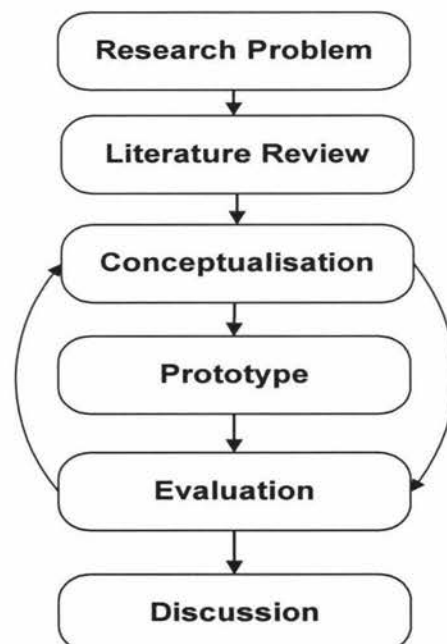


Figure 1.3 Steps in the research approach.

1.4 Outline of Thesis

This thesis consists of six chapters. Chapter 1 is the introductory chapter. Chapter 2 reviews the background of HCI patterns, and describes the types of relationships between patterns and PLML in detail, as well as introducing the categorisation of patterns. Chapter 3 reviews the existing specifications and tools for managing pattern collections. The conceptualisation of a new UI pattern management system is identified. Chapter 4 describes the prototype design of a pattern management tool (MUIP). A revised version of PLML is presented within this chapter. The detailed prototype for managing pattern collections with related implementation issues is also outlined in this chapter. Chapter 5 discusses the evaluation of the project. Finally, Chapter 6 gives an overall conclusion and a description of future work.